SPRINGER SERIES IN COMPUTATIONAL NEUROSCIENCE

Analysis of Parallel Spike Trains

Sonja Grün
Stefan Rotter
Editors

Springer

Springer Series in Computational Neuroscience

Volume 7

Series Editors

Alain Destexhe
CNRS
Gif-sur-Yvette
France

Romain Brette
École Normale Supérieure
Paris
France

For further volumes:
http://www.springer.com/series/8164

Sonja Grün • Stefan Rotter
Editors

Analysis of Parallel Spike Trains

Springer

Editors
Sonja Grün
Laboratory for Statistical Neuroscience
RIKEN Brain Science Institute
2-1 Hirosawa
Wakoshi
Saitama 351-0198
Japan
gruen@brain.riken.jp

Stefan Rotter
Bernstein Center Freiburg
& Faculty of Biology
Albert-Ludwig University
Hansastrasse 9a
79104 Freiburg
Germany
stefan.rotter@biologie.uni-freiburg.de

ISBN 978-1-4419-5674-3 e-ISBN 978-1-4419-5675-0
DOI 10.1007/978-1-4419-5675-0
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010933856

© Springer Science+Business Media, LLC 2010
All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.
The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Cover design: The cover artwork (original in oil pastel and acrylic) was designed and created by Adrián Ponce Alvarez.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Foreword

Close to fifty years ago, when two generations of neuroscientists contributing to this book had not yet been born and I was still happy in high school, George Gerstein, Donald Perkel, and colleagues, in a series of papers, laid the foundations for what was to become the quantitative analysis of the dynamics of neuronal spiking activity and the interactions within and across neuronal networks. In the context of the present textbook, it is instructive to reread some of these papers.

The full abstract of the 1960 *Science* paper (Gerstein 1960) reads:

> *The use of a high-speed digital computer for investigation of neural firing patterns is described. The high sensitivity of the method permits detection of stimulus-response relations buried in a background of spontaneous activity.*

The abstract of the 1964 *Science* paper (Gerstein and Clark 1964) continues:

> *A tungsten microelectrode with several small holes burnt in the vinyl insulation enables the action potentials from several adjacent neurons to be observed simultaneously. A digital computer is used to separate the contributions of each neuron by examining and classifying the waveforms of the action potentials. These methods allow studies to be made of interactions between neurons that lie close together.*

Such studies were indeed made, albeit that their numbers increased only very slowly initially, mostly due to rather formidable problems with the experimental technology involved. With regard to the analysis of data resulting from such experiments, the abstracts of the two companion papers in *Biophysical Journal* 1967 (Perkel et al. 1967a, 1967b) are strikingly explicit and, even in 2010, scaringly timely. In fact, taken together, they read as an ambitious programme manifesto, aimed at establishing a novel, theory-driven data analysis paradigm for network physiology, which, as we now know, it indeed did.

First, on the spiking activity of single neurons:

> *In a growing class of neurophysiological experiments, the train of impulses* ("*spikes*") *produced by a nerve cell is subjected to statistical treatment involving the time intervals between spikes. The statistical techniques available for the analysis of single spike trains are described and related to the underlying mathematical theory, that of stochastic point processes, i.e., of stochastic processes whose realizations may be described as series of point events occurring in time, separated by random intervals. For single stationary spike trains,*

several orders of complexity of statistical treatment are described; the major distinction is that between statistical measures that depend in an essential way on the serial order of interspike intervals and those that are order-independent. The interrelations among the several types of calculations are shown, and an attempt is made to ameliorate the current nomenclatural confusion in this field. Applications, interpretations, and potential difficulties of the statistical techniques are discussed, with special reference to types of spike trains encountered experimentally. Next, the related types of analysis are described for experiments which involve repeated presentations of a brief, isolated stimulus. Finally, the effects of nonstationarity, e.g. long-term changes in firing rate, on the various statistical measures are discussed. Several commonly observed patterns of spike activity are shown to be differentially sensitive to such changes.

Then, on the activity of pairs of neurons and their interactions:

The statistical analysis of two simultaneously observed trains of neuronal spikes is described, using as a conceptual framework the theory of stochastic point processes. The first statistical question that arises is whether the observed trains are independent; statistical techniques for testing independence are developed around the notion that, under the null hypothesis, the times of spike occurrence in one train represent random instants in time with respect to the other. If the null hypothesis is rejected—if dependence is attributed to the trains—the problem then becomes that of characterizing the nature and source of the observed dependencies. Statistical signs of various classes of dependencies, including direct interaction and shared input, are discussed and illustrated through computer simulations of interacting neurons. The effects of nonstationarities on the statistical measures for simultaneous spike trains are also discussed. For two-train comparisons of irregularly discharging nerve cells, moderate nonstationarities are shown to have little effect on the detection of interactions. Combining repetitive stimulation and simultaneous recording of spike trains from two (or more) neurons yields additional clues as to possible modes of interaction among the monitored neurons; the theory presented is illustrated by an application to experimentally obtained data from auditory neurons.

Rereading these abstracts leaves us wondering, what the present book is all about and why, in the year 2010, we would still need it. Indeed, these early studies in the 1960s already formulated the fundamental questions that are still very much with us: How can neuronal interactions be segregated from ongoing background activity? How can the concepts of activity dynamics and correlations be refined and reconciled? And, most of all: What is their role in brain function? It is, therefore, a reassuring thought to have the same George Gerstein contributing a chapter on these issues in this new textbook.

In the 1980s, Computational Neuroscience established itself as a new research field in the neurosciences (Sejnowski et al. 1988; Schwartz 1990; Rumelhart et al. 1986). As Valentino Braitenberg so aptly put it (Braitenberg 1992): “We are convinced that ultimately a satisfactory explanation of thought and behavior will be given in a language akin to that of physics, i.e. in mathematical terms.” Indeed, the theory of neuronal networks explaining basic features of single neurons and simple networks made rapid progress, and many theoreticians were attracted to the field. Thus, over the next decade, many papers, dedicated conferences (e.g., Palm and Aertsen 1987; Aertsen and Braitenberg 1992; Aertsen 1993; Aertsen and Braitenberg 1996), and monographs (e.g., Abeles 1982; Amit 1992; Braitenberg 1984; Palm 1982) explored the foundations of a future theory of brain function. Soon, text books suitable for advanced courses, followed (e.g., Abeles

1991; Dayan and Abbot 2001; Gerstner and Kistler 2002; Hertz et al. 1991; Koch 1998).

Compared to this rapid progress in the development of computational models and theory, progress in the analysis of experimental spike train recordings soon lagged behind. The nonstationarity and irregularity of spiking activity recorded from the brain in action, its considerable variability across repetitions of a task, and the complex interplay of multiple time scales, all taken together constituted very severe problems—and continue to do so—for the adequate handling of the experimental data. As a result, an unambiguous interpretation of the various quantitative measures and the search for meaningful structure in the recorded data all too often remained illusive. I recall that one night, many years ago, I awoke in front of the TV screen after the program had ended. After studying the nonprogram on the screen for a while, I concluded that it was evidently packed with highly interesting spatiotemporal patterns—in any case, I saw them occurring and recurring virtually anywhere, almost at will. Needless to say, though, that it proved a bit hard to establish their statistical and functional significance. Thus, despite some early groundbreaking overviews on advanced data analysis methods for specific areas (e.g., Eggermont 1990; Rieke et al. 1997), it should take another 20 years before this first comprehensive textbook on the state of the art in the analysis of parallel spike train recordings could be produced.

Important techniques are now available for routine usage in the laboratory, and their limits are in most cases well understood. Thus, it is timely to bring this knowledge to the classroom now. Consequently, the present book luckily avoids the repetition of lengthy conceptual discussions from earlier times and, instead, hurries to present the material in a format suitable for the practitioner and the student. The scope of the contributions ranges from the mathematical framework underlying the various methods to practical questions like the generation of random numbers. Closing the cycle, one chapter in the book returns to the earlier cited question on *The use of a high-speed digital computer for investigation of neural firing patterns.* It explains how, with the help of modern computer clusters and high-level parallel programming, we can now compute statistical tests capturing biological constraints to a precision inaccessible by analytical approaches of the past.

However, this book in no way presents an end point to the quest. The authors can only hint at the challenges posed by the advent of massively parallel recording technology. Many such systems are already installed worldwide. Yet, presently they are still mostly used to increase the speed of data collection, rather than utilizing the qualitatively new opportunity to assess the higher-order structure of neuronal assemblies in action. I am confident that in the years to come, the authors will continue in their endeavor and will surprise us with many fascinating new insights into the functioning of the brain. A rewarding prospect, indeed, since the alternative of studying a TV screen outside broadcasting time has meanwhile ceased to be an option.

Bernstein Center Freiburg Ad Aertsen

References

Abeles M (1982) Local cortical circuits. Springer, Berlin
Abeles M (1991) Corticonics. Cambridge University Press, Cambridge
Aertsen A (ed) (1993) Brain theory: spatio-temporal aspects of brain function. Elsevier Science Publ., Amsterdam
Aertsen A, Braitenberg V (eds) (1992) Information processing in the cortex: experiments and theory. Springer, Berlin
Aertsen A, Braitenberg V (eds) (1996) Brain theory: biological basis and computational principles. Elsevier Science Publ., Amsterdam
Amit D (1992) Modeling brain function: the world of attractor neural networks. Cambridge University Press, Cambridge
Braitenberg V (1984) Vehicles: experiments in synthetic psychology. MIT Press, Cambridge
Braitenberg V (1992) Manifesto of brain science. In: Aertsen A, Braitenberg V (eds) Information processing in the cortex. Springer, Berlin
Dayan P, Abbot LF (2001) Theoretical neuroscience. MIT Press, Cambridge
Eggermont JJ (1990) The correlative brain. Springer, Berlin
Gerstein GL (1960) Analysis of firing patterns in single neurons. Science 131:1811–1812
Gerstein GL, Clark WA (1964) Simultaneous studies of firing patterns in several neurons. Science 143:1325–1327
Gerstner W, Kistler WM (2002) Spiking neuron models. Cambridge University Press, Cambridge
Hertz JA, Krogh A, Palmer RG (1991) Introduction to the theory of neural computation. Perseus Publishing, Cambridge
Koch C (1998) Biophysics of computation: information processing in single neurons. Oxford University Press, New York
Palm G (1982) Neural assembilies – an alternative approach to artificial intelligence. Springer, Berlin
Palm G, Aertsen A (eds) (1987) Brain theory. Springer, Berlin
Perkel DH, Gerstein GL, Moore GP (1967a) Neuronal spike trains and stochastic point processes. I. The single spike train. Biophysical J 7:391–418
Perkel DH, Gerstein GL, Moore GP (1967b) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophysical J 7:419–440
Rieke F, Warland D, de Ruyter van Steveninck R, Bialek W (1997) Spikes: exploring the neural code. MIT Press, Cambridge
Rumelhart DE, McClelland JL, the PDP Research Group (1986) Parallel distributed processing, explorations in the microstructure of cognition. MIT Press, Cambridge
Schwartz E (ed) (1990) Computational neuroscience. MIT Press, Cambridge
Sejnowski TJ, Koch C, Churchland PC (1988) Computational neuroscience. Science 241:1299–1306

Preface

Why Data Analysis of Parallel Spike Trains is Needed

The brain is composed of billions of neurons, the elementary units of neuronal information processing. The neocortex, which is critical to most higher brain functions, is a highly complex network of neurons each of which receives signals from thousands of other neurons and projects its own spikes to thousands of other neurons. In order to observe neuronal activity in the active brain, a large variety of recording techniques are being employed: intra-cellular recordings of the membrane potentials of individual neurons, extra-cellular spike recordings from one or more individual neurons, and recordings of signals that measure the activity of populations of neurons either locally as the local field potential, or from larger brain regions via the EEG, MEG, or fMRI. Any particular choice of the recording technique reflects the hypothesis the researcher has in mind about the mechanisms of neuronal processing.

The focus on spike recordings from individual neurons imposed on this book implies that one strives to understand the elementary units of neuronal processing. Early electro-physiological experiments were bound to record from single neurons only. The resulting insights are now the basis for the "classical" view of sensory coding: firing rates are modulated in a feed-forward hierarchy of processing steps. Signals from sensory epithelia are assumed to eventually converge to cortical detectors for certain combinations of stimulus features. Highly specific percepts would be represented by the elevated firing of a single nerve cell or a small group of neurons. Due to a number of conceptual shortcomings, however, it has been seriously questioned whether such a scheme qualifies as a universal method for representation in the brain.

It was Donald Hebb (1949) who first demonstrated the conceptual power of a brain theory based on cell assemblies. Inspired by Hebb's influential writings, and driven by more recent physiological and anatomical evidence in favor of a distributed network hypothesis, brain theorists constructed models that rely on groups of neurons, rather than single nerve cells, as the *functional* building blocks for representation and processing of information. Despite conceptual similarities, such concepts of neuronal cooperativity differ in their detailed assumptions with respect to the spatio-temporal organization of the neuronal activity.

To understand the principles of coordinated neuronal activity and its spatio-temporal scales, it is obligatory to observe the activity of multiple single-neurons simultaneously. Due to recent technological developments in recording methodology, this can easily and regularly be done. Coordinated activity of neurons is only visible in (wide-sense) correlations of their respective spike trains, which typically admit no simple interpretation in terms of fixed synaptic wiring diagrams. Rather, it became evident that the correlation dynamics apparent in time-resolved multiple-channel measurements reflect variable and context-dependent coalitions among neurons and groups of neurons. Thus, the analysis of simultaneously recorded spike trains allows us to relate concerted activity of ensembles of neurons to behavior and cognition. Different analysis approaches are thereby relevant to distinguish different or even complementary spatio-temporal scales. Analysis of parallel spike trains is the logical next step to improve our understanding of the neuronal mechanisms underlying information processing in the brain.

Purpose of the Book

Solid data analysis is the most important basis for the meaningful evaluation and reliable interpretation of experiments. Although the technology of parallel spike train recording using multielectrode arrangements has been available for decades now, it only recently gained wide popularity among electro-physiologists. However, the relevant literature for the analysis of cooperative phenomena in parallel spike trains is unfortunately scattered across many journal publications, and we know of the pain to compile a useful reader for our students. Reinforced by the considerable interest in courses and schools on data analysis on these issues (offered by us and other colleagues), the idea came up to collect the knowledge on spike train analysis that is available in the literature in a single book.

This first textbook on spike train analysis concentrates on the analysis of *parallel* spike trains. The focus is on concepts and methods of correlation analysis (synchrony, patterns, rate covariance), combined with a solid introduction into approaches for single spike trains, that represent the basis of correlation analysis. Any specific method of analysis must make assumptions about the nature of the data it operates on. It happens that those who apply any particular analysis procedure are not well informed about the underlying assumptions, because they are only implicit or not discussed at all in the literature. Many traditional analysis methods are based on firing rates obtained by trial-averaging, and some of the assumptions for such procedures to work can indeed be ignored without serious consequences.

The situation is different for correlation analysis, the result of which may be considerably distorted if certain critical assumptions for the method to work are violated by the data. Therefore, we have put efforts in stating explicitly the underlying assumptions of the various methods, and giving methods at hand allowing to test if the assumptions are indeed fulfilled. In addition, we supply the reader also with stochastic modeling tools and numerical methods to test if the analysis method chosen serves the intended purpose. In the still ongoing discussion on the relevance of

temporal coordination of spikes it is even more important to be aware of prerequisites and pitfalls to avoid potentially wrong interpretations of data due to violations of critical assumptions.

Intended Audience

This book is, on the one hand, written for the practitioners of data analysis, irrespective of whether they use available software packages or whether they implement the procedures themselves. On the other hand, some parts of the book feature abstract mathematical background necessary for a deeper understanding of the formal foundations of point processes that underlie most discussed analysis methods. Thus, the book is directed to research neurophysiologists, data analysts, and theoretical neuroscientists, but also to graduate and postgraduate students in systems neuroscience, computational neuroscience, and related neuroscience schools and courses.

Organization of the Book

There are *very* many different approaches to the analysis of parallel spike trains, and the field is far from having agreed upon a set of canonical tools everybody knows and everybody uses in the lab. In fact, almost every major publication written since the seminal papers by George Gerstein and his colleagues in the 1960s featured a new variant of multiple spike train analysis, matched to the specific experimental design and associated data format, and catering to the particular question their authors had in mind. Doing justice to all of them would require an encyclopedic work, not necessarily helpful in providing guidance for the inexperienced. Instead we concentrate here on a selection of methods that we consider most sound and also most inspiring with respect to their actual purpose: elucidating some aspect of brain function on the level of individual neurons and their interactions.

Part I of the book first gives an introduction to stochastic point processes which is the appropriate mathematical object to represent a spike train, or a statistical ensemble of spike trains, to be precise (Chap. 1). In this framework, estimating the firing rate of a neuron becomes a well-defined statistical procedure (Chap. 2). Second-order properties of spike trains associated with irregularity in time or variability across trials can also be dealt with consistently if embedded into point process theory (Chap. 3). Peculiarities of signals obtained in a periodic setting are then dealt with separately (Chap. 4).

Part II concentrates on the pairwise comparison of spike trains and first introduces classical cross-correlation techniques, both in the time domain and in the frequency domain (Chap. 5). Time scale issues of correlation analysis are the topic of a separate article (Chap. 6). A general framework for the comparison of spike trains in terms of abstract distance functions is then developed (Chap. 7). Finally,

a method is described that can, based on pairwise comparisons, identify clusters of similar spike trains in larger populations (Chap. 8).

Part III focuses on multineuron spike configurations and first describes the concept of precise spatio-temporal firing patterns (Chap. 9). Patterns of near-simultaneous spikes are then discussed, along with statistical issues associated with their detection (Chap. 10). An information-geometric perspective on spike patterns in discrete time (Chap. 11) is finally complemented by a continuous-time framework to deal with correlations of higher order in parallel spike trains (Chap. 12).

Part IV introduces population-based analyses and gives first an introduction to the classical theory of Shannon information (Chap. 13). Encoding and decoding of neuronal population activity is then discussed in the light of information theory (Chap. 14). Finally, different types of multivariate point process models are employed to characterize neuronal ensemble spike trains (Chap. 15).

Part V deals with practical issues that are critical for neuronal data analysis. Methods to numerically simulate stochastic point processes with well-defined statistical properties are first explained (Chap. 16). Alternatively, data recorded from real neurons can be manipulated to obtain surrogate spike trains that preserve some properties and destroy others (Chap. 17). For testing complex hypotheses about neuronal spike trains, the bootstrap method can be effectively employed (Chap. 18). For numerical simulation, manipulation of recorded data, or bootstrap procedures, the availability of high-quality pseudo-random numbers is of great importance (Chap. 19). The last contribution in the book deals with the effective use of parallel and distributed computing facilities for data analysis in a neuroscience laboratory (Chap. 20).

How to Read This Book

The book can be read at two levels. Researchers whose main interest is in applications should read enough of each chapter to understand purpose and scope of the method presented and to be able to explore their own data with the software provided by the authors (see below). The theoretically oriented reader will find some mathematical details in most chapters, and in all cases pointers are given to the primary literature on the subject.

Software Download

For supplying the reader with the analysis software kindly provided by various authors of this book, we set up a website http://www.apst.spiketrain-analysis.org that links to all the websites of the various contributors. This has the advantage—in contrast to supplying the readers with the software on CD—that the software can stay at the original websites of the respective authors, and can there be updated as needed. Also links can easily be changed if necessary. If you are also willing to provide your software to the public, please let us know, and we add the links.

Acknowledgements

We would like to thank all the people who supported us in the adventure to prepare this book, who contributed to the book, and in particular also all the referees who critically read the manuscripts and helped to further improve the various contributions. Thus we thank (in alphabetical order): Denise Berger, Christian Borgelt, Stefano Cardanobile, Markus Diesmann, Junji Ito, Bjørg Kilavik, Sebastien Louis, Martin Nawrot, Antonio Pazienti, Alexa Riehle, Hideaki Shimazaki, Benjamin Staude, Tom Tetzlaff, and the unknown referees who made the book possible due to their positive evaluations of the proposal for this book.

Wako-Shi, Japan — Sonja Grün
Freiburg, Germany — Stefan Rotter

Contents

Contributors

Moshe Abeles The Leslie and Susan Gonda (Goldschmied) Multidisciplinary Brain Research Center, Bar-Ilan University, Ramat-Gan 52900, Israel, abelesm@mail.biu.ac.il

Ad Aertsen Neurobiology and Biophysics, Institute of Biology III, Faculty of Biology, Albert-Ludwig University, Schänzlestrasse 1, 79104 Freiburg, Germany, aertsen@biologie.uni-freiburg.de

Shun-ichi Amari RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan, amari@brain.riken.jp

Go Ashida Department of Biology, University of Maryland, 1210 Biology–Psychology Building, College Park, MD 20742, USA, ashida@umd.edu

Christian Borgelt Intelligent Data Analysis and Graphical Models Research Unit, European Centre for Soft Computing, Edificio Científico-Tecnológico, Campus Mieres, c/ Gonzalo Gutiérrez Quirós s/n, E-33600 Mieres (Asturias), Spain, christian.borgelt@softcomputing.es

Stefano Cardanobile Bernstein Center Freiburg, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany, cardanobile@bccn.uni-freiburg.de

Catherine E. Carr Department of Biology, University of Maryland, 1210 Biology–Psychology Building, College Park, MD 20742, USA, cecarr@umd.edu

Michael Denker Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan, mdenker@brain.riken.jp

Markus Diesmann Laboratory for Computational Neurophysics, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan, diesmann@brain.riken.jp and Bernstein Center for Computational Neuroscience, Albert-Ludwig University, Freiburg, Germany

Jos J. Eggermont, Department of Physiology and Pharmacology, Department of Psychology, University of Calgary, Calgary, Alberta, Canada, eggermon@ucalgary.ca

Denny Fliegner Department of Nonlinear Dynamics, Max-Planck-Institute for Dynamics and Self-Organization, Bunsenstr. 10, 37073 Göttingen, Germany, fliegner@nld.ds.mpg.de

George Gerstein Department of Neuroscience, University of Pennsylvania, 215 Stemmler Hall, 3405 Hamilton Walk, Philadelphia, PA 19104-6074, USA, george@mulab.physiol.upenn.edu

Ilan N. Goodman Department of Electrical and Computer Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA, igoodman@alumni.rice.edu

Sonja Grün Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan, gruen@brain.riken.jp

Don H. Johnson Department of Electrical and Computer Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA, dhj@rice.edu

Sebastien Louis Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan, sgr-louis@brain.riken.jp

Cesare Magri Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy, cesare.magri@gmail.com

Fernando Montani Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy, fmontani@gmail.com

Abigail Morrison Functional Neural Circuits Group, Bernstein Center Freiburg & Faculty of Biology, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany, abigail@brain.riken.jp

Martin Paul Nawrot Neuroinformatics and Theoretical Neuroscience, Institute of Biology, Freie Universität Berlin, Königin Luise Straße 1-3, 14195 Berlin, Germany, martin.nawrot@fu-berlin.de

Giuseppe Notaro Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy, giuseppe.notaro@iit.it

Stefano Panzeri Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy, stefano.panzeri@iit.it

Rasmus S. Petersen University of Manchester, Faculty of Life Sciences, Oxford Road, Manchester M13 9PT, UK, r.petersen@manchester.ac.uk

Hans Ekkehard Plesser Dept. of Mathematical Sciences and Technology, Norwegian University of Life Sciences, PO Box 5003, 1432 Aas, Norway, hans.ekkehard.plesser@umb.no and RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan

Keith P. Purpura Division of Systems Neurology and Neuroscience, Department of Neurology and Neuroscience, Weill Cornell Medical College, 1300 York Avenue, New York, NY 10065, USA, kpurpura@med.cornell.edu

Stefan Rotter Bernstein Center Freiburg & Faculty of Biology, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany, stefan.rotter@biologie.uni-freiburg.de

Christopher J. Rozell School of Electrical and Computer Engineering, Georgia Institute of Technology, 777 Atlantic Dr. NW, Atlanta, GA 30332-0250, USA, crozell@ece.gatech.edu

Shigeru Shinomoto Department of Physics, Graduate School of Science, Kyoto University, Sakyo-ku, Kyoto 606-8502, Japan, shinomoto@scphys.kyoto-u.ac.jp

Benjamin Staude Bernstein Center Freiburg, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany, staude@bcf.uni-freiburg.de

Tom Tetzlaff Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences, Drøbakveien 31, 1432 Ås, Norway, tom.tetzlaff@umb.no

Wilson Truccolo Department of Neuroscience and Brown Institute for Brain Science, Brown University, S.E. Frank Hall, 185 Meeting Street, Office 583, Providence, RI 02912, USA, wilson_truccolo@brown.edu and Department of Neurology, Massachusetts General Hospital, Boston, MA 02114, USA

Valérie Ventura Statistics Department and the Center for the Neural Basis of Cognition, Carnegie Mellon University, 5000 Forbes avenue, Baker Hall 132, Pittsburgh, PA 15213, USA, vventura@stat.cmu.edu

Jonathan D. Victor Division of Systems Neurology and Neuroscience, Department of Neurology and Neuroscience, Weill Cornell Medical College, 1300 York Avenue, New York, NY 10065, USA, jdvicto@med.cornell.edu

Carl van Vreeswijk Laboratoire de Neurophysique et Physiologie, CNRS UMR 8119, Université Paris Descartes, 45 rue des Saints Pères, 75270 Paris Cédex 06, France, cornelis.van-vreeswijk@univ-paris5.fr

Hermann Wagner Institute for Biology II, RWTH Aachen, Mies-van-der-Rohe-Strasse 15, 52074 Aachen, Germany, wagner@bio2.rwth-aachen.de

Bernd Wiebelt Bernstein Center Freiburg, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany, wiebelt@bcf.uni-freiburg.de

Part I
Basic Spike Train Statistics: Point Process Models

Chapter 1
Stochastic Models of Spike Trains

Carl van Vreeswijk

Abstract This chapter reviews the theory of stochastic point processes. For simple renewal processes, the relation between the stochastic intensity, the inter-spike interval (ISI) distribution, and the survival probability are derived. The moment and cumulant generating functions and the relation between the ISI distribution and the autocorrelation is investigated. We show how the Fano factor of the spike count distribution depends on the coefficient of variation of the ISI distribution. Next we investigate models of renewal processes with variable rates and C_{V2}, which is often used to assess the variability of the spike train in this case and compare the latter to the C_V. The second half of the chapter deals with stochastic point processes with correlations between the intervals. Several examples of such processes are shown, and the basic analytical techniques to deal with these processes are expounded. The effect of correlations in the ISIs on the Fano factor of the spike count and the C_{V2} are also explored.

1.1 Introduction

Neurons in the vertebrate central nervous system fire highly irregularly: Upon stimulation, there is a large trial-to-trial variability, not only in the precise timing of the spikes, but also in the total number of spikes that the stimulus elicits. This behavior can be replicated in deterministic model neurons if a sufficiently noisy input is injected into the cell. However, the analysis of the spike statistics for such neurons is very difficult. For simple models, such as the nonleaky integrate and fire neuron, one can determine spike statistics if the input has constant mean and temporally uncorrelated fluctuations (Gerstein and Mandelbrod 1964), but beyond this, analytical solutions are few and far between.

C. van Vreeswijk (✉)
Laboratoire de Neurophysique et Physiologie, CNRS UMR 8119, Université Paris Descartes, 45 rue des Saints Pères, 75270 Paris Cédex 06, France
e-mail: cornelis.van-vreeswijk@univ-paris5.fr
url: http://www.neurophys.biomedicale.univ-paris5.fr/~carl/

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_1, © Springer Science+Business Media, LLC 2010

If one nevertheless wants to investigate how spike train statistics affects a measure of neuronal activity, one has to come up with models of spike trains that are susceptible to analytical treatment or at least can be simulated easily. An important class of such models are stochastic models of spike trains. In these models one does not aim to capture the behavior of the neurons between spikes, but instead one posits a probabilistic model for the timing of the spikes, based on the spike history.

The simplest such model is the Poisson process. In a Poisson process the probability of a spike between time t and $t + \delta t$ is, for sufficiently small δt, simply given by $R\delta t$, independently of the position of previous spikes. Here R is the firing rate. Since the probability of a spike at some time t is totally independent of the history, the Poisson process is the easiest stochastic model for spike trains to analyze.

However, because the spiking is completely independent of the history, Poisson neurons behave in some aspects quite differently from real neurons. For example, they have no refractory period. Therefore, if one has developed some measure of the spike statistics, one cannot investigate how this measure depends on the properties of the refractory period using Poisson processes. Hence it is useful to consider more complex model for stochastic spike trains. There is of course a price to pay for considering more complex models; namely the analysis is rather more involved than that for Poisson processes.

Nevertheless an enormous amount of work has been done on the analysis of stochastic models of spike trains. In this chapter I will review some of these models and of the analytical techniques that can be used to investigate them. The analysis of stochastic point processes is a subject that has been investigated for decades and has a vast literature, from which one cannot hope to cover even a substantial part of the topics most relevant to neuroscience in a single chapter. In this chapter we will therefore only consider spike trains from a single neuron. Issues involving the interplay of several spike trains will regretfully have to be ignored here. For a review of these and other topics, one should consult some of the monographs on point processes such as (Andersen et al. 1994; Cox 1962; Cox and Isham 1980).

Before we turn to these processes, let us introduce some notation that will be used throughout this chapter: Spikes occur at times $t_1, t_2, t_3, \ldots$. Inter-spike intervals will be denoted by Δ, where Δ_n is given by $\Delta_n = t_n - t_{n-1}$. The firing rate is denoted by $R(t)$ or simply by R if the rate is constant, and the coefficient of variation will be written as $\mathrm{C_V}$.

1.2 Renewal Processes

As mentioned above, the simplest model for a stochastic spike train is the Poisson process in which in each sufficiently small interval δt there is a probability $R\delta t$ that there is a spike. (For simplicity, we will assume here that the rate is constant.) Consider the survival probability, $S(\Delta)$, which is the probability that, if there is a spike at time t_0, the next spike will occur after time $t_0 + \Delta$. Obviously, $S(0) = 1$, and $S(\Delta + \delta t)$ is equal to $S(\Delta)$ times the probability that there was no spike between times $t_0 + \Delta$ and $t_0 + \Delta + \delta t$, or $S(\Delta + \delta t) = S(\Delta)(1 - R\delta t)$. From this we obtain

$$S(\Delta) = \mathrm{e}^{-R\Delta}.$$

This allows us to calculate the inter-spike interval (ISI) distribution $\varrho(\Delta)$, where $\varrho(\Delta)\delta t$ is the probability that there is an ISI between Δ and $\Delta + \delta t$. The ISI distribution is given by

$$\varrho(\Delta) = -\frac{\mathrm{d}S(\Delta)}{\mathrm{d}\Delta} = R\mathrm{e}^{-R\Delta}.$$

This gives us another way to look at the Poisson process with constant rate. The Poisson process with constant rate is a process where the ISIs are independently drawn from an exponential distribution.

This suggests models for spike trains in which the ISIs are independently drawn from some other distribution ϱ. Such models are known as renewal processes, because the time of the next spike only depends on the time of the last spike. As soon as the latter is known, the previous history becomes irrelevant for the statistics of later spikes.

1.2.1 Some General Properties of Renewal Processes

For a renewal process with constant rate R, or stationary ISI distribution, ϱ has to satisfy

$$\int_0^\infty \varrho(\Delta)\,\mathrm{d}\Delta = 1 \quad \text{and} \quad \int_0^\infty \Delta\varrho(\Delta)\,\mathrm{d}\Delta \equiv \langle\Delta\rangle = R^{-1}.$$

(In principle one could have that $\int_0^\infty \varrho(\Delta)\,\mathrm{d}\Delta = 1 - p$, with $0 < p < 1$. This should be interpreted to mean that after each spike there is a probability p that the neuron stops spiking. We will not consider this case here.)

Using the definition of the coefficient of variation, $\mathrm{C_V} = \sqrt{\langle\Delta^2\rangle/\langle\Delta\rangle^2 - 1}$, one has that $\mathrm{C_V}$ is given by

$$\int_0^\infty \Delta^2\varrho(\Delta)\,\mathrm{d}\Delta = \left(\mathrm{C_V^2} + 1\right)R^{-2}.$$

For the Poisson process, we used that the probability of a spike between time t and $t + \delta t$ is equal to $R\delta t$, independently of when the previous spike occurred. For an arbitrary renewal process, the probability of a spike between times t and $t + \delta t$ will depend on the time of the last spike. If this spike occurred at time t_0, we can use as an alternative description for the renewal process that the probability of a spike in the interval $[t, t + \delta t]$ is equal to $H(t - t_0)\delta t$. The function H is called the stochastic intensity or hazard rate (Collett 2003).

Following the same logic as for the Poisson process, for the survival function, we can write $S(\Delta + \delta t) = S(\Delta)[1 - H(\Delta)\delta t]$. Thus the survival probability S and the ISI distribution ϱ can be expressed in terms of the stochastic intensity H as

$$S(\Delta) = \exp\left(-\int_0^\Delta H\left(\Delta'\right)\mathrm{d}\Delta'\right) \quad \text{and} \quad \varrho(\Delta) = H(\Delta)\exp\left(-\int_0^\Delta H\left(\Delta'\right)\mathrm{d}\Delta'\right).$$

Conversely, H can be expressed in terms of ϱ as

$$H(\Delta) = \frac{\varrho(\Delta)}{S(\Delta)} = \frac{\varrho(\Delta)}{\int_\Delta^\infty \varrho(\Delta')\,\mathrm{d}\Delta'}.$$

Let us now consider the probability distribution for n consecutive events, $\varrho_n(t_n)$, the probability density for the nth spike at t_n, given that $t_0 = 0$. Clearly $\varrho_1(t_1) = \varrho(t_1)$. The probability density $\rho_n(t_n)$ is the probability density for t_{n-1} times the probability density for $\Delta_n = t_n - t_{n-1}$, integrated over t_{n-1}:

$$\varrho_n(t_n) = \int_0^{t_n} \varrho_{n-1}(t_{n-1})\varrho(t_n - t_{n-1})\,\mathrm{d}t_{n-1}. \tag{1.1}$$

From this we see that the Laplace transform (see box) $\varrho_{n,\mathrm{L}}$ of ϱ_n can be expressed in terms of the Laplace transform ϱ_L of ϱ as

$$\varrho_{n,\mathrm{L}}(s) = \varrho_\mathrm{L}(s)\varrho_{n-1,\mathrm{L}}(s) = \varrho_\mathrm{L}^n(s),$$

where

$$\varrho_\mathrm{L}(s) = \int_0^\infty \varrho(t)\mathrm{e}^{-st}\,\mathrm{d}t \quad \text{and} \quad \varrho_{n,\mathrm{L}}(s) = \int_0^\infty \varrho_n(t)\mathrm{e}^{-st}\,\mathrm{d}t.$$

The Laplace transform is also convenient to determine the moments of the ISI distribution (Tuckwell 1988). Since $\mathrm{d}^k\varrho_\mathrm{L}(s)/\mathrm{d}s^k = (-1)^k \int_0^\infty t^k\varrho(t)\exp(-st)$, the kth moment μ_k defined as $\mu_k \equiv \langle\Delta^k\rangle$ can be written as

$$\mu_k = (-1)^k \frac{\mathrm{d}^k}{\mathrm{d}s^k}\varrho_\mathrm{L}(s)\bigg|_{s=0} = \varrho_\mathrm{L}^{(k)}(0),$$

where $\varrho_\mathrm{L}^{(k)}$ denotes the kth derivative of ϱ_L. If the moments satisfy $\mu_k/k! \le CM^k$, for some finite C and M, we can for $|s| < 1/M$ write ϱ_L as

$$\varrho_\mathrm{L}(s) = \sum_{k=0}^\infty (-1)^k \mu_k \frac{s^k}{k!}. \tag{1.2}$$

Here we have used that $\mu_0 \equiv \langle\Delta^0\rangle = 1$.

Alternatively, we can write ϱ_L as

$$\varrho_\mathrm{L}(s) = \exp\left(\sum_{k=1}^\infty (-1)^k \kappa_k \frac{s^k}{k!}\right),$$

where the cumulants κ_k are given by $\kappa_k = (-1)^k\,\mathrm{d}^k \log[\varrho_\mathrm{L}(s)]/\mathrm{d}s^k|_{s=0}$.

The Laplace Transform

In probability theory the Laplace transform is an extremely useful tool. Here we will very briefly review some of its most important features. For more details, see (Davies 2002; Spiegel 1965; Wikipedia 2009).

The Laplace transform is closely related to the Fourier transform, and the Laplace transform $f_{\mathrm{L}}(s)$ of a function $f(t)$ is defined as

$$f_{\mathrm{L}}(s) = \mathcal{L}\big[f(t)\big](s) \equiv \int_0^\infty f(t)\mathrm{e}^{-st}\,\mathrm{d}t.$$

It can be shown that if two continuous functions f and g have the same Laplace transform, $f_{\mathrm{L}}(s) = g_{\mathrm{L}}(s)$, then f and g are the same, $f(t) = g(t)$.

The Laplace transform is a linear operator: $\mathcal{L}[af(t)+bg(t)](s) = af_{\mathrm{L}}(s)+bg_{\mathrm{L}}(s)$.

The Laplace transform of the convolution $f * g$ of two functions f and g, defined as $f * g(t) \equiv \int_0^t f(t')g(t-t')\,\mathrm{d}t'$, is the product of their Laplace transforms f_{L} and g_{L},

$$\mathcal{L}\big[f * g(t)\big](s) = f_{\mathrm{L}}(s)g_{\mathrm{L}}(s).$$

The inverse Laplace transform $\mathcal{L}^{-1}$ is given by the complex inversion integral

$$f(t) = \mathcal{L}^{-1}\big[f_{\mathrm{L}}(s)\big](t) \equiv \frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty} f_{\mathrm{L}}(s)\mathrm{e}^{st}\,\mathrm{d}s,$$

where the integration is over the line $c+ix$ in the complex plain, with x going from $-\infty$ to $+\infty$, and the real value c has to be large enough, so that all singularities of f_{L} lie to the left of the line $s = c+ix$. The inverse Laplace transform is often difficult to calculate, but most books on the Laplace transform have extensive tables of Laplace pairs f and f_{L} (see, for example, Davies 2002; Spiegel 1965). Together with the manipulations on the functions f and f_{L} in Table 1.1A, these suffice in most cases to find the inverse Laplace transform. Table 1.1B gives some of the important Laplace pairs.

Table 1.1 **A**: Effect of some transformations of f on its Laplace transform. **B**: Laplace pairs f and f_{L} for some elementary functions. Θ is the Heaviside function: $\Theta(x) = 1$ for $x \geq 0$ and $\Theta(x) = 0$ for $x < 0$

A. Transformations		B. Laplace pairs	
$f(t)$	$f_{\mathrm{L}}(s)$	$f(t)$	$f_{\mathrm{L}}(s)$
$\int_0^t f(t')\,\mathrm{d}t'$	$f_{\mathrm{L}}(s)/s$	1	$\frac{1}{s}$
$f'(t)$	$f(0) + sf_{\mathrm{L}}(s)$	t^n	$\frac{n!}{s^{n+1}}$
$\mathrm{e}^{-at}f(t)$	$f_{\mathrm{L}}(s+a)$	$\sin(\omega t)$	$\frac{\omega}{s^2+\omega^2}$
$f(at)$	$\frac{1}{a}f_{\mathrm{L}}(\frac{s}{a})$	$\cos(\omega t)$	$\frac{s}{s^2+\omega^2}$
$\Theta(t-a)f(t-a)$	$e^{-as}f_{\mathrm{L}}(s)$	$\exp(-at)$	$\frac{1}{s+a}$
$tf(t)$	$-f'_{\mathrm{L}}(s)$	$\frac{t^{\nu-1}}{\Gamma(\nu)}\exp(-at)$	$\frac{1}{(a+s)^\nu}$

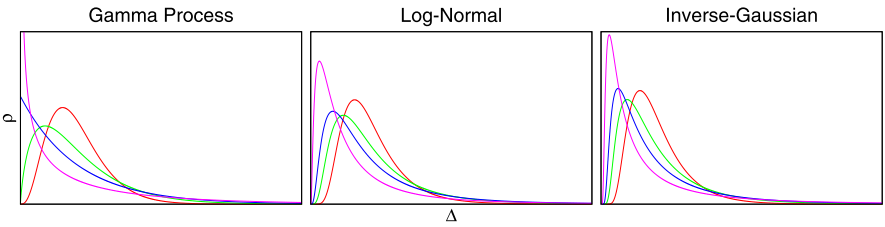

Fig. 1.1 ISI distributions for Gamma, Log-Normal, and inverse Gaussian processes. For each class, the distribution is shown for $C_V = 0.5$ (*red*), $C_V = 0.75$ (*green*), $C_V = 1$ (*blue*), and $C_V = 1.5$ (*purple*)

1.2.2 Examples of Renewal Processes

Besides the Poisson process, processes that are commonly used to model spike trains are the Gamma process, the Log-Normal process, and the Inverse Gaussian Process. The ISI distribution is shown for these processes for different values of the C_V in Fig. 1.1. Note that the Poisson process is a special case of the Gamma process, namely the Gamma process with $C_V = 1$. Table 1.2 show some the properties of these processes discussed. For other distributions and their properties, the reader is referred to (Balakrishnan and Nevzorov 2003; Evans et al. 2002).

The Gamma process is often used to model spike trains because it is the easiest non-Poisson process to analyze. However, if the C_V is larger than 1, the ISI distribution diverges as Δ approaches 0. It is not unlikely that this leads to pathological results, and it is not recommended that Gamma processes be used to model spike trains of bursty neurons. If one still insists in using them for bursty neurons, one should at least, in a limited number of cases, check whether the results also hold for other renewal processes.

The Log-Normal process is a process in which the logarithm of the ISI is drawn from a Gaussian distribution. This makes it very easy to generate spike trains with a Log-Normal distribution: Using standard software, one draws values y_n independently from a normalized Gaussian distribution and sets $\Delta_n = \exp(\sigma y_n + \mu)$. It is straightforward to show that with this procedure, $\langle \Delta^k \rangle = \exp(k\mu + k^2\sigma^2/2)$, so that for a given rate and coefficient of variation, one has to use $\sigma = \sqrt{\log(C_V^2 + 1)}$ and $\mu = -\log(R) - \frac{1}{2}\log(C_V^2 + 1)$. A disadvantage of the Log-Normal distribution is that there is no closed-form expression for its Laplace transform ϱ_L or for the distribution ρ_n of n consecutive intervals. This limits how much can be done analytically with this distribution.

The final ISI distribution we will mention here is the Inverse Gaussian or Wald distribution. This distribution is obtained by considering Brownian motion of a variable x with a positive drift: x satisfies $x(0) = 0$ and $dx/dt = \nu + \sigma\eta(t)$, where $\eta(t)$ is a stochastic variable with Gaussian white noise statistics. If Δ is the first time x exceeds the value 1, the distribution of Δ is the Wald distribution. One can show that $\langle \Delta \rangle = \nu^{-1}$ and $\langle \Delta^2 \rangle - \langle \Delta \rangle^2 = \sigma^2 \nu^{-3}$ (Gerstein and Mandelbrod 1964), so that

Table 1.2 Properties of some commonly used renewal processes. The ISI distribution, ϱ, its Laplace transform, ϱ_{L}, the distribution of the sum of n intervals, ϱ_n and the cumulants, κ_k are given, except for the Log-Normal distribution, for which there is no analytical expression for ϱ_{L} and ϱ_n and for which the moments μ_k instead of the cumulants κ_k are given

Poisson process:	
$\varrho(\Delta) = R\mathrm{e}^{-R\Delta}$,	$\varrho_{\mathrm{L}}(s) = \frac{R}{R+s}$,
$\varrho_n(\Delta) = R\frac{(R\Delta)^{n-1}}{(n-1)!}\mathrm{e}^{-R\Delta}$,	$\kappa_k = (k-1)!/R^k$.
Gamma process:	
$\varrho(\Delta) = \gamma R\frac{(\gamma R\Delta)^{\gamma-1}}{\Gamma(\gamma)}\mathrm{e}^{-\gamma R\Delta}$,	$\varrho_{\mathrm{L}}(s) = (\frac{\gamma R}{\gamma R+s})^{\gamma}$,
$\varrho_n(\Delta) = \gamma R\frac{(\gamma R\Delta)^{n\gamma-1}}{\Gamma(n\gamma)}\mathrm{e}^{-\gamma R\Delta}$,	$\kappa_k = \gamma(k-1)!/[\gamma R]^k$,
$\mathrm{C_V} = \sqrt{1/\gamma}$.	
Log-Normal process:	
$\varrho(\Delta) = \frac{1}{\sqrt{2\pi}\sigma\Delta}\exp(-\frac{(\log\Delta-\mu)^2}{2\sigma^2})$,	$\mu_k = \exp(k\mu + k^2\sigma^2/2)$,
$R = \exp(-\mu-\sigma^2/2)$,	$\mathrm{C_V} = \sqrt{\exp(\sigma^2)-1}$.
Inverse Gaussian process:	
$\varrho(\Delta) = \frac{\mathrm{C_V^{-1}}}{\sqrt{2\pi R\Delta^3}}\exp(-\frac{\mathrm{C_V^{-2}}}{2}\frac{(R\Delta-1)^2}{R\Delta})$,	$\varrho_{\mathrm{L}}(s) = \exp(\mathrm{C_V^{-2}}[1-\sqrt{1+\mathrm{C_V^2}s/R}])$,
$\varrho_n(\Delta) = \frac{n\mathrm{C_V^{-1}}}{\sqrt{2\pi R\Delta^3}}\exp(-\frac{\mathrm{C_V^{-2}}}{2}\frac{(R\Delta-n)^2}{R\Delta})$,	$\kappa_k = \mathrm{C_V^{-2}}\frac{(2k)!}{2^k k!}[\mathrm{C_V^2}/R]^k$.

we need to use $\nu = R$ and $\sigma = \mathrm{C_V}\sqrt{R}$ to get the desired rate and coefficient of variation. We will not derive this distribution here, but note that its form is specified in Table 1.2.

The Laplace transform can also be derived analytically. The distribution ϱ_n can be inferred for the Brownian model: If we initialize x at $x(0) = 0$ and let it evolve according to the Brownian motion with positive drift, the first spike occurs the first time that x reaches 1, the second when x reaches 2, etc. So the nth spike occurs the first time the x reaches n. This means that ϱ_n has the same shape as ρ, except that R has to be replaced by R/n and $\mathrm{C_V}$ by $\mathrm{C_V}/\sqrt{n}$.

ISIs drawn from an Inverse Gaussian distribution can be generated efficiently using the following recipe (for details, see Chhikara and Folks 1989): Draw y_n from a normalized Gaussian distribution. Calculate $x_n = 1 + \mathrm{C_V^2}y_n^2/2 - \sqrt{[1+\mathrm{C_V^2}y_n^2/2]^2 - 1}$. Draw p_n from a uniform distribution between 0 and 1. If $p_n < 1/(1+x_n)$, take $\Delta_n = x_n/R$, else take $\Delta_n = 1/(Rx_n)$.

1.2.3 Autocorrelation

For renewal processes, the autocorrelation of the spikes is simply related to ISI distribution. The autocorrelation A can be written as $A(\tau) = R\delta(\tau) + \tilde{A}(\tau)$, where $R\delta(\tau)$ is the joint probability density of a spike at time t and *the same* spike at time $t+\tau$, while $\tilde{A}(\tau)$ is the joint probability density of a spike at time t and *another* spike at time $t+\tau$. The probability density of a spike k occurring at time t is R,

while the probability density of spike $k+n$ occurring at time $t+\tau$, given that spike k occurred at time t, is equal to $\varrho_n(\tau)$, so the joint probability density of a spike at time t and the nth next spike at time $t+\tau$ is equal to $R\varrho_n(\tau)$. For $\tau \geq 0$, the joint probability $\tilde{A}(\tau)$ of a spike occurring at time t and another spike at time $t+\tau$ is just the sum of the probability densities of a spike at time t and the nth spike at time $t+\tau$ for all $n > 0$. Thus,

$$\tilde{A}(\tau) = R \sum_{n=1}^{\infty} \varrho_n(\tau).$$

Using (1.1) for ϱ_n, this can be written as

$$\tilde{A}(\tau) = R\varrho(\tau) + \int_0^{\tau} \varrho\big(\tau'\big)\tilde{A}\big(\tau - \tau'\big)\,\mathrm{d}\tau'.$$

From this we obtain for the Laplace transform $\tilde{A}_{\mathrm{L}}$ of $\tilde{A}$

$$\tilde{A}_{\mathrm{L}}(s) = \frac{R\varrho_{\mathrm{L}}(s)}{1-\varrho_{\mathrm{L}}(s)}.$$

1.2.4 Spike Count and Fano Factor

Because the spike count distribution is often available in experimental data, it is useful to consider the probability of observing n spikes if the spikes are counted in a time window of duration T, in renewal processes. To determine this, we first have to specify how the time window is chosen. The simplest case, the so-called ordinary renewal process, is when the process is started at time $t = 0$, the beginning of the counting period. In this case the probability density for the first spike occurring at time t_1 is $\varrho(t_1)$, while the probability density for the nth spike occurring at time t_n is $\varrho_n(t_n)$. The survival probability $S_n(t)$, the probability that, at time t, the neuron has not yet spiked n times is given by $S_n(t) = \int_t^{\infty} \varrho_n(t')\,\mathrm{d}t'$, which has a Laplace transform $S_{n,\mathrm{L}}(s) = [1 - \varrho_{n,\mathrm{L}}(s)]/s = [1 - \varrho_L^n(s)]/s$. The probability $P(n,T)$ of observing n is simply the probability that $t_n < T$ and $t_{n+1} > T$, $P(n,T) = S_{n+1}(T) - S_n(T)$, which has a relatively simple Laplace transform,

$$P_{\mathrm{L}}(n,s) = \frac{[1-\varrho_{\mathrm{L}}(s)]}{s}\varrho_{\mathrm{L}}^n(s).$$

The second important case is the equilibrium renewal process. Here we consider a process that has been started at time $t \ll 0$, while spikes are counted between times $t = 0$ and $t = T$. Let $t_0 < 0$ be the time of the last spike before we start counting. For a given t_0, the probability density for the first spike to occur at time t_1, given that the last spike occurred at t_0, is $\varrho(t_1 - t_0)$, while the probability for a spike between t_0 and $t_0 + \delta t$ is $R\delta t$. Hence the probability density, $\tilde{\varrho}_1(t_1)$ for the first at t_1 is $\tilde{\varrho}_1(t_1) = R\int_{-\infty}^{0} \varrho(t_1 - t_0)\,\mathrm{d}t_0 = RS_1(t_1)$. The probability density for the nth spike

in an equilibrium renewal process, $\tilde{\varrho}_n(t_n)$, is given by $\tilde{\varrho}_n(t_n) = \int_0^{t_n} \tilde{\varrho}_1(t_1)\varrho_{n-1}(t_n - t_1)\,\mathrm{d}t_1$. Note that for the Poisson process, $\varrho(t) = \tilde{\varrho}(t) = R\exp(-Rt)$, so that the equilibrium renewal process and the ordinary renewal process have the same spike statistics. From $\tilde{\varrho}_n(t)$ the survival functions $\tilde{S}_n$ and their Laplace transforms $\tilde{S}_{n,\mathrm{L}}$ are readily determined. Using $\tilde{S}_{n,\mathrm{L}}$, we obtain for the Laplace transform of $P(n,T)$ that

$$P_{\mathrm{L}}(n,s) = \frac{[1-\varrho_{\mathrm{L}}(s)]^2}{\mu_1 s^2 \varrho_{\mathrm{L}}(s)}\varrho_{\mathrm{L}}^n(s) \quad \text{for } n \geq 1$$

and $P_{\mathrm{L}}(0,s) = 1/s - [1-\varrho_{\mathrm{L}}(s)]/\mu_1 s^2$. Here $\mu_1 = R^{-1}$ is the average ISI interval.

For the Poisson process, $P(n,T)$ is given by $P(n,T) = (RT)^n \exp(-RT)/n!$. For most other renewal processes, an analytical expression for $P(n,T)$ is not easily obtained. If the period T over which the spikes are counted becomes large compared to the average ISI, the spike count distribution approaches a Gaussian distribution, for which the asymptotic value of the mean and variance can be calculated. The kth moment $M_k(T)$ of the spike count distribution satisfies $M_k(T) = \sum_{n=1}^{\infty} n^k P(n,T)$ and has a Laplace transform $M_{k,\mathrm{L}}(s)$ which satisfies

$$M_{k,\mathrm{L}}(s) = \sum_{n=0}^{\infty} n^k P_{\mathrm{L}}(n,s) = C(s)\sum_n n^k \varrho_{\mathrm{L}}^n(s),$$

where $C(s) = [1-\varrho_{\mathrm{L}}^n(s)]/s$ for the ordinary renewal process, and $C(s) = [1-\varrho_{\mathrm{L}}^n(s)]^2/s^2\mu_1\varrho_{\mathrm{L}}(s)$ for the equilibrium renewal process. Using $\sum_n nx^n = x/(1-x)^2$ and $\sum_n n^2x^n = (x+x^2)/(1-x)^3$ for $|x|<1$, one obtains for the Laplace transform of the first two moments

$$M_{1,\mathrm{L}}(s) = \frac{\varrho_{\mathrm{L}}(s)}{s[1-\varrho_{\mathrm{L}}(s)]}, \qquad M_{2,\mathrm{L}}(s) = \frac{\varrho_{\mathrm{L}}(s)+\varrho_{\mathrm{L}}^2(s)}{s[1-\varrho_{\mathrm{L}}(s)]^2},$$

for the ordinary renewal process, while for the equilibrium renewal process, they are given by

$$M_{1,\mathrm{L}}(s) = \frac{1}{\mu_1 s^2}, \qquad M_{2,\mathrm{L}}(s) = \frac{1+\varrho_{\mathrm{L}}(s)}{\mu_1 s^2[1-\varrho_{\mathrm{L}}(s)]}.$$

From this we see that for the equilibrium renewal process, the average the spike count M_1 is given by $M_1(T) = RT$. For the ordinary renewal process, we can find the approximate value of $M_1(T)$ for large T by expanding $M_{1,\mathrm{L}}(s)$ for small s using (1.2) for $\varrho_{\mathrm{L}}(s)$, $M_{1,\mathrm{L}}(s) = 1/\mu_1 s^2 + (\mu_2 - 2\mu_1^2)/2\mu_1^2 s + \cdots$. This yields $M_1(T) = RT + (\mathrm{C_V^2}-1)/2 + \cdots$. Notice that if $\mathrm{C_V} \neq 1$, the difference in the average spike count for the ordinary and equilibrium renewal processes stays finite as T is increased and approaches $(\mathrm{C_V^2}-1)/2$.

For the Laplace transform of the second moment, we obtain for small s, $M_{2,\mathrm{L}}(s) = 2/\mu_1^2 s^3 + (2\mu_2 - 3\mu_1^2)/\mu_1^3 s^2 + \cdots$ and $M_{2,\mathrm{L}}(s) = 2/\mu_1^2 s^3 + (\mu_2 - \mu_1^2)/\mu_1^3 s^2 + \cdots$ for the ordinary and equilibrium renewal processes, respectively. Thus, for large T, the second moment of the spike count distribution is given by $M_2(T) = R^2T^2 + (2\mathrm{C_V^2}-1)RT$ for the ordinary renewal process and $M_2(T) = R^2T^2 +$

$C_V^2 RT$ for the equilibrium renewal process. The difference in the second moment for the two processes increases as $(C_V^2 - 1)RT$. Perhaps surprisingly, the variance in the spike count, $\sigma_N^2(T) = M_2(T) - M_1^2(T)$, is to leading order the same for both processes and given by $\sigma_N^2(T) = C_V^2 RT$. For renewal processes, the Fano factor of the spike count, F_N, defined as the spike count variance divided by its mean, satisfies, for large T, $F_N = C_V^2$. It should be emphasized however that this result only holds in the large T limit. For short time window, the Fano factor of the spike count can differ substantially from the square of the coefficient of variation.

1.2.5 Variable Rates

So far we have only considered renewal processes with a constant firing rate $R(t) = R$. In many cases we will be interested in stochastic spike trains with variable rates. For example, if we measure from neurons in the primary visual cortex, we will observe a transient response when a visual stimulus is presented. If the same stimulus is presented several times, the precise timing of the spikes will vary for different stimulus presentations, but the firing rates will evolve in roughly the same way for each of the presentations. It is natural to try to model this with a stochastic point process with a rate that changes with time. The simplest point process to model this is the inhomogeneous Poisson process in which the rate $R(t)$ is not constant. For each value of t, we put a spike with probability $R(t)\delta t$ in interval between t and $t + \delta t$. For this process, the probability density $\Pr(t_1|t_0)$ for the next spike at time t_1, given that the last spike occurred at time t_0, is $\Pr(t_1|t_0) = R(t_1)\exp(-\int_{t_0}^{t_1} R(t)\,\mathrm{d}t)$.

There are many possible ways in which renewal processes with variable rate can be formulated (Bagdonavičius and Nikulin 2001), the precise formulation being dependent on how the ISI distribution changes with the firing rate. One straightforward way is the so-called proportional hazard or Cox model (Cox 1975). Assume that for some base-line rate R_0, the stochastic intensity for a spike at time t, given that the last spike was at time t_0, is $H_0(t - t_0)$. We can make the assumption that when the rate varies, the stochastic intensity can be written as $H(t) = \lambda(t)H_0(t - t_0)$ with $\lambda > 0$. The firing rate R depends on λ, and for constant λ, it is relatively straightforward to determine the ISI distribution and hence how R depends on λ. However, if λ varies with time, it is generally not possible to find an analytical expression that relates $\lambda(t)$ and $R(t)$. This severely limits the usefulness of this model for analysis.

A more useful model of renewal processes with changing rates is the process which, in the area of survival analysis, is known as the accelerated failure time (AFT) model (Hougaard 1999). The fundamental assumption here is that if the firing rate is increased, the effect on the ISI distribution is to shrink it along the x axis, without otherwise changing shape. With constant rate R, the ISI distribution ϱ_R is given by $\varrho_R(\Delta) = R\hat{\varrho}(R\Delta)$. This can be generalized to a spike train with variable rate $R(t)$ by writing, for the probability density $\Pr(t_1|t_0)$ for a spike at time t_1, given that the last spike occurred at t_0,

$$\Pr(t_1|t_0) = R(t_1)\hat{\varrho}\left(\int_{t_0}^{t_1} R(t)\,\mathrm{d}t\right).$$

In such a process it is useful to consider a rescaled time variable τ given by $\tau(t) = \int_0^t R(t')\,\mathrm{d}t'$. In this rescaled time, the renewal process with varying rate becomes a renewal process with a constant rate of 1 and ISI distribution $\hat{\varrho}$. Thus, in the rescaled time, the probability density for the n spike $\hat{\varrho}_n$ and the autocorrelation $\hat{A}$ can be calculated as before. This can be used to calculate the probability density $\Pr(t_n|t_0)$ for the nth spike at time t_n, given that the zeroth spike occurred at time t_0:

$$\Pr(t_n|t_0) = R(t_n)\hat{\varrho}_n\big[\tau(t_n) - \tau(t_0)\big].$$

Similarly, the joint probability density $A(t_1, t_0)$ of a spike at time t_1 and a spike at time t_0 is given by

$$A(t_1, t_0) = R(t_1)R(t_0)\hat{A}\big[\tau(t_1) - \tau(t_0)\big].$$

If the firing rate changes, the naive measure for the coefficient of variation confounds the intrinsic variability in spike generation with the effect of the rate change on the ISI distribution. If one knows how the rate R evolves over time, one can make the transformation to the rescaled time $\tau(t)$ and use $\tau_n = \tau(t_n)$ to calculate the mean $\hat{\mu}_1 \equiv \langle \tau_{n+1} - \tau_n \rangle$ and mean square $\hat{\mu}_2 \equiv \langle (\tau_{n+1} - \tau_n)^2 \rangle$ to calculate the $\mathrm{C_V}$ in rescaled time. This can be used as a measure for the intrinsic variability.

If $R(t)$ is not known, we cannot determine the intrinsic variability exactly. However, if we assume that the rate changes on a time scale that is long compared to the average inter-spike interval, we can assume that the rate is approximately constant over the period between three consecutive spikes. This has led Holt et al. (1996) to propose a new measure, $\mathrm{C_{V2}}$, for the variability of the spike train that can be used when the firing rate changes slowly. This measure is defined as

$$\mathrm{C_{V2}} = 2\left\langle \frac{|\Delta_{n+1} - \Delta_n|}{\Delta_{n+1} + \Delta_n} \right\rangle. \tag{1.3}$$

The factor of 2 in this definition is chosen so that for a Poisson process, we have $\mathrm{C_{V2}} = 1$.

Nothing prevents us from using $\mathrm{C_{V2}}$ in models where the rate is constant. This allows us to explore the relationship between $\mathrm{C_{V2}}$ and $\mathrm{C_V}$. It is clear from the definition, (1.3), that $\mathrm{C_{V2}} \leq 2$, and therefore $\mathrm{C_{V2}} \not\equiv \mathrm{C_V}$. For Gamma processes, where $\mathrm{C_V} = 1/\sqrt{\gamma}$, it can be shown that $\mathrm{C_{V2}}$ is given by $\mathrm{C_{V2}} = \Gamma(2\gamma)/\{\gamma[2^{\gamma-1}\Gamma(\gamma)]^2\}$. For Log-Normal processes, where $\mathrm{C_V} = \sqrt{\exp(\sigma^2) - 1}$, $\mathrm{C_{V2}}$ satisfies the equation $\mathrm{C_{V2}} = 4\pi^{-1/2}\int_0^\infty \exp(-x^2)\tanh(\sigma x)\,\mathrm{d}x$. For the Log-Normal process, this gives $\mathrm{C_{V2}} \approx 0.8$ if $\mathrm{C_V} = 1$, while for the Poisson process, $\mathrm{C_{V2}} = \mathrm{C_V} = 1$. This illustrates that different processes with the same $\mathrm{C_V}$ can have different $\mathrm{C_{V2}}$. The two measures $\mathrm{C_V}$ and $\mathrm{C_{V2}}$ capture somewhat different aspects of the ISI variability.

Figure 1.2 shows $\mathrm{C_{V2}}$ plotted against $\mathrm{C_V}$ for Gamma and Log-Normal processes. In both cases we have that $\mathrm{C_{V2}} \approx 2\mathrm{C_V}/\sqrt{\pi}$ for small $\mathrm{C_V}$ and that, as $\mathrm{C_V}$ increases, the slopes of the curves decrease. For increasing $\mathrm{C_V}$, $\mathrm{C_{V2}}$ asymptotically approaches 2. However, in the Log-Normal case, the approach is much slower than

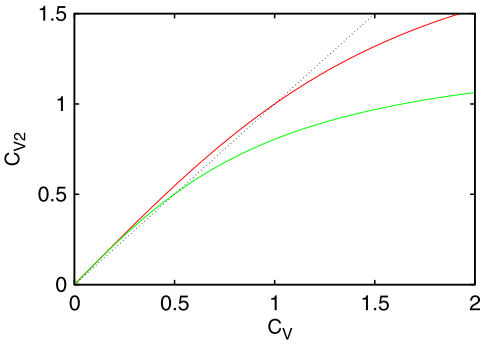

Fig. 1.2 C_{V2} plotted against C_V for Gamma processes (*red*) and Log-Normal processes (*green*). For comparison, the line $x = y$ (*dotted line*) is also shown

that for Gamma processes. Indeed, for Gamma processes, $2 - C_{V2} \approx 2\log 2/C_V^2$, when C_V is large. For the Log-Normal process, under the same condition, $2 - C_{V2} \approx 4/\sqrt{2\pi \log C_V}$.

1.3 Spike-Train Models with Memory

Up to now we have considered only renewal processes, in which the probability of a spike at time t_{n+1} only depends on the time t_n of the previous spike and not on the times of earlier spikes. This makes the analysis relatively simple but ignores biophysical features which may be important to take into account.

For example, with renewal processes we can generate spike trains which look bursty if we assume an ISI distribution with a large coefficient of variation, but in such a process next ISI has the same probability distribution independent of the duration of previous ISI. For real bursting neurons, a short ISI indicates that it is more likely that the neuron is in the burst state and therefore the next ISI has a higher probability of also being short. Another important feature which renewal processes cannot capture is spike adaptation. For neurons with spike adaptation, the level of adaptation is likely to be higher after a short interval, increasing the probability that the next ISI will be long.

To capture these and other phenomena, we have to go beyond renewal processes and consider processes with serial correlations in ISIs. The rest of this chapter gives an outline of how to deal with such processes.

1.3.1 Some Models of Stochastic Spike Trains with Serial Correlation

Before we discuss the analytical approach to stochastic spike train models with correlations, we will first briefly mention a few examples of such processes.

Hidden Markov processes: In a hidden Markov process, the neuron can be in one of N states. If at time t_n the neuron is in state A_n, then the next ISI is drawn from a

distribution $\varrho(\Delta|A_n)$, and at the next spike the state will be A_{n+1} with probability $Q(A_{n+1}|A_n)$.

Processes with memory of the N last ISIs: Interval Δ_n is drawn from a distribution $\varrho(\Delta_n|\Delta_{n-1}, \Delta_{n-2}, \ldots, \Delta_{n-N})$, which depends on the N preceding ISIs.

Processes with adaptation: A stochastic spike train model that mimics the effect of adaptation could be model variable rate in which

$$\Pr(t_{n+1}|t_n, t_{n-1}, t_{n-2}, \ldots) = R(t_{n+1})\varrho\left(\int_{t_n}^{t_{n+1}} R(t)\,\mathrm{d}t\right),$$

where the rate R depends on the level of adaptation a as $R(t) = R_0(1 - a(t))$, and a satisfies

$$\tau_A \frac{\mathrm{d}}{\mathrm{d}t}a = -a + \gamma(1 - a)\sum_k \delta(t - t_k).$$

Doubly stochastic processes: This is a process where the spikes are drawn according to a renewal process with a variable rate R, which itself has a stochastic component. For example, $\Pr(t_{n+1}|t_n, t_{n-1}, \ldots) = R(t_{n+1})\hat{\varrho}[\tau(t_{n+1}) - \tau(t_n)]$, where $\tau(t) = \int_0^t R(t')\,\mathrm{d}t'$, and the rate is given by $R = f(z)$, where z satisfies the stochastic differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}z = -g(z) + \eta(t),$$

where f and g are some suitably chosen functions, and $\eta(t)$ is Gaussian white noise.

1.3.2 General Description of Stochastic Point Processes with Memory

These and many other models belong to the class of models in which the neuron at time of the nth spike is in state a_n and the next ISI, Δ_{n+1}, and state a_{n+1} are drawn from a distribution $\varrho(\Delta_{n+1}, a_{n+1}|a_n)$. Before we use this to develop the theory for stochastic processes with serial correlations, let us verify that this statement is true for the four examples above.

The hidden Markov model is the most straightforward. In this model, the state a_n is the integer state value A_n, and $\varrho(\Delta_{n+1}, a_{N+1}|a_n) = \varrho(\Delta_{n+1}|a_n)Q(a_{n+1}|a_n)$.

In the model where the ISI depends on the N preceding intervals, the "state" a is the N-dimensional vector $(\Delta_n, \Delta_{n-1}, \ldots, \Delta_{n+1-N})$. If we write a_n as $a_n = (\Delta_n^{(1)}, \Delta_n^{(2)}, \ldots, \Delta_n^{(N)})$, $\varrho(\Delta_{n+1}, a_{n+1}|a_n)$ is given by

$$\varrho(\Delta_{n+1}, a_{n+1}|a_n) = \delta\big(\Delta_{n+1} - \Delta_{n+1}^{(1)}\big)\varrho\big(\Delta_{n+1}^{(1)}|\Delta_n^{(1)}, \Delta_n^{(2)}, \ldots, \Delta_n^{(N)}\big) \times \prod_{j=2}^{N} \delta\big(\Delta_{n+1}^{(j)} - \Delta_n^{(j-1)}\big).$$

In the model with adaptation we have that if immediately after the nth spike at time t_n, the level of adaptation is a_n, then $a(t_n + \Delta) = a_n \exp(-\Delta/\tau_A)$ for $\Delta < t_{n+1} - t_n$. Hence we can write $\Pr(t_{n+1}|t_n, t_{n-1}, t_{n-2}, \ldots) = \varrho(t_{n+1} - t_n|a_n)$, where

$$\varrho(\Delta|a_n) = R_0\big[1 - a_n \exp(-\Delta/\tau_A)\big]\varrho\big(R_0\Delta - a_n R_0 \tau_A\big[1 - \exp(-\Delta/\tau_A)\big]\big).$$

Given a_n and Δ_{n+1}, the level of adaptation immediately after the $(n+1)$st spike satisfies $a_{n+1} = \gamma + (1-\gamma)\exp(-\Delta_{n+1}/\tau_A)a_n$. In this model, the level of adaptation, a_n, immediately after the nth spike can be used as the state variable, and

$$\varrho(\Delta_{n+1}, a_{n+1}|a_n) = \varrho(\Delta|a_n)\delta\big(a_{n+1} - a_n(1-\gamma)\mathrm{e}^{-\Delta_{n+1}/\tau_A}\big).$$

In the doubly stochastic process, the rate $R(t)$ can be used the state variable, since if $R_n \equiv R(t_n)$, then $z(t_n) = f^{-1}(R_n)$ is known, and this fully specifies the joint probability distributions of $R(t)$ and $\tau(t) - \tau(t_n)$ for $t > t_n$. This allows us, at least in principle, to calculate the probability density $P(\Delta_{n+1}, R_{n+1}|R_n)$, so that we can use R_n as the state variable.

These examples show that the state a_n is a rather abstract concept. It can refer to the state of neuronal variables directly, such as in the model with adaptation, or indirectly, as in the model with memory of the N previous intervals, or to the state of the neuron's input as in the doubly stochastic model.

1.3.3 Properties of Stochastic Point-Processes with Memory

The transition probability density $Q(a_{n+1}|a_n)$ for the transition from state a_n at the spike n to state a_{n+1} at spike $n+1$ is given by

$$Q(a_{n+1}|a_n) = \int \varrho(\Delta_{n+1}, a_{n+1}|a_n)\,\mathrm{d}\Delta_{n+1}.$$

Using Q, the equilibrium distribution p_{eq} for the states can be determined and is given by $\int Q(a'|a)p_{\mathrm{eq}}(a)\,\mathrm{d}a = p_{\mathrm{eq}}(a')$ and $\int p_{\mathrm{eq}}(a)\,\mathrm{d}a = 1$. The inter-spike interval distribution $\varrho(\Delta)$ satisfies

$$\varrho(\Delta) = \iint \varrho\big(\Delta, a'|a\big)p_{\mathrm{eq}}(a)\,\mathrm{d}a\,\mathrm{d}a'.$$

Assuming that $t_0 = 0$, the probability density $\varrho_n(t_n, a_n|a_0)$ that the nth spike occurs at time t_n and has a state a_n, given that at time $t = 0$ the state is a_0, can be determined iteratively using

$$\varrho_{n+1}(t_{n+1}, a_{n+1}|a_0) = \iint \varrho(t_{n+1} - t_n, a_{n+1}|a_n)\varrho_n(t_n, a_n|a_0)\,\mathrm{d}a_n\,\mathrm{d}t_n,$$

where $\varrho_1(t_1, a_1|a_0) = \varrho(t_1, a_1|a_0)$.

As for the renewal process, this relation is simpler for the Laplace transform $\varrho_{n,\mathrm{L}}$ of ϱ_n, $\varrho_{n+1,\mathrm{L}}(s, a_{n+1}|a_0) = \int \varrho_{\mathrm{L}}(s, a_{n+1}|a_n)\varrho_{n,\mathrm{L}}(s, a_n|a_0)\,\mathrm{d}a_n$.

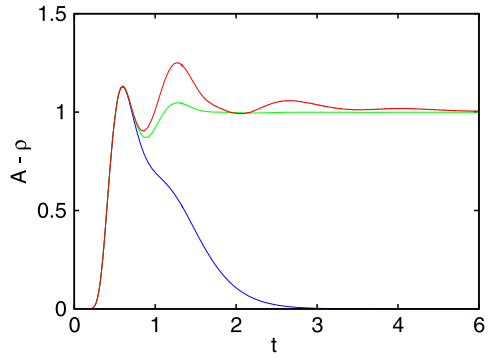

Fig. 1.3 Autocorrelation (*red*) and ISI distribution (*blue*) for a two-state hidden Markov process. In both states the ISI is drawn from an inverse Gaussian distribution with $C_V = 0.3$, while the rate R satisfies $R = 1.5$ in state 1 and $R = 0.75$ in state 2. After each spike, the probability of a state switch is 0.15. The autocorrelation for a renewal process with the same ISI distribution is also shown (*green*)

At this point it is useful to introduce the operator $\mathbf{P}_n(s)$ and the "bra-ket" notation (Cohen-Tannoudji et al. 1977; Dirac 1939; Wikipedia 2009), in which $\mathbf{P}_n(s)|f\rangle$ and $\langle g|\mathbf{P}_n(s)$ are defined as

$$\mathbf{P}_n(s)|f\rangle \equiv \int \rho_{n,\mathrm{L}}\big(s,a'|a\big)f(a)\,\mathrm{d}a \quad \text{and} \quad \langle g|\mathbf{P}_n(s) \equiv \int g\big(a'\big)\rho_{n,\mathrm{L}}\big(s,a'|a\big)\,\mathrm{d}a',$$

while $\langle g|\mathbf{P}_n(s)|f\rangle$ is given by $\langle g|\mathbf{P}_n(s)|f\rangle \equiv \iint g(a')\rho_{n,\mathrm{L}}(s,a'|a)f(a)\,\mathrm{d}a\,\mathrm{d}a'$.

Since $\mathbf{P}_n|f\rangle = \mathbf{P}\cdot\mathbf{P}_{n-1}(s)|f\rangle = \mathbf{P}^n|f\rangle$, where $\mathbf{P}$ is the operator associated with ϱ_{L}, we can write

$$\varrho_{n,\mathrm{L}}(s) = \langle 1|\mathbf{P}_n(s)|p_{\mathrm{eq}}\rangle = \langle 1|\mathbf{P}^n(s)|p_{\mathrm{eq}}\rangle,$$

where we have used $\langle 1|$ to denote the function g with $g(a') = 1$.

It is important to note here that $\langle 1|\mathbf{P}^n(s)|p_{\mathrm{eq}}\rangle \neq \langle 1|\mathbf{P}(s)|p_{\mathrm{eq}}\rangle^n$, so that $\varrho_{n,\mathrm{L}}(s) \neq \varrho_{\mathrm{L}}^n(s)$. Due to the serial correlations, the probability density for the nth spike is not equal to the convolution of n inter-spike interval distributions.

With these results the Laplace transform of $\tilde{A}$ can be written as

$$\tilde{A}(s) = R\sum_{n=1}^{\infty}\langle 1|\mathbf{P}^n(s)|p_{\mathrm{eq}}\rangle = R\langle 1|\mathbf{P}(\mathbf{I}-\mathbf{P})^{-1}|p_{\mathrm{eq}}\rangle,$$

where $\mathbf{I}$ is the identity operator. This implies that $\tilde{A}_{\mathrm{L}}$ does not satisfy $\tilde{A}_{\mathrm{L}}(s) = R\varrho_{\mathrm{L}}(s)/[1-\varrho_{\mathrm{L}}(s)]$. This is illustrated in Fig. 1.3. This figure shows the ISI distribution and the autocorrelation for a spike train generated by a two-state hidden Markov model where in both states the ISI is drawn from an inverse Gaussian distribution, with a different rate in the two states. For comparison, the ISI distribution for a renewal process with the same ISI distribution is also shown. This is clearly different from the autocorrelation of the process with serial correlations.

The operator $\mathbf{P}$ can also be used to characterize the moments and correlations of the ISIs. Because $\langle 1|\mathbf{P}(s_k)\mathbf{P}(s_{k-1})\cdots\mathbf{P}(s_1)\mathbf{P}(s_0)|p_{\rm eq}\rangle = \langle \exp(-\sum_{j=0}^{k} s_j \Delta_{n+j})\rangle$ we can write for the moments of the ISI distribution $\langle \Delta^k\rangle = (-1)^k \langle 1|\mathbf{P}^{(k)}(0)|p_{\rm eq}\rangle$, while the two-point correlations satisfy $\langle \Delta_n \Delta_{n+k}\rangle = \langle 1|\mathbf{P}^{(1)}(0)\mathbf{P}^{k-1}(0)\mathbf{P}^{(1)}(0)|p_{\rm eq}\rangle$. Here $\mathbf{P}^{(k)} = \mathrm{d}^k\mathbf{P}/\mathrm{d}s^k$. Higher-order correlations can be treated analogously.

1.3.4 Dependence of the Fano Factor on Serial Correlations

For renewal processes, we have seen that the Fano factor of the spike count approaches $\mathrm{F}_N = \mathrm{C}_\mathrm{V}^2$ as the observation window becomes much larger than the typical ISI. It should be clear that serial correlations in the ISIs will affect the spike count distribution. An exact derivation of this effect is beyond the scope of this chapter. We will instead give a simple heuristic derivation of its effect on F_N.

Suppose that the process is started at time $t = 0$ and we count spikes up to time T. The average value of the time t_k of the kth spike is given by $\langle t_k\rangle = \sum_{l=1}^{k}\langle \Delta_l\rangle = k\langle\Delta\rangle$. The mean square of t_k satisfies $\langle t_k^2\rangle = \sum_{l=1}^{k}\sum_{m=1}^{k}\langle \Delta_l\Delta_m\rangle$, so that we can write the variance $\sigma^2(t_k)$ of t_k as

$$\sigma^2(t_k) = \sum_{l,m=1}^{k}\left[\langle \Delta_l\Delta_m\rangle - \langle\Delta\rangle^2\right].$$

If the correlation time is finite, there will be a value n_0 such that $\langle \Delta_l\Delta_{l+n}\rangle - \langle\Delta\rangle^2$ is negligible for $|n| > n_0$. Thus, if $k \gg n_0$, we can, for $n_0 < l < k - n_0$, write $\sum_{m=1}^{k}[\langle \Delta_l\Delta_m\rangle - \langle\Delta\rangle^2] \approx \sum_{n=-\infty}^{\infty}[\langle \Delta_l\Delta_{l+n}\rangle - \langle\Delta\rangle^2] \equiv \Sigma_0^2$. This holds for almost all values of l between 1 and k, so that the variance in t_k can be approximated by $\sigma^2(t_k) \approx k\Sigma^2$.

Now assume that $T = K\langle\Delta\rangle$ with $K \gg n_0$. Then the statistics for the Kth spike has a mean $\langle t_K\rangle = T$ and standard deviation $\sigma(t_K) = \sqrt{T/\langle\Delta\rangle}\,\Sigma$. On the other hand, if the spike count is N, the spike times t_N and t_{N+1} satisfy $t_N \le T < t_{N+1}$. For T sufficiently large, we can assume that $t_N = T$. The difference $N - K$ will approximately be given by $N - K = (t_N - t_K)/\langle\Delta\rangle = (T - t_K)/\langle\Delta\rangle$. Thus the average of the spike count satisfies $\langle N\rangle = K = T/\langle\Delta\rangle$, and its standard deviation is given $\sigma(N) = \sigma(t_K)/\langle\Delta\rangle = \sqrt{T/\langle\Delta\rangle^3}\,\Sigma$. Accordingly, the Fano factor of the spike count is given by

$$\mathrm{F}_N \equiv \frac{\sigma^2(N)}{\langle N\rangle} = \sum_{n=-\infty}^{\infty}\left[\frac{\langle \Delta_l\Delta_{l+n}\rangle}{\langle\Delta\rangle^2} - 1\right]. \tag{1.4}$$

Detailed analysis (McFadden 1962) shows that this result is exact in the large T limit. Note also that if the ISI are generated with a renewal process, so that $\langle \Delta_l\Delta_{l+n}\rangle = \langle\Delta\rangle^2$ for $n \neq 0$, we recover $\mathrm{F}_N = \mathrm{C}_\mathrm{V}^2$.

For renewal processes, the Fano factor is well approximated by $\mathrm{F}_N = \mathrm{C}_\mathrm{V}^2$ if $T \gg \langle\Delta\rangle$. In the above derivation we have used that $K \gg n_0$, so for (1.4) to give a

good approximation of the Fano factor, T has to satisfy $T \gg n_0\langle\Delta\rangle$, where $n_0\langle\Delta\rangle$ can be interpreted as the time over which the ISIs are correlated.

It is also important to note that if the correlation time is large, the correlations in the ISIs can have an appreciable effect on the F_N even if $\langle\Delta_l\Delta_{l+n}\rangle - \langle\Delta\rangle^2$ is small for all $n \neq 0$ because many terms will contribute. This makes the comparison between F_N and C_V^2 a very sensitive test for the renewal property. Alternatively, one could argue that calculations of F_N which rely on the renewal property almost certainly will not apply to experimental data.

1.3.5 Sensitivity of C_{V2} to Serial Correlations

In the section on renewal processes with varying rate, we introduced the measure C_{V2} as an estimate for the intrinsic variability in the spiking. But if there are serial correlations in the ISIs, one has to be very careful in interpreting the C_{V2}. Positive (negative) correlations will tend to make consecutive ISIs more (less) similar and therefore increase (decrease) the C_{V2}.

It is instructive to consider this with an example: Consider a point process in which the intervals Δ_n satisfy

$$\log\Delta_n = \alpha\log\Delta_{n-1} + \sqrt{1-\alpha^2}\sigma\xi_n + (1-\alpha)\mu, \tag{1.5}$$

where $-1 < \alpha < 1$, and ξ_n are independently drawn from a normalized Gaussian (Farkhooi et al. 2009).

The distribution ρ of the ISIs is a Log-Normal distribution with parameters μ and σ and is independent of α. It can be shown that $\langle\Delta_l\Delta_{l+n}\rangle = \exp(\alpha^{|n|}\sigma^2)\langle\Delta\rangle^2$. With positive (negative) α, the correlations between consecutive ISIs are positive (negative). To calculate the C_{V2}, it is useful to write $\Delta_n = \exp(\mu + \sqrt{(1+\alpha)/2}x_+ + \sqrt{(1-\alpha)/2}x_-)$ and $\Delta_{n+1} = \exp(\mu + \sqrt{(1+\alpha)/2}x_+ - \sqrt{(1-\alpha)/2}x_-)$, where x_+ and x_- are independent normalized Gaussian variables. With this it is easy to show that C_{V2} is given by $C_{V2} = 4\pi^{1/2}\int_0^\infty \exp(-x^2)\tanh(\sqrt{1-\alpha}\sigma x)\,dx$. Comparing this to the C_{V2} for the Log-Normal renewal process, we see that adding a correlation $[\exp(\alpha\sigma^2) - 1]\langle\Delta\rangle^2$ to consecutive ISIs has, for this model, the same effect on C_{V2} as the transformation $\sigma^2 \to (1-\alpha)\sigma^2$, or $C_V^2 \to (C_V^2+1)^{1-\alpha} - 1$ and keeping the ISIs uncorrelated.

In Fig. 1.4, C_{V2} and F_N are plotted against α for a neuron with $C_V = 1$ ($\sigma = \sqrt{\log 2}$). The coefficient C_{V2} decreases monotonically as α is increased, reaching $C_{V2} = 0$ as α reaches 1. Starting at $\alpha = 0$, the Fano factor first decreases as α is decreases, but for α less than $\alpha \approx -0.6$, the Fano factor increases with decreasing α even though correlation between consecutive intervals continues to decrease. The reason for this is that the correlation between intervals Δ_l and Δ_{l+2n} is positive and growing as α is decreased further.

If we observe a change in C_{V2} in experimental data, this could be the result of either a change in the C_V or change in the serial correlations. These two potential causes imply an opposite effect on, for example, F_N, and one should be careful in the functional meaning that is assigned to this change in C_{V2}.

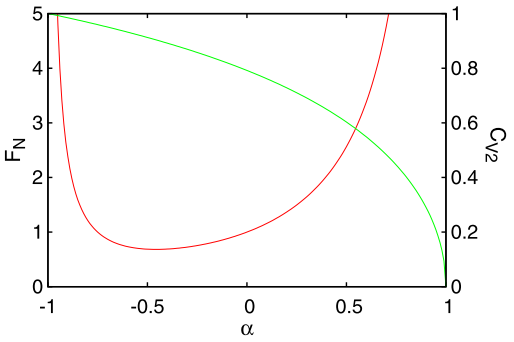

Fig. 1.4 Dependence of C_{V2} and F_N on serial correlations in the ISIs. The Fano factor (*red*, *left-hand scale*) in the large T limit and the C_{V2} (*green*, *right-hand scale*) are plotted for a correlated Log-Normal process with $C_V = 1$ against α

References

Andersen P, Borgan O, Gill R, Keiding N (1994) Statistical models based on counting processes. Springer, Berlin

Bagdonavičius V, Nikulin M (2001) Accelerated life models. Chapman & Hall/CRC, Boca Raton

Balakrishnan N, Nevzorov V (2003) A primer on statistical distributions. Wiley, New York

Chhikara R, Folks L (1989) The inverse Gaussian distribution: theory, methodology, and applications. Dekker, New York

Cohen-Tannoudji C, Diu B, Laoë F (1977) Quantum mechanics. Hermann, Paris

Collett D (2003) Modeling survival data in medical research, 2nd edn. Chapman & Hall/CRC, Boca Raton

Cox R (1962) Renewal theory. Methuen, London

Cox R (1975) Partial likelihood. Biometrica 62:269–276

Cox R, Isham V (1980) Point processes. Chapman & Hall/CRC, Boca Raton

Davies B (2002) Integral transforms and their applications, 3rd edn. Springer, New York

Dirac P (1939) A new notation for quantum mechanics. Proc Roy Soc London A 35:416–418

Evans M, Hastings N, Peacock B (2002) Statistical distributions, 3rd edn. Wiley, New York

Farkhooi F, Strube-Bloss M, Nawrot M (2009) Serial correlation in neural spike trains: Experimental evidence, stochastic modeling, and single neuron variability. Phys Rev E 79:021905

Gerstein G, Mandelbrod B (1964) Random walk models for the spike activity of a single cell. Biophys J 4:41–68

Holt G, Softky W, Douglas R, Koch C (1996) A comparison of discharge variability in vitro and in vivo in cat visual cortex neurons. J Neurophysiol 75:1806–1814

Hougaard P (1999) Fundamentals of survival data. Biometrica 55:13–22

McFadden J (1962) On the lengths of intervals in stationary point processes. J Roy Stat Soc B 24:364–382

Spiegel M (1965) Theories and problems of Laplace transforms. McGraw-Hill, New York

Tuckwell H (1988) Introduction to theoretical neurobiology. Vol. 2. Nonlinear and stochastic theories. Cambridge University Press, Cambridge

Wikipedia (2009) Bra-ket notation. World Wide Web electronic publication. http://en.wikipedia.org/wiki/Bra-ket_notation

Wikipedia (2009) Laplace transform. World Wide Web electronic publication. http://en.wikipedia.org/wiki/Laplace_transform

Chapter 2
Estimating the Firing Rate

Shigeru Shinomoto

Abstract Neuronal activity is measured by the number of stereotyped action potentials, called spikes, elicited in response to a stimulus or the behavioral conditions of an animal. Any nonparametric method for grasping the time-varying rate of spike firing contains a single parameter that controls the jaggedness of the estimated rate, such as the binsize of the time histogram or the bandwidth of the kernel smoother. In most neurophysiological studies, the parameter that determines the interpretation of neuronal activity has been selected subjectively by individual researchers. Recently, theories for objectively selecting the parameter have been developed. This chapter introduces the standard rate estimation tools, such as the peri-stimulus time histogram (PSTH), kernel density estimation, or Bayes estimation, and shows ways of selecting their parameters under the principles of minimizing the mean integrated squared error or maximizing the likelihood. We also sum up the methods in handy recipes that may be useful in practical data analysis.

2.1 Introduction

In the beginning of the last century, Edgar Adrian discerned that a neuron expresses the intensity of a stimulus in the frequency or occurrence rate of stereotyped action potentials (Adrian 1928; Rieke et al. 1997). Since then, the most basic protocol in neurophysiology has been computing the occurrence rate of neuronal action potentials, called spikes or firings, as the correlate of an animal's behavior (Gerstein and Kiang 1960; Johnson 1996; Dayan and Abbott 2001). The rate of neuronal firing, defined in the unit of frequency [Hz], or spikes per second [sp/s], can be computed

S. Shinomoto (✉)
Department of Physics, Graduate School of Science, Kyoto University, Sakyo-ku, Kyoto 606-8502, Japan
e-mail: shinomoto@scphys.kyoto-u.ac.jp
url: http://www.ton.scphys.kyoto-u.ac.jp/~shino/english.html

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_2, © Springer Science+Business Media, LLC 2010

by simply counting the number of spikes and dividing it by the period of the observation interval. Thus the computation of the firing rate is a mere realization of simple descriptive statistics, as exemplified by a census of the population.

However, to grasp the temporal modulation of neuronal activity in relation to sequential behaviors of the animal, one has to go beyond the simple description and treat a hard problem of inferential statistics in estimating an instantaneous rate of firing. If neurons were generating temporally regular spikes, the instantaneous spike rate or the firing frequency may be given by simply inverting the interspike intervals (ISIs). However, this method leads to a messy diagram for highly irregular spike trains recorded from cortical neurons in vivo. If, on the contrary, we spend a long time collecting a large number of spikes for precise estimation of the firing rate, we cannot grasp the fine temporal modulation. This seemingly naive issue may arise in any sophisticated method. Any rate estimation tool has a (hyper)parameter, such as the binsize of a time histogram and the bandwidth of a kernel smoother that controls the jaggedness of the estimate. The estimated rate may become highly fluctuating if the binsize or bandwidth is small, and constant in the opposite limit. In neurophysiological studies, the parameter that critically determines the goodness of rate estimation has mostly been selected subjectively by individual researchers.

Originally, neurons generate spikes according to their inputs in a mostly deterministic manner, and there is no definite entity for the firing rate in the brain. The endeavor to capture the rate of spike occurrence from discrete data can be viewed as a process for compressing information. Rate estimation inevitably depends on the method of information compression and is therefore not determined uniquely for a set of data. Nevertheless, the range of plausible principles and estimation methods is limited, and the rate estimated from a given set of data should not vary among principled methods. In applying statistical principles, one views spikes as sample events derived from a certain underlying probabilistic process and strives to estimate the underlying probability from the spike data. In this respect, methods for selecting the parameter based on statistically plausible principles, such as minimizing the mean integrated squared error (MISE) or maximizing the likelihood, have recently been developed.

In this chapter, we introduce several standard rate estimation tools, the peristimulus time histogram (PSTH) (Gerstein and Kiang 1960; Abeles 1982), the kernel smoother (Parzen 1962; Richmond et al. 1990; Nawrot et al. 1999), and the Bayesian estimation method, and unfold the principles for optimizing rate estimation methods. These methods are summed up in handy recipes that may be useful for neurophysiologists in analyzing their own data. Package programs are available at

http://www.ton.scphys.kyoto-u.ac.jp/~shino/toolbox/english.htm or http://www.apst.spiketrain-analysis.org/.

2.2 Methods for Estimating the Firing Rate

Let us first describe how to construct a PSTH and a kernel density estimation, given a set of spike data obtained from repeated trials.

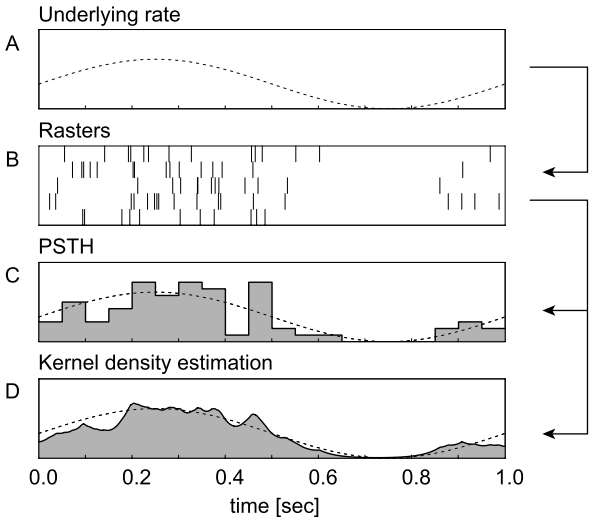

Fig. 2.1 Methods for estimating instantaneous rate. **A**: An underlying spike rate $r(t)$. **B**: Sample spike raster diagrams for five trials. **C**: A peri-stimulus time histogram (PSTH). **D**: A kernel density estimation

2.2.1 PSTH

A histogram is a realization of a simple counting method and can be constructed from a set of spike trains, according to the following instructions (Figs. 2.1A–C).

A method for constructing a PSTH

1. Align n spike trains at the onset or offset of stimuli.
2. Divide an observation period T into intervals of width Δ.
3. Count spikes collected from all trials as k_i for ith bin.
4. Draw a bar at the height $k_i/(n\Delta)$ in a time period of $[(i-1)\Delta, i\Delta]$.
5. Repeat 3 and 4 for each bin from $i = 1$ to $N_b(= T/\Delta)$.

In many neurophysiological papers, the height of the PSTH is represented by the raw spike count per bin. We recommend representing the height of the PSTH in the units of spike rate for one trial, so that integration over the period T gives the number of spikes averaged over trials. Due to this normalization procedure, the heights of PSTHs obtained with different binsize Δ should be approximately equal, and estimates can be compared across different binsizes.

2.2.2 The Kernel Density Estimation

A kernel density estimation can be obtained by blurring each spike with a kernel function, according to the following instructions (Figs. 2.1A, B, and D).

A method for constructing a kernel density estimation

1. Align n spike trains at the onset or offset of stimuli.
2. At every spike, apply a kernel function $f_\Delta(t)$ of bandwidth Δ.
3. Divide the summed function by the number of trials n, or

$$\hat{r}(t) = \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{N_s^j} f_\Delta\left(t - t_i^j\right), \tag{2.1}$$

where t_i^j is the time of the ith spike in the jth trial, and N_s^j is the total number of spikes recorded in the jth trial.

The height of the density estimation measures the spike rate per one trial, similar to the above-mentioned PSTH. The filtering kernel should satisfy the normalization to unit area, $\int f(t)\,dt = 1$. The kernel is normally assumed to be nonnegative, $f(t) \geq 0$, and to have a finite bandwidth defined by the variance that is normally finite, $\Delta^2 = \int t^2 f(t)\,dt < \infty$, and is often chosen to be symmetric, $f(t) = f(-t)$. Many kernel functions satisfy these conditions (Paulin 1992; Nawrot et al. 1999; Paulin and Hoffman 2001; Cherif et al. 2008): A rectangular "boxcar" kernel may give rise to a jagged density function with an appearance similar to that of the PSTH. The width of the boxcar should be $2\sqrt{3}\Delta$ in order to give the variance of Δ^2,

$$f_\Delta(t) = \begin{cases} \frac{1}{2\sqrt{3}\Delta} & \text{for } -\sqrt{3}\Delta \leq t \leq \sqrt{3}\Delta, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

A smooth density estimation can be obtained by using a smooth kernel such as the Gaussian function,

$$f_\Delta(t) = \frac{1}{\sqrt{2\pi}\Delta} \exp\left(-\frac{t^2}{2\Delta^2}\right). \tag{2.3}$$

Alternatively, emphasis may be put on the evidence for spike occurrence from the cusp of the exponential kernel,

$$f_\Delta(t) = \frac{1}{\sqrt{2}\Delta} \exp\left(-\sqrt{2}\left|\frac{t}{\Delta}\right|\right). \tag{2.4}$$

It is noteworthy that exponential functions turn out to be the optimal kernels under the MISE minimization principle for several underlying rates (Koyama and Shinomoto 2004).

Both the Gaussian kernel (2.3) and the exponential kernel (2.4) have infinite support. However, since they decay rapidly, they can be approximated as functions with compact support. For instance, the normalization condition to unit area is in practice not violated even if we cut them off outside the range of $\pm 5\Delta$; the integrated areas

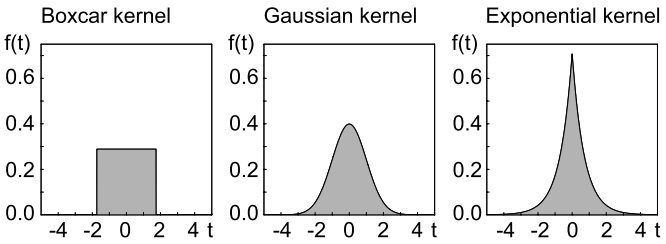

Fig. 2.2 The boxcar, Gaussian, and exponential kernels that give an identical bandwidth, or the variance $\int t^2 f(t)\,dt = 1$

$\int_{-5\Delta}^{5\Delta} f_\Delta(t)\,dt$ become 0.999999 and 0.999151, respectively, for the Gaussian and exponential kernels, thus practically losing nothing in both cases. (See Fig. 2.2.)

2.3 Methods for Optimizing the Rate Estimation

For a single set of spike trains, either a PSTH or a kernel density estimation is not given uniquely, and its shape depends greatly on the choice of the binsize or bandwidth. Figure 2.3 exemplifies three PSTHs constructed from an identical set of spike trains. If the binsize is too small, the time histogram fluctuates greatly, and one cannot discern the underlying spike rate, whereas if it is too large, one cannot capture the time dependence of the rate. There would be an appropriate binsize or bandwidth for each set of spike sequences, based on the goodness-of-fit of a PSTH or a kernel density estimation.

Several plausible principles exist for the goodness-of-fit of the estimator. Here, we introduce two of them and demonstrate formulae that may be practically useful in the application to spike data. One is the principle of minimizing the distance between the estimated rate and the unknown underlying rate measured in the MISE, and another is the principle of maximizing the likelihood function for a given set of data.

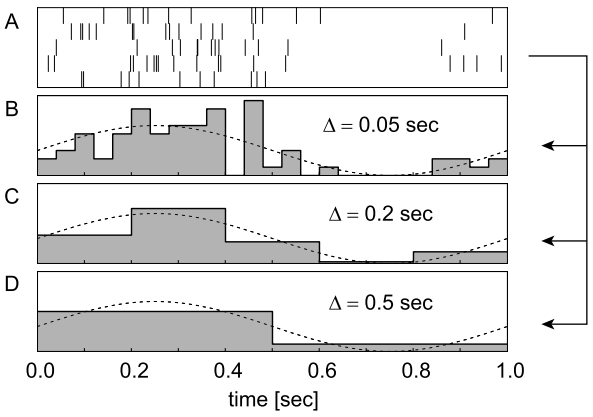

Fig. 2.3 Time histograms with various binsizes. **A**: Spike data given in Fig. 2.1. **B–D**: The time histograms constructed with small, medium, and large binsizes. *Dashed line* represents the underlying rate $r(t)$ from which the spikes were derived

2.3.1 MISE Principle

First, we select the binsize or the bandwidth parameter Δ such that the PSTH or the kernel rate estimator $\hat{r}_\Delta(t)$ best approximates the underlying rate $r(t)$ under the principle of minimizing the MISE,

$$\mathrm{MISE}(\Delta) \equiv \frac{1}{T}\int_0^T E\big(\hat{r}_\Delta(t) - r(t)\big)^2\, dt, \tag{2.5}$$

where E refers to the expectation over different realizations of point events, given $r(t)$.

The definition of MISE contains the underlying rate $r(t)$ itself, which is unknown by nature. Nevertheless, using the spike data as a clue, it is possible to minimize the expected MISE with respect to the parameter Δ (Rudemo 1982; Shimazaki and Shinomoto 2007a, 2007b, 2010). Note that the optimization method can be derived rigorously without assuming anything about the time-dependent rate $r(t)$, such as the continuity of the rate. The only assumption needed in deriving the rule is that spikes are drawn from an inhomogeneous Poisson process in which spikes are independently derived from an underlying rate $r(t)$. Though spikes recorded from a biological neuron correlate in each sequence (Shinomoto et al. 2003, 2005, 2009), spikes collected from a number of independent trials are statistically independent, and the superimposed sequence can be approximated as a single inhomogeneous Poisson process (Snyder 1975; Daley and Vere-Jones 2003; Kass et al. 2005).

2.3.1.1 MISE Optimization of PSTH

Leaving the derivation of the rule for minimizing MISE for the PSTH to the literature (Shimazaki and Shinomoto 2007a), we introduce the method of selecting the binsize Δ in a form of a simple recipe in the following.

A method for selecting the binsize of PSTH

1. Compute the average $\bar{k}$ and the variance v of the spike counts $\{k_i\}$ for all bins, $i = 1, 2, \ldots, N_b$,

$$\bar{k} \equiv \frac{1}{N_b}\sum_{i=1}^{N_b} k_i, \tag{2.6}$$

$$v \equiv \frac{1}{N_b}\sum_{i=1}^{N_b} (k_i - \bar{k})^2, \tag{2.7}$$

and evaluate the cost function (Shimazaki and Shinomoto 2007a),

$$C(\Delta) = \frac{2\bar{k} - v}{(n\Delta)^2}, \tag{2.8}$$

where n is the number of trials, and Δ is the binsize.

2. Repeat 1 by moving the initial binning position to take an average of $C(\Delta)$.
3. Repeat 2 while changing Δ to draw the cost as a function of the bin size.
4. The MISE optimal binsize Δ^* is given by the one that minimizes $C(\Delta)$.

Note here that the variance v is not the unbiased variance, but the biased variance as defined by (2.7). The averaging of $C(\Delta)$ over the initial binning positions is useful for limiting the possible fluctuation due to the finiteness of data. Figure 2.4 displays the raw and averaged cost functions computed for the set of spike data given in Fig. 2.3, demonstrating significant fluctuation for a raw cost function.

By applying the optimization method to spike trains whose intensity does not modulate greatly in time, it might come to pass that the original cost function $C(\Delta)$ computed for n spike sequences does not have a minimum or has a minimum at a binsize comparable to the observation period T. This implies that the data are insufficient for estimating the time-dependent rate. The divergence of Δ^* implies that any PSTH of a finite binsize captures a spurious rate and therefore is worse than simply drawing a constant rate, in the sense of MISE (Koyama and Shinomoto 2004). This cost function evaluation may be useful in justifying the estimation of the time-dependent rate, in particular for discussing the presence or absence of oscillatory activity.

Because a shortage of data underlies the divergence of the optimized binsize, one would consider carrying out more experiments to obtain reliable rate estimation. In making the experimental plan, one may want to estimate how many experiments should be added to secure the PSTH resolution one deems sufficient. This can be done through extrapolating the cost function $C(\Delta)$ obtained with a given number of trials n to the case of a different number of trials m, as

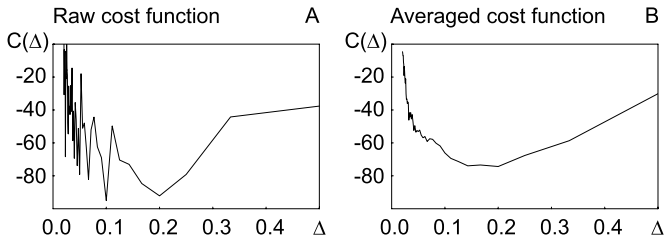

Fig. 2.4 The cost functions $C(\Delta)$ obtained for the set of spike data given in Fig. 2.1B. **A**: Raw cost function obtained for a single set of counting statistics ($\Delta^* = 0.1$). **B**: The cost function averaged over the initial binning positions ($\Delta^* = 0.2$)

$$C(\Delta|m) = \left(\frac{1}{m} - \frac{1}{n}\right)\frac{\bar{k}}{n\Delta^2} + C(\Delta), \tag{2.9}$$

where $\bar{k}$ is the average spike count (2.6) obtained from n sequences. Even if the optimal binsize Δ^* diverged for a given number of spike sequences, implying the incapability of drawing a meaningful PSTH, one can still estimate the number of additional experimental trials needed to make the optimal binsize finite, justifying the evaluation of the time-dependent rate from the data.

It is also possible to extend the estimation method from the bar-graph PSTH to the line-graph PSTH, which generally provides a superior goodness-of-fit (Koyama and Shinomoto 2004). We do not go into the details of the line-graph histogram optimization, which is more complicated than the optimization method presented here (Shimazaki and Shinomoto 2007a).

2.3.1.2 MISE Optimization of Kernel Density Estimation

The same principle of minimizing the MISE can also be applied to the kernel density estimation (Shimazaki and Shinomoto 2007b, 2010). Leaving the derivation of the rule to the literature, we introduce the method of selecting the bandwidth Δ in a form of a simple recipe in the following.

A method for selecting the bandwidth of the kernel density estimator

1. Compute the cost function (Shimazaki and Shinomoto 2007b, 2010)

$$C(\Delta) = \frac{1}{n^2}\sum_{i,j}\phi_\Delta(t_i - t_j) - \frac{2}{n^2}\sum_{i\neq j} f_\Delta(t_i - t_j), \tag{2.10}$$

   where $\phi_\Delta(t_i - t_j) \equiv \int f_\Delta(s - t_i) f_\Delta(s - t_j)\, ds$.
2. Repeat 1 while changing Δ to draw the cost as a function of the bandwidth.
3. The MISE optimal bandwidth Δ^* is given by the one that minimizes $C(\Delta)$.

Note here that the computational cost can be significantly reduced by neglecting unnecessary parts in the double summation in (2.10), based on the practical compactness of either the Gaussian or the exponential kernel. In addition, the convoluted function is given simply as $\phi_\Delta(t) = 1/(2\sqrt{\pi}\Delta)\exp(-t^2/4\Delta^2)$, for the Gaussian kernel, $f_\Delta(t) = 1/(\sqrt{2\pi}\Delta)\exp(-t^2/2\Delta^2)$.

Kernel density estimation with a smooth kernel function is generally much superior to PSTH in its goodness-of-fit to the underlying rate, as demonstrated in Fig. 2.5.

2.3.1.3 TIPS

Application programs for optimizing the PSTH and kernel density estimation methods under the MISE principle were provided by Hideaki Shimazaki at

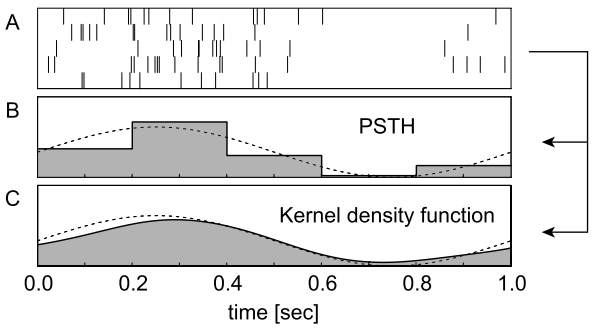

Fig. 2.5 Comparison of the optimized PSTH and optimized kernel density estimator. **A**: Spike data given in Fig. 2.1. **B**: The time histograms of the optimal binsize $\Delta^* = 0.2$. **C**: The Gaussian kernel density estimation of the optimal bandwidth $\Delta^* = 0.1$

http://www.ton.scphys.kyoto-u.ac.jp/~shino/toolbox/sshist/hist.html or http://www.apst.spiketrain-analysis.org/

and

http://www.ton.scphys.kyoto-u.ac.jp/~shino/toolbox/sskernel/kernel.html or http://www.apst.spiketrain-analysis.org/.

By simply copying and pasting a data set of spike times to the application, one can obtain the optimal binsize or bandwidth Δ^* for the PSTH or the kernel, and a list of $\{t, \hat{r}_{\Delta^*}(t)\}$, with which to draw the PSTH or kernel density function. One may also download MATLAB codes for these optimization algorithms.

2.3.2 Likelihood Principle

Another standard optimization principle is maximizing the likelihood. This can be done by treating a rate function $r(t)$ as a distribution function of spike times. If one does not have a prior idea for a particular set of rate functions, one may construct a rate estimator nonparametrically. However, one still needs to determine a (hyper)parameter for the smoothness or jaggedness of the estimator. Here, we introduce a relatively simple Bayesian method (Koyama and Shinomoto 2005; Koyama et al. 2007; Shimokawa and Shinomoto 2009) and show the ways to determine a hyperparameter that plays a role similar to the bandwidth of the filtering kernel.

2.3.2.1 Empirical Bayes Method of Rate Estimation

Let us first consider a time-dependent Poisson process in which spikes are derived randomly in time from a given underlying rate $r(t)$. In this process, the probability for spikes to occur at $\{t_i\} \equiv \{t_1, t_2, \ldots, t_{N_s}\}$ in the period of $t \in [0, T]$ is given by

$$p\big(\{t_i\} \mid r(t)\big) = \left[\prod_{i=1}^{N_s} r(t_i)\right] \exp\left(-\int_0^T r(t)\,dt\right). \tag{2.11}$$

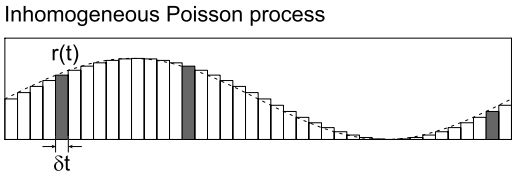

Fig. 2.6 The rate-modulated Poisson process. The probability for a spike to occur in each short interval δt is $r(t)\delta t \ll 1$, and the probability of having no spike is $1 - r(t)\delta t \approx \exp(-r(t)\delta t)$

Here the exponential term is the survivor function that represents the probability that spikes have not occurred in the interval (Cox and Lewis 1966; Daley and Vere-Jones 2003): The probability for a spike to occur in each short interval of δt is $r(t)\delta t \ll 1$ (see Fig. 2.6), and the probability of having no spike from the time t_1 to t_2 is given by

$$\lim_{\delta t \to 0} \prod_m \bigl(1 - r(t_m)\delta t\bigr) = \exp\biggl(-\int_{t_1}^{t_2} r(t)\,dt\biggr).$$

We invert the arguments of the conditional probability (2.11) so that the unknown underlying rate is inferred from the spikes observed. This "inverse probability" can be obtained using the Bayes formula

$$p_\beta\bigl(r(t) \mid \{t_i\}\bigr) = \frac{p(\{t_i\} \mid r(t))\, p_\beta(r(t))}{p_\beta(\{t_i\})}. \tag{2.12}$$

As a Bayesian prior distribution of $r(t)$, we incorporates the tendency of the estimated rate to be flat by penalizing the large gradient, $|dr(t)/dt|$,

$$p_\beta\bigl(r(t)\bigr) \propto \exp\biggl[-\beta \int_0^T \biggl(\frac{dr(t)}{dt}\biggr)^2 dt\biggr], \tag{2.13}$$

where β is a hyperparameter representing the "flatness"; the estimated rate becomes insensitive/sensitive to individual spike occurrences as β is large/small. The flatness may be replaced by the smoothness by changing the first-order derivative to the second-order derivative. The result of the inference is known to be robust to the order of differentiation in the prior distribution (Nemenman and Bialek 2002). The probability of having spikes at $\{t_i\} \equiv \{t_1, t_2, \ldots, t_{N_s}\}$ is given by the "marginal likelihood function" or the "evidence,"

$$p_\beta\bigl(\{t_i\}\bigr) = \int D\bigl\{r(t)\bigr\}\, p\bigl(\{t_i\} \mid r(t)\bigr)\, p_\beta\bigl(r(t)\bigr), \tag{2.14}$$

where $\int D\{r(t)\}$ represents a functional integration over all possible paths of the unknown underlying rate $r(t)$. The method of selecting the hyperparameter according to the principle of maximizing the marginal likelihood function is called the Empirical Bayes method (Good 1965; Akaike 1980; MacKay 1992; Carlin and Louis 2000). The marginalization path integral (2.14) for a given set of spike data $\{t_i\}$ can be carried out by the Expectation Maximization (EM) method (Dempster et al. 1977; Smith and Brown 2003) or the Laplace approximation (Koyama and Paninski 2009). With the hyperparameter determined as $\beta = \beta^*$, we can further obtain the maximum a posteriori (MAP) estimate of the rate $\hat{r}(t)$, so that their posterior distribution,

$$p_{\beta^*}\bigl(r(t) \mid \{t_i\}\bigr) \propto p\bigl(\{t_i\} \mid r(t)\bigr)\, p_{\beta^*}\bigl(r(t)\bigr), \tag{2.15}$$

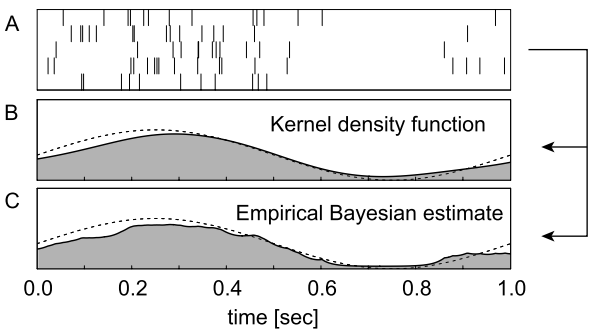

Fig. 2.7 Comparison of the optimized kernel density estimator and Empirical Bayes rate estimator. **A**: Spike data given in Fig. 2.1. **B**: The kernel density estimation of the optimal binsize $\Delta^* = 0.1$. **C**: The MAP rate estimate of the Empirical Bayes method

is maximized. The MAP estimate resembles the kernel filtering method in that the estimated rate $\hat{r}(t)$ is lifted at every spike. The Bayesian MAP rate estimate gives a similar result to the optimized kernel density estimator for a given set of data (Fig. 2.7). With the flatness constant β large/small, the rate estimation becomes insensitive/sensitive to individual spike occurrences. Here we outline the algorithms for maximizing the marginalized likelihood (2.14) by using the EM method, and Kalman filtering and smoothing or "point-process filtering" (Brown et al. 1998) for a given set of spike trains.

An Empirical Bayes method for estimating the rate

1. Perform the marginalization path integral (2.14) using a state-space model or the hidden Markov model, in which the rates and spikes are treated as states and observations, respectively defined at every time a spike occurred.
2. Optimize the hyperparameter by the EM method.
3. Obtain the expected values and the expected variances of the latent variables with Kalman filtering and smoothing.

2.3.2.2 TIPS

An application program that helps the readers to analyze their own neuronal data with the Empirical Bayesian rate estimation was provided by Takeaki Shimokawa at

http://www.ton.scphys.kyoto-u.ac.jp/~shino/toolbox/ssBayes/bayes.html or http://www.apst.spiketrain-analysis.org/.

By simply copying and pasting a data set of spike times, one may obtain the MAP estimate of the rate and a list of $\{t, \hat{r}(t)\}$, with which one can draw the Bayesian

estimation of the rate. The extension to the simultaneous estimation of the rate and the instantaneous irregularity is given at

http://www.ton.scphys.kyoto-u.ac.jp/~shino/toolbox/ssNeCo09/page_SULAB2.html or http://www.apst.spiketrain-analysis.org/

which are based on the theory developed in Shimokawa and Shinomoto (2009).

2.4 Discussion

In this chapter, we introduced typical rate estimation methods, the PSTH, kernel density estimation, and Bayesian estimation. We also demonstrated optimization of the PSTH and the kernel method under the principle of minimizing the MISE and optimizing the Bayesian rate estimation under the principle of maximizing the likelihood function.

Those who intended to practically perform rate estimation may waver regarding the choice of the multiple methods of rate estimation and multiple principles for parameter optimization to use. There is in fact no oracle that may select one of the different optimization principles, and therefore one may choose any of them. Nevertheless, one may expect that reasonable optimization principles provide similar rate estimations for a given set of spike data. It is interesting in this respect to examine whether the different methods may give qualitatively different conclusions for the detectability of rate modulation. For a small number of spike trains derived from an underlying rate $r(t)$ that fluctuates moderately in time, the optimal binsize may diverge, indicating the incapability of detecting a time-dependent rate. For a doubly stochastic process in which the rate is modulated according to the Ornstein–Uhlenbeck process, we may analytically obtain the conditions for the divergence of the optimized binsize Δ^* (Koyama and Shinomoto 2004) and the optimized flatness constant β^* (Koyama et al. 2007). It is noteworthy that both parameters Δ^* and β^* diverge for the identical condition, $n\sigma^2\tau/\mu \leq 1/2$, where the μ, σ^2, and τ are the mean, variance, and timescale of rate fluctuation, respectively. This implies that the different principles such as MISE minimization for a PSTH and marginal likelihood maximization for the Bayes estimation may render similar detectability for rate modulation.

Though these optimization principles may provide reasonable rate estimations, there may still be room for improvement. The practical problem we did not consider in this chapter is the nonstationarity of the rate, such that neuronal activity exhibits abrupt changes in response to a stimulus given to an animal. In such a case, it would be more efficient to modulate the hyperparameter according to the time-varying circumstances. Methods have been proposed for locally adaptive determination of hyperparameters (Abramson 1982; Sain and Scott 1996, 2002; Loader 1999a, 1999b; DiMatteo et al. 2001; Kass et al. 2003; Endres et al. 2008; Shimazaki and Shinomoto 2010).

When applying any sophisticated optimization method to neurophysiological data, one might feel the optimized binsize or bandwidth to be larger than expected.

In practice, the PSTH binsize chosen in neurophysiological literature has tended to be much smaller than the optimal one. This would be because we are inclined to see the details rather than to ignore unnecessary parts. Therefore, the choice lies with the researcher to not use any of the optimization principles. However, there are circumstances in which objective optimization principles are essential. The most important application of the optimal rate estimation method would be the detectability of a significant rate modulation, such as oscillation in neuronal firing. The presence of rate modulation may be judged from the finiteness of the optimal binsize or bandwidth and the systematic modulation of the estimated rate.

Another important application of rate estimation is the analysis of the intrinsic regularity of neuronal firing: It has been revealed that neuronal firing bears non-Poisson aspects; the spike occurrence depends on the preceding spike. The neuronal firing patterns have been analyzed by paying attention to the variability of ISIs (Kuffler et al. 1957; Gerstein and Mandelbrot 1964; Stein 1965; Baker and Lemon 2000; Barbieri et al. 2001; Oram et al. 1999; Kostal and Lansky 2006). However, ISI analysis is vulnerable to firing rate fluctuation, which tends to broaden the ISI distribution and makes the regular sequence evaluated closer to Poissonian randomness. The analysis of firing patterns includes an essential difficulty in its definition of firing regularity. An apparently random sequence of spikes can be interpreted as being derived either irregularly in time from a constant rate or regularly from a fluctuating rate. To determine which interpretation is more plausible in any given case, we introduced a metric measuring the ISI variability rescaled locally in time (Shinomoto et al. 2003, 2005, 2009) and also considered rescaling the sequence of spikes by the instantaneous firing rate (Reich et al. 1998; Koyama and Shinomoto 2005; Nawrot et al. 2008; Shimokawa and Shinomoto 2009). To carry out the time rescaling correctly, one needs to accurately estimate the instantaneous rate. Inversely, the information on firing regularity may be used in improving the firing rate estimation (Cunningham et al. 2008). In this way, the rate and the regularity are complementary aspects of a single spike train; this interesting issue will be discussed in other chapters in more detail (Chaps. 1, 3).

Acknowledgements I thank Shinsuke Koyama, Hideaki Shimazaki, and Takeaki Shimokawa for working with me to develop theories and algorithms of spike train analysis. In addition, Hideaki Shimazaki and Takeaki Shimokawa have rendered services to the public by providing the codes of user-friendly application programs.

References

Abeles M (1982) Quantification, smoothing, and confidence-limits for single-units histograms. J Neurosci Methods 5:317–325

Abramson I (1982) On bandwidth variation in kernel estimates—a square root law. Ann Statist 10:1217–1223

Adrian ED (1928) The basis of sensation: the action of the sense organs. Christophers, London

Akaike H (1980) Likelihood and Bayes procedure. In: Bernardo JM, DeGroot MH, Lindley DV, Smith AFM (eds) Bayesian statistics. University Press, Valencia, p 143

Baker SN, Lemon RN (2000) Precise spatiotemporal repeating patterns in monkey primary and supplementary motor areas occur at chance levels. J Neurophysiol 84:1770–1780
Barbieri R, Quirk MC, Frank LM, Wilson MA, Brown EN (2001) Construction and analysis of non-Poisson stimulus-response models of neural spiking activity. J Neurosci Methods 105:25–37
Brown EN, Frank LM, Tang D, Quirk MC, Wilson MA (1998) A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. J Neurosci 18:7411–7425
Carlin BP Louis TA (2000) Bayes and empirical bayes methods for data analysis, 2nd edn. Chapman and Hall, New York
Cherif S, Cullen KE, Galiana HL (2008) An improved method for the estimation of firing rate dynamics using an optimal digital filter. J Neurosci Methods 173:165–181
Cox DR, Lewis PAW (1966) The statistical analysis of series of events. Wiley, New York
Cunningham JP, Yu BM, Shenoy KV, Sahani M (2008) Inferring neural firing rates from spike trains using Gaussian processes. Adv Neural Inf Process Syst 20:329–336
Daley D, Vere-Jones D (2003) An introduction to the theory of point processes, vol. 1: Elementary theory and methods, 2nd edn. Springer-Verlag, New York
Dayan P, Abbott L (2001) Theoretical neuroscience: computational and mathematical modeling of neural systems. MIT Press, Cambridge
Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. J Roy Statist Soc Ser B 39:1–38
DiMatteo I, Genovese CR, Kass RE (2001) Bayesian curve-fitting with free-knot splines. Biometrika 88:1055–1071
Endres D, Oram M, Schindelin J, Földiák P (2008) Bayesian binning beats approximate alternatives: estimating peri-stimulus time histograms. Adv Neural Inf Process Syst 20:393–400
Gerstein GL, Kiang, NYS (1960) An approach to the quantitative analysis of electrophysiological data from single neurons. Biophys J 1:15–28
Gerstein GL, Mandelbrot B (1964) Random walk models for the spike activity of a single neuron. Biophys J 4:41–68
Good IJ (1965) The estimation of probabilities: an essay on modern Bayesian methods. MIT Press, Cambridge
Johnson DH (1996) Point process models of single-neuron discharges. J Comput Neurosci 3:275–299
Kass RE, Ventura V, Cai C (2003) Statistical smoothing of neuronal data. Network Comput Neural Syst 14:5–15
Kass RE, Ventura V, Brown EN (2005) Statistical issues in the analysis of neuronal data. J Neurophysiol 94:8–25
Kostal L, Lansky P (2006) Classification of stationary neuronal activity according to its information rate. Network Comput Neural Syst 17:193–210
Koyama S, Shinomoto S (2004) Histogram bin-width selection for time-dependent point processes. J Phys A Math Theor 37:7255–7265
Koyama S, Shinomoto S (2005) Empirical Bayes interpretations of random point events. J Phys A Math Theor 38:L531–L537
Koyama S, Shimokawa T, Shinomoto S (2007) Phase transitions in the estimation of event rate: a path integral analysis. J Phys A Math Theor 40:F383–F390
Koyama S, Paninski L (2009) Efficient computation of the maximum a posteriori path and parameter estimation in integrate-and-fire and more general state-space models. J Comput Neurosci doi:10.1007/s10827-009-0179-x
Kuffler SW, Fitzhugh R, Barlow HB (1957) Maintained activity in the cat's retina in light and darkness. J Gen Physiol 40:683–702
Loader CR (1999a) Bandwidth selection: classical or plug-in? Ann Statist 27:415–438
Loader CR (1999b) Local regression and likelihood. Springer-Verlag, New York
MacKay DJC (1992) Bayesian interpolation. Neural Comput 4:415–447
Nawrot M, Aertsen A, Rotter S (1999) Single-trial estimation of neuronal firing rates: from single-neuron spike trains to population activity. J Neurosci Methods 94:81–92

Nawrot MP, Boucsein C, Rodriguez-Molina V, Riehle A, Aertsen A, Rotter S (2008) Measurement of variability dynamics in cortical spike trains. J Neurosci Methods 169:374–390
Nemenman I, Bialek W (2002) Occam factors and model-independent Bayesian learning of continuous distributions. Phys Rev E 65:026137
Oram MW, Wiener MC, Lestienne R, Richmond BJ (1999) Stochastic nature of precisely timed spike patterns in visual system neuronal responses. J Neurophysiol 81:3021–3033
Parzen E (1962) Estimation of a probability density-function and mode. Ann Math Statist 33:1065
Paulin MG (1992) Digital filters for firing rate estimation. Biol Cybern 66:525–531
Paulin MG, Hoffman LF (2001) Optimal filtering rate estimation. Neural Networks 14:877–881
Reich DS, Victor JD, Knight BW (1998) The power ratio and the interval map: spiking models and extracellular recordings. J Neurosci 18:10090–10104
Richmond BJ, Optican LM, Spitzer H (1990) Temporal encoding of two-dimensional patterns by single units in primate primary visual cortex. I. Stimulus-response relations. J Neurophysiol 64:351–369
Rieke F, Warland D, de Ruyter van Steveninck R, Bialek W (1997) Spikes: exploring the neural code. MIT Press, Cambridge
Rudemo M (1982) Empirical choice of histograms and kernel density estimators. Scand J Statist 9:65–78
Sain S, Scott D (1996) On locally adaptive density estimation. J Amer Statist Assoc 91:1525–1534
Sain S, Scott D (2002) Zero-bias locally adaptive density estimators. Scand J Statist 29:441–460
Shimazaki H, Shinomoto S (2007a) A method for selecting the bin size of a time histogram. Neural Comput 19:1503–1527
Shimazaki H, Shinomoto S (2007b) Kernel width optimization in the spike-rate estimation. Budelli R, Caputi A, and Gomez L (eds) Neural coding 2007, pp 143–146
Shimazaki H, Shinomoto S (2010) Kernel bandwidth optimization in spike rate estimation. J Comput Neurosci, published on line. doi:10.1007/s10827-009-0180-4
Shimokawa T, Shinomoto S (2009) Estimating instantaneous irregularity of neuronal firing. Neural Comput 21:1931–1951
Shinomoto S, Shima K, Tanji J (2003) Differences in spiking patterns among cortical neurons. Neural Comput 15:2823–2842
Shinomoto S, Miyazaki Y, Tamura H, Fujita I (2005) Regional and laminar differences in in vivo firing patterns of primate cortical neurons. J Neurophysiol 94:567–575
Shinomoto S, Kim H, Shimokawa T, Matsuno N, Funahashi S, Shima K, Fujita I, Tamura H, Doi T, Kawano K, Inaba N, Fukushima K, Kurkin S, Kurata K, Taira M, Tsutsui K, Komatsu H, Ogawa T, Koida K, Tanji J, Toyama K (2009) Relating neuronal firing patterns to functional differentiation of cerebral cortex. PLoS Comput Biol 5:e1000433
Smith AC, Brown EN (2003) Estimating a state-space model from point process observations. Neural Comput 15:965–991
Snyder D (1975) Random point processes. Wiley, New York
Stein RB (1965) A theoretical analysis of neuronal variability. Biophys J 5:173–194

Chapter 3
Analysis and Interpretation of Interval and Count Variability in Neural Spike Trains

Martin Paul Nawrot

Abstract Understanding the nature and origin of neural variability at the level of single neurons and neural networks is fundamental to our understanding of how neural systems can reliably process information. This chapter provides a starting point to the empirical analysis and interpretation of the variability of single neuron spike trains. In the first part, we cover a number of practical issues of measuring the inter-spike interval variability with the coefficient of variation (CV) and the trial-by-trial count variability with the Fano factor (FF), including the estimation bias for finite observations, the measurement from rate-modulated spike trains, and the time-resolved analysis of variability dynamics. In the second part, we specifically explore the effect of serial interval correlation in nonrenewal spike trains and the impact of slow fluctuations of neural activity on the relation of interval and count variability in stochastic models and in in vivo recordings from cortical neurons. Finally, we discuss how we can interpret the empirical results with respect to potential neuron-intrinsic and neuron-extrinsic sources of single neuron output variability.

3.1 Introduction

In the living animal, neural signals fluctuate on various temporal and spatial scales. Across experimental repetitions, neural responses may vary considerably in microscopic and macroscopic signals, both in invertebrate and vertebrate brains. Understanding how nervous systems ensure reliable function under the variable and seemingly noisy in vivo conditions is a key issue in computational systems neuroscience that is of fundamental importance for theories on sensory coding, learning and memory, and behavioral control.

M.P. Nawrot (✉)
Neuroinformatics and Theoretical Neuroscience, Institute of Biology, Freie Universität Berlin, Königin Luise Straße 1-3, 14195 Berlin, Germany
e-mail: martin.nawrot@fu-berlin.de
url: http://www.biologie.fu-berlin.de/neuroinformatik

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_3, © Springer Science+Business Media, LLC 2010

In this chapter, we introduce methods to analyze two aspects of neural output variability. The variance of inter-spike intervals reflects intra-trial variability on a relatively fast time scale of tens to hundreds of milliseconds. In contrast, the variance of the number of spikes counted during repeated experimental observations reflects a variability on a comparably slow time scale of seconds or even minutes. On theoretical grounds, interval and count statistics are fundamentally related. We will thus place a special focus on the coanalysis of both aspects, and we suggest ways to interpret their empirical relation in the light of stochastic models. The present chapter emphasizes practical issues that are relevant for the analysis of experimental data. The Appendix provides reference to a number of Matlab tools for point process simulation and spike train analysis which are publicly available with the FIND toolbox (Meier et al. 2008). Additional course material including example data sets is made available at the portal site of the German Neuroinformatics Node (http://www.g-node.org or http://www.apst.spiketrain-analysis.org/).

3.2 The Analysis of Inter-Spike Interval Variability

3.2.1 The Coefficient of Variation and Bias of Estimation

Definition 1 We consider the empiric observation of a series of spike events within a finite interval $(a, b]$ with $a < b$ and duration $T = b - a$. We assume a finite number of spike events N within $(a, b]$. We denote the spike times as $a < t_1 < t_2 < \cdots < t_N \leq b$ and define the $N - 1$ inter-spike intervals as $X_1, X_2, \ldots, X_{N-1}$, where $X_i = t_{i+1} - t_i$. Repeated and independent observations j result in an ensemble of k independent spike trains, each with a spike count N^j.

Practically, we obtain repeated independent measurements of action potentials either during repeated experimental trials, the time-frame of which is defined by the experimental protocol (e.g., in fixed temporal relation to a sensory stimulus presentation). Repeated observations may also be obtained through segmentation of a continuous spike train (e.g., recorded during sleep or under spontaneous conditions) into subsequent, nonoverlapping observation windows of equal length. In this section we assume the repeated observation of a spiking process that has a constant spike rate, and we assume that the constant firing rate is identical in each trial.

The empirical distribution of inter-spike intervals, its mean, variance, and higher moments generally depend on the length T of the observation window. Suppose that we empirically sample intervals X that were drawn from a fix interval distribution $f(x)$ within a finite observation window of length T as in Fig. 3.1A, where the observation window is expressed in multiples of the mean inter-spike interval (we will call this the operational time axis). Evidently, we can only observe intervals X that are shorter than the observation window T, and thus the empirical interval distribution is $\hat{f}(x) = 0$ for $x > T$ (cf. Fig. 3.1B). For all intervals $x \in (0, T]$, the likelihood of their observation is proportional to $T - x$, which leads to the following expression for the empiric distribution

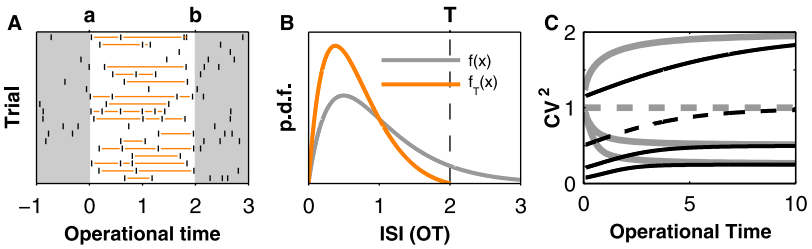

Fig. 3.1 Bias of CV estimator for a finite observation window. (**A**) Independent empiric observations (trials) of a gamma renewal process within the finite observation window $(0, 2]$ in operational time results in an empiric sample of inter-spike intervals X (*orange*). Intervals $X > T'$ cannot be observed; long intervals are more likely to span across one or both interval limits than short ones. (**B**) Gamma distribution $f(x)$ of order $\alpha = 2$ (*gray*) and distribution $\hat{f}(x)$ restricted to intervals $X \le T' = 2$ (*orange*, (3.1), normalized to unit area). Mean and variance are clearly smaller for $\hat{f}(x)$ than for $f(x)$. (**C**) Dependence of the CV on the observation window. Shown is the expectation value for the empiric squared coefficient of variation CV^2 (*black*) and the Fano factor (*gray*; cf. 3.3.1) in dependence on the interval T' in operational time for gamma processes of different order $\alpha = 0.5, 1, 2, 4$ (from *top* to *bottom*). *Dashed lines* correspond to the Poisson process. For increasing T' the empiric CV^2 and the empiric FF approach $\mathrm{CV}^2_\infty = 1/\alpha$

$$\hat{f}(x) = \begin{cases} \eta^{-1}(T - x)\, f(x) & \text{for } x \in [0, T], \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

where

$$\eta = \int_0^T (T - s)\, f(s)\, ds$$

normalizes the distribution to unit area. Thus, long intervals ($X \lesssim T$) are less frequently observed than short ones ($X \ll T$), a statistical effect also known as right censoring (Wiener 2003). This becomes intuitively clear when we consider that long intervals are likely to span across the left or right limit of our observation window such that, e.g., $t_i < a < t_{i+1}$. On the contrary, multiple small intervals may fit into one single observation (cf. Fig. 3.1A).

Definition 2 We define the empiric coefficient of variation for a set of inter-spike intervals as the standard deviation of interval lengths divided by the mean interval length

$$\mathrm{CV} = \frac{\mathrm{SD}[X]}{\mathrm{E}[X]}. \tag{3.2}$$

In the case of repeated independent observations (trials) j we have two options for computing the CV. The standard procedure is to compute the CV across the complete set of intervals pooled from all observations. Alternatively, we may first compute the individual CV^j for each trial separately and in a second step calculate the mean $\overline{\mathrm{CV}} = \frac{1}{k} \sum \mathrm{CV}^j$ across trials. Under stationary conditions where the generating stochastic process has a constant rate which is identical in all trials it follows that $\mathrm{CV} = \overline{\mathrm{CV}}$, on expectation.

Right censoring introduces a systematic error to the empirical estimation of the coefficient of variation (Nawrot et al. 2008). For a unimodal interval distribution the empirical CV underestimates the theoretical value CV_∞ that is derived from the full distribution. To explore this effect in more detail we calculated the empirical $\mathrm{CV}(T')$ for the widely used model of the gamma interval distribution (see Appendix) as a function of the observation time. In Fig. 3.1C we explore this dependence for the *squared* coefficient of variation because it directly relates to the Fano factor (see Subsect. 3.3.1). We find that the empiric CV^2 drops with decreasing observation time. Conversely, with increasing observation time, the empiric CV^2 approaches the theoretical CV^2_∞. The dashed line refers to the special case of the Poisson process with $\alpha = 1$. Note, that we expressed observation time $T' = T/\mathrm{E}[X]$ in multiples of the mean inter-spike interval $\mathrm{E}[X]$ (operational time), which gives results that are independent of the actual firing rate. In practice, we may produce calibration curves similar to those in Fig. 3.1C from experimental data to explore this bias behavior in a given set of data. Elsewhere, we estimated that for regular spiking cortical neurons, observation intervals that comprise about 5–10 ISIs are practically of sufficient length to avoid a strong bias (Nawrot et al. 2008).

Due to the *finite length* T of the observation window, one cannot sample the full interval distribution $f(x)$ that is generally defined on $\mathbb{R}_+$. This introduces a *bias of estimation for the empiric* CV which generally leads to the *under*estimation of the theoretic CV_∞ (Fig. 3.1). Practical consequences: 1. Use long observation windows, i.e., clearly longer than the average ISI ($T \gg \mathrm{E}[X]$). 2. If short observation windows are necessary, e.g., to uncover fast variability dynamics (see Fig. 3.2D), use a fixed window size in operational time to ensure a *constant* bias across repeated measurements.

3.2.2 Analysis of Rate-Modulated Spike Trains

The CV measures the dispersion of the interval distribution. It characterizes the irregularity of spike trains and allows one to quantify the stochastic nature of the observed spiking process. However, the CV is a useful measure only if the spike rate is constant over time and if the variation of intervals is of stochastic nature such as in the case of the gamma renewal process illustrated in Fig. 3.1. Whenever a neuron modulates its output firing rate, e.g., in response to a sensory stimulus, then this rate modulation strongly influences the empiric interval distribution. Any rate modulations that are slow compared to the mean ISI will increase the dispersion of the empiric interval distribution and thus lead to an increased CV which no longer reflects the stochastic nature of the spiking process alone (see Fig. 3.2A).

Here, we describe one possible strategy to overcome this problem which requires two additional steps of analysis as demonstrated in Fig. 3.2. First, one obtains an estimate $\hat{\lambda}(t)$ of the time-varying firing rate on the basis of repeated measurements.

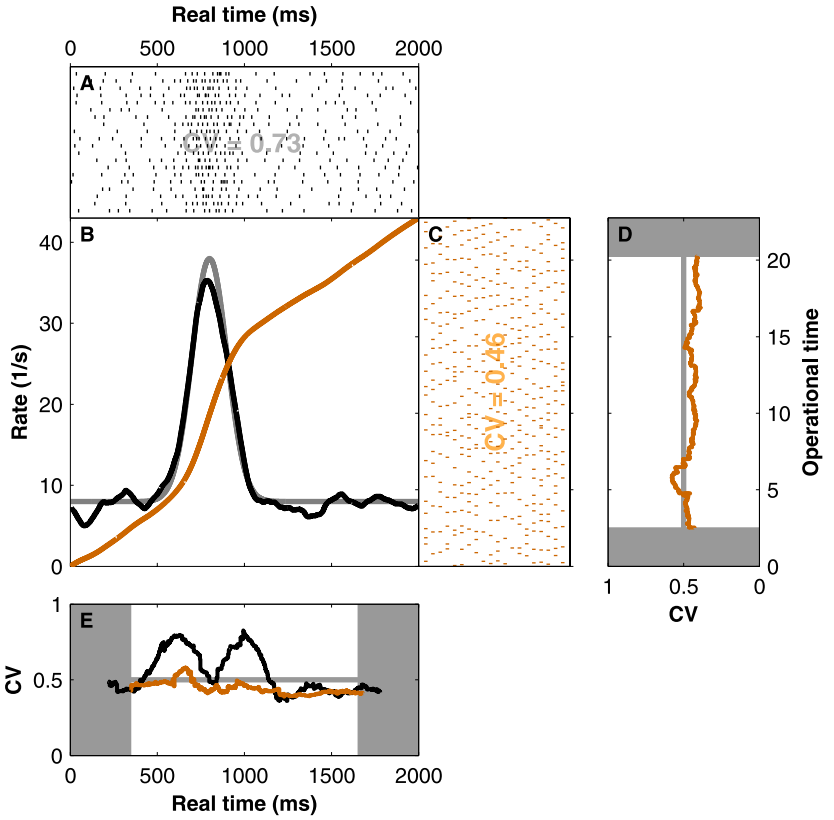

Fig. 3.2 Estimation of interval variability in operational time. (**A**) Repeated observations of a rate-modulated gamma process with order $\alpha = 4$. We expect for the gamma renewal process $\mathsf{CV}_\infty = 1/\sqrt{\alpha} = 0.5$. The empiric estimate $\mathsf{CV} = 0.73$ is artificially increased due to the changing rate. (**B**) Kernel estimate of the time-varying firing rate $\hat{\lambda}(t)$ (*black*) from $N = 20$ spike trains in **A** (triangle kernel, $\sigma_k = 45$ ms) and integrated rate function $\Lambda(t)$ (*maroon*). The *gray function* depicts the original intensity used to simulate the spike trains in **A**. (**C**) Demodulated spike trains in operational time. Each single spike time in **A** was transformed according to $t_i' = \Lambda(t_i)$. In operational time the empiric estimate $\mathsf{CV} = 0.46$ agrees well with the expectation $\mathsf{CV}_\infty = 0.5$. (**D**) Time-resolved estimate of the CV in operational time. Window width is $T' = 5$. (**E**) Time-resolved CV as in **D** back-transformed to experimental time (*maroon*). The time-resolved CV estimated from the original spike trains in **A** (*black*) is modulated due to the changes in firing rate

Second, one transforms the experimental time axis to the so-called operational time axis such that the firing rate modulation is compensated (time warping). In operational time we then proceed with estimating the CV.

3.2.2.1 Step 1. Estimation of the Rate Function

Obtaining a reasonably good estimate of the rate function is crucial. Here, we use the method of linear kernel convolution (Nawrot et al. 1999; Shimazaki and Shinomoto

2010; Parzen 1962) with a fixed kernel function. After choosing a kernel shape which has little influence on the quality of the estimate, one has to fix the kernel width which determines the time resolution of the rate estimate. In the example of Fig. 3.2, we first pooled the spike trains from all observations (trials) and then estimated the trial-averaged rate function. To this end, we chose a symmetric kernel of triangular shape. To obtain an estimate for the optimal kernel width σ_k (defined as the standard deviation of the normalized kernel function) on the basis of the empirical data, we applied a heuristic method outlined elsewhere (Nawrot et al. 1999). Recently, Shimazaki and Shinomoto (2010) formalized this optimization of the kernel width for fixed and variable width kernels using a Bayesian approach on the basis of a specific model assumption for the generating point process. For fixed width kernels, this approach is outlined in detail in Chap. 2 of this book.

3.2.2.2 Step 2. Demodulation and Analysis in Operational Time

Based on the estimated firing rate $\lambda(t)$, we define the time transformation (Reich et al. 1998; Brown et al. 2002; Nawrot et al. 2008)

$$t' = \Lambda(t) = \int_0^t \lambda(s)\, ds, \tag{3.3}$$

according to the integrated rate function for all spike events t_i^j. We call t' the operational time because on this new time axis the empirical spiking process has constant unit rate. Figure 3.2B shows the integral $\Lambda(t)$ (maroon) of the empiric rate function $\lambda(t)$ (black). The transformed spike trains depicted in Fig. 3.2C do not display any overt rate modulation and result in an empiric estimate $\mathsf{CV} = 0.46$, which is close to the theoretic $\mathsf{CV}_\infty = 0.5$ of the underlying gamma process that was used for simulation.

> Transformation of spike times from the experimental time axis to the *operational time axis* according to the integrated rate function can eliminate rate fluctuations in the spike train. In a next step, this allows us to obtain an empiric estimate of the CV in operational time. This method requires a reliable estimate of the time-varying rate function (Fig. 3.2).

3.2.2.3 Time-Resolved Analysis of the CV

It is now straightforward to analyze the $\mathsf{CV}(t')$ as a function of operational time using a sliding window approach. The window width T' defines the time resolution of this analysis, and we are faced with a trade-off between short windows that ensure a good time resolution of our analysis and large windows that reduce the variance and the bias of estimation (see Subsect. 3.2.1). In Fig. 3.2D, we estimated $\mathsf{CV}(t')$ within a window of length $T' = 5$, i.e., the window size is 5 times the average

interval. We find no significant variation with a mean of $\langle \mathsf{CV}(t') \rangle = 0.45$, a faithful representation of the underlying gamma process used for simulation in Fig. 3.2A. In a final step we may use the inverse time transformation of (3.2) (Meier et al. 2008; Nawrot et al. 2008) to represent our time-resolved estimate $\mathsf{CV}(t')$ in experimental time $\mathsf{CV}(t)$ (see Fig. 3.2E). Note that the support points at which the measured values $\mathsf{CV}(t)$ are represented are not equidistant in experimental time.

3.2.2.4 Alternative Methods

There are several alternative parametric and nonparametric methods to estimate interval variability from rate-modulated spiking activity and in a time-resolved manner. A number of nonparametric so-called *local measures* have been proposed that estimate normalized interval variability locally in time. The common idea behind these approaches is that a temporally confined estimate will largely ignore rate modulations that are comparatively slow. At each step in time, local measures are based on rather small data samples and are thus inherently noisy—i.e., they express a large variance of estimation—and they are in general subject to estimation biases. Estimation variance may be decreased by temporal averaging over local estimates. Here, I briefly outline two local measures. A simple yet efficient method for estimating the local CV from repeated trials has been introduced by Benda (2002). At any given point in time t, this method computes the empiric CV from all intervals in all trials that contain t, i.e., for which $t_i < t < t_{i+1}$. Evidently, shorter intervals are less likely to be observed than longer ones. This introduces an estimation bias with respect to the CV_∞ which is opposed to the one we described in Subsect. 3.2.1, and which can be compensated (Nawrot and Benda 2006). Rate fluctuations on a time scale that are longer than the average ISI will have little influence on this measure. It is, however, sensitive to across-trial nonstationarities of the rate. The "classical" local measure termed C_{V2} was introduced in 1996 by Holt et al. (1996). It simply computes the coefficient of variation for each successive pair of intervals (X_i, X_{i+1}), i.e., it normalizes the variance across two successive intervals by their mean and thus becomes insensitive to across-trial nonstationarities and largely ignores rate modulations that are slower than twice the average ISI. Other local measures are mostly variants thereof, and each has been designed under a certain optimization constraint. The robustness of these measures is typically increased by averaging across trials and across time. An in-depth review and calibration of the C_{V2} and three other local measures (Shinomoto et al. 2005; Davies et al. 2006; Miura et al. 2006) was recently published by Ponce-Alvarez et al. (2009).

In competition to nonparametric local measures, a few parametric methods of estimating the firing irregularity have been proposed. They assume a specific underlying model (e.g., a nonhomogeneous Poisson process) and estimate a single or several model parameters from the empiric spike train. Recently, Shimokawa and Shinomoto (2009) introduced an elegant method for which they assume a gamma process with time-varying intensity (firing rate) $\lambda(t)$ and time-varying regularity (order of the gamma process) $\alpha(t)$. Using a Bayesian approach, the proposed method allows us to estimate both $\lambda(t)$ and $\alpha(t)$ from a given set of empirical data.

3.3 The Combined Analysis of Interval and Count Variability

In the case of a mathematically defined point process model, its interval and count statistics are uniquely determined and inherently related (see Appendix). To characterize an unknown neural spiking process on the basis of experimental observations, it is therefore useful to coanalyze interval *and* count statistics, and their specific relation. This can help to choose a particular stochastic model (or a class of models) that adequately describes the experimentally observed process.

3.3.1 Fano Factor and Bias of Estimation

The Fano factor is a well-established measure of count variability and has been repeatedly used to quantify spike train variability (for review, see Nawrot et al. 2008).

Definition 3 The empiric Fano factor FF is defined as the ratio of the variance and the mean of the spike count N^j as measured within an observation interval of length T during repeated observations j:

$$\mathsf{FF} = \frac{Var[N^j]}{\mathsf{E}[N^j]}. \tag{3.4}$$

The distribution of spike count across repeated observations and thus the mean and variance of this distribution generally depend on the length T of the observation window. This introduces an estimation bias for the empiric FF with respect to the limit value $\mathsf{FF}_\infty = \lim_{T\to\infty} \mathsf{FF}$ which can be derived analytically from the definition of the process. In Fig. 3.1C we demonstrate how the Fano factor depends on the observation window $T' = T/\mathsf{E}[X]$ for gamma processes of different order α. With increasing observation time T', the FF estimates approach the limit values FF_∞. As for the CV, an observation window of length $T' = 5$–10 seems practically sufficient to avoid a strong bias if the observed process is more regular than the Poisson process ($\alpha \geq 1$), e.g., in regular spiking cortical neurons (Nawrot et al. 2008). For decreasing observation times $T' \to 0$, the Fano factor approaches unity. This can be easily understood as an approximation of the Bernoulli process for which in each small interval we observe either 1 spike with probability p or 0 spikes with probability $1 - p$. As $T' \to 0$, the variance $p(1 - p)$ of the Bernoulli distribution approaches the mean p (Teich et al. 1997).

The *finite length* T of the observation window introduces a *bias of estimation for the empiric FF* (Fig. 3.1C). As $T' \to 0$, the Fano factor approaches unity. Practical consequences as in Subsect. 3.2.1: 1. Use long observation windows, and 2. If short observation intervals are necessary, use a fix window size in operational time to ensure a *constant* bias.

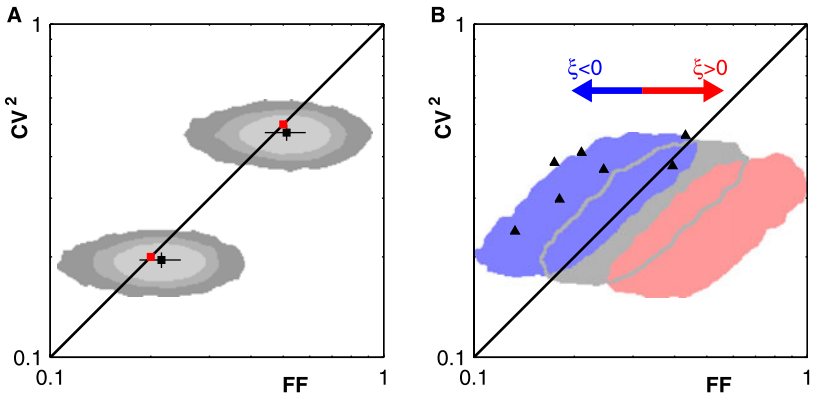

Fig. 3.3 Count variability (FF) versus interval variability (CV^2). (**A**) Variance of estimator and residual bias effect. The renewal prediction $FF_\infty = CV^2_\infty$ is depicted by the *black diagonal*. *Grey shadings* represent 90% confidence regions for numeric ensemble simulation of gamma processes of order $\alpha = 2$ ($CV^2_\infty = 0.5$) and $\alpha = 5$ ($CV^2_\infty = 0.2$). *Grey colors* indicate number of trials ($N = 20, 50, 100$, from *dark* to *light gray*). The observation time comprised $T' = 10$ intervals on expectation. Each confidence region was computed from 10,000 ensembles as follows. A 2D Gaussian filter produced a smooth 2D histogram of all log-transformed value pairs. After sorting all histogram entries starting with the largest entry, their cumulative sum was computed. All indices up to the index for which the 90% quantile was reached define the 2D confidence region. The *black squares* and *lines* depict average and standard deviation for $n = 100$ trials. The *red squares* indicate expectation values. (**B**) The effect of serial interval correlation on interval and count variability. *Colored shadings* represent the 90% confidence regions from 10,000 numeric ensemble simulations (50 trials, $T' = 10$) of the autoregressive process with marginal log-normal interval distribution (see text). The *blue* (*red*) *region* shows the effect of a negative (positive) serial interval correlation with respective parameter $\beta = -0.3$ ($\beta = 0.3$); *gray region* is computed from the interval-shuffled spike trains which do not exhibit serial correlations. *Black triangles* reproduce the empiric results obtained from 7 cortical cells (Nawrot et al. 2007). Data courtesy of Clemens Boucsein, University of Freiburg, Germany

3.3.2 *Fano Factor vs. Squared Coefficient of Variation*

For the class of renewal point processes, the expectation values of FF and CV^2 are simply related by

$$FF_\infty = CV^2_\infty. \tag{3.5}$$

Renewal processes are widely used as models for neural spiking. As a starting point for our analysis of experimental data, we may therefore formulate the renewal prediction (3.5) as the null-hypothesis. Any deviation from this null-hypothesis may then trigger further analysis.

A direct way of jointly visualizing the empiric relation of interval and count variability is to plot FF against CV^2 in a scatter diagram as demonstrated in Figs. 3.3 and 3.4. Individual empirical estimates of FF and CV^2 are computed from a finite number of samples and are, therefore, subject to statistical errors that are expressed in the variance of estimation (Nawrot et al. 2008). Repeated measurements will

thus lead to values that scatter around the theoretic expectation value. Figure 3.3A demonstrates the effect of a limited sample size in numeric simulations of the gamma renewal process with order parameters $\alpha = 2$ and $\alpha = 5$ and corresponding expectation values $\mathsf{CV}^2_\infty = \mathsf{FF}_\infty = 1/\alpha$. We chose different numbers of trials $n = 20, 50, 100$ and constructed the 90% confidence region from 10,000 independent simulations, depicted as gray shadings. The empirical sample size of intervals and counts scales linearly with the number of trials. Consequently, reducing the number of trials increases the variance of estimation for both, FF (horizontal) and CV^2 (vertical). The number of intervals additionally scales with T' and, thus, reducing observation time will increase the variance of the CV^2 estimator (not shown; Nawrot et al. 2008).

In practice, residual estimation biases due to the experimentally limited observation time T for CV^2 (see Subsect. 3.2.1) and FF (see Subsect. 3.3.1) may affect their empirical relation. As a consequence, in Fig. 3.3A the average empiric values for $T' = 10$ (black squares) of the Fano factor is larger, and the average empiric value of the CV^2 is smaller than the expectation values indicated by red squares.

For any (stationary) renewal point process, the relation of Fano factor and coefficient of variation is given by $\mathsf{FF}_\infty = \mathsf{CV}^2_\infty$. For the special case of the Poisson process, it holds that $\mathsf{FF} = \mathsf{CV}^2_\infty = 1$.

3.3.3 The Effect of Serial Interval Correlation

Renewal processes represent the most prominent class of stochastic models for neural spiking. Yet, serial correlations of inter-spike intervals have been observed experimentally in various systems including neocortical cells (for review, see Farkhooi et al. 2009). For stationary point processes in equilibrium with serially correlated inter-event intervals, the following equality holds (McFadden 1962; Cox and Lewis 1966):

$$\lim_{T\to\infty} \mathsf{FF} = \mathsf{CV}^2_\infty (1 + 2\,\xi) \quad \text{with } \xi = \sum_{i=1}^{\infty} \xi_i, \tag{3.6}$$

where ξ_i denotes the ith-order linear correlation coefficient, i.e., the expected linear correlation for all pairs of intervals $(\mathsf{ISI}_k, \mathsf{ISI}_{k+i})$ that are separated by $i-1$ intermediate intervals. If all correlation coefficients vanish, we obtain the renewal statistics where $\mathsf{FF}_\infty = \mathsf{CV}^2_\infty$. Overall negative serial correlation $\xi < 0$ will result in a Fano factor that is smaller than the CV^2, while a positive correlation $\xi > 0$ leads to an increased count variability.

We demonstrate this effect in numerical simulations of a simple autoregressive model as outlined in (Farkhooi et al. 2009) (see Appendix). The intervals X of this model are log-normal distributed. The serial correlation of intervals is controlled

by an additional correlation parameter β. Correlations are short ranged, i.e., the linear correlation coefficients ξ_i quickly diminish with increasing serial correlation order i (Farkhooi et al. 2009). In Fig. 3.3B, we consider two cases: (i) *negative* serial correlation of ISIs ($\xi < 0$, blue), and (ii) *positive* correlation ($\xi > 0$, red). Both are compared to the corresponding renewal process ($\xi = 0$, gray). In each case we simulated 10,000 spike train ensembles of 50 trials, and each ensemble represents the repeated measurement of one neuron with a single estimate for CV^2 and FF. For each neuron, we adjusted the model parameters to obtain a specific value of the squared coefficient of variation in the range $\mathsf{CV}^2_\infty \in [0.2, 0.5]$. This covers the empirically relevant range for regular spiking neocortical neurons under stationary conditions (e.g., Nawrot et al. 2007; Nawrot et al. 2008). From all 10,000 simulated samples we numerically constructed the confidence region which covers 90% of the measurements. As theoretically predicted, the negative serial correlations reduce the Fano factor, in this case by about 30%, while positive correlations increase the Fano factor by about 60%.

To compare the modeling results with experimental findings, we reanalyzed intracellular recordings from rat somatosensory cortex of the anesthetized rat (Nawrot et al. 2007). 7 of 8 regular spiking cortical cells expressed short-ranged negative interval correlations with $\xi \approx -0.2$ leading to a count variability reduced by 30% (black triangles in Fig. 3.3B).

Negative serial interval correlations ($\xi < 0$) in a stationary point process realization lead to a *reduced* count variance as compared to the count variance of a renewal process with the same interval distribution, and thus $\mathsf{FF} < \mathsf{CV}^2$. *Positive* serial interval correlations ($\xi > 0$) lead to an *increased* count variance, and thus $\mathsf{FF} > \mathsf{CV}^2$; see (3.6) and Fig. 3.3B.

3.3.4 The Effect of Nonstationarity

In the typical experimental situation, we make repeated observations in time (trial design). This allows us to perform statistical analyses on the trial ensemble, e.g., estimating the trial-averaged firing rate or the variance of the spike count across trials. By doing so, we make the implicit assumption that the observed spiking process is stationary in time and across trials (Knoblauch and Palm 2005; Nawrot et al. 2008). However, this assumption is often violated in neural systems. In this section we explore the influence of a particular type of nonstationarity: we assume slow modulations of the firing rate on time scales of seconds or even minutes. In the living animal, such modulations are likely to occur for various reasons (see Sect. 3.4).

3.3.4.1 Slow Activity Fluctuations Introduce Across-Trial Nonstationarity

To model slow fluctuations of the firing rate, we use the following approach. In a first step we generate a time-dependent intensity $\phi(t)$ using a moving average process with log-normal distributed noise (see Appendix). The intensity (or rate) fluctuates about the mean of $r' = 1$ on a slow time scale $\tau' = 20$ in operational time (e.g., equivalent to a rate of 10 Hz and $\tau = 2$ s in experimental time). In the next step we generate a realization of a rate-modulated gamma process with intensity $\phi(t)$ and order parameter $\alpha = 2$ and with a total duration of 500 expected spikes in operational time. In a final step we divide this spike train into $n = 50$ observations (or trials) of length $T' = 10$ and analyze interval and count variability. Again, we compute confidence regions for FF vs. CV^2 in the scatter diagram of Fig. 3.4A.

The Fano factor is boosted by the additional nonstationarity (green shadings) and can reach very high values that are up to 20 times larger than in the stationary case (gray shading). This effect can be easily understood. The expectation value for the spike count varies from trial to trial as the process intensity modulates on long time scales and thus across trials. This has a dramatic effect on the distribution and variance of the spike count. The CV^2 is only slightly increased (light green shading), and the effect dominates in ensembles that also show high count variability. The general explanation for the increased CV^2 is simple: shorter intervals in trials with higher intensity and longer intervals in trials of lower intensity will lead to an additional dispersion of the interval distribution. This effect can be avoided. In Subsect. 3.2.1 we introduced an alternative way of estimating the $\overline{\mathrm{CV}}^2$ by estimating the CV_i in each individual trial and subsequent averaging. This procedure normalizes per trial and thus is not susceptible to across-trial nonstationarities. In Fig. 3.4A the dark green shading indicates the corresponding confidence region. In summary, the FF is strongly increased, while the distribution of the $\overline{\mathrm{CV}}^2$ with mean 0.37 is similar to that of the control with mean 0.38.

We compare the simulation results to a set of in vivo single unit recordings from the primary motor cortex of a monkey (*Macaca mulatta*) that performed a delayed center-out reaching task (Rickert et al. 2009). We analyzed interval and count variability during the first 900 ms of the 1-s delay period. At the start of this period, the monkey had received a directional cue but was not allowed to move his arm until the GO signal appeared at the end of the delay period. Many neurons showed a task-related activation profile that was tuned for the specific target cue. We therefore estimated the trial-averaged firing rate and performed the analysis in operational time (see Subsect. 3.2.2). The results are shown in Fig. 3.4B. The Fano factor assumes high values with a mean of $\mathrm{FF} = 1.87$ (median 1.39), while the values of the CV^2 are considerably lower with average $\mathrm{CV}^2 = 0.76$ (median 0.70). The shape of the 90% confidence region compares to that of the numeric simulations in Fig. 3.4A. Two additional factors will likely lead to an overestimation of the empiric CV^2 in the present data. First, we may assume that the activity is not stationary across trials due to slow modulations, as in our model simulations. As a consequence, the resulting estimate of the task related rate profile from the trial-averaged spike trains does not properly reflect the single-trial rate profile. Second, we deal with another type

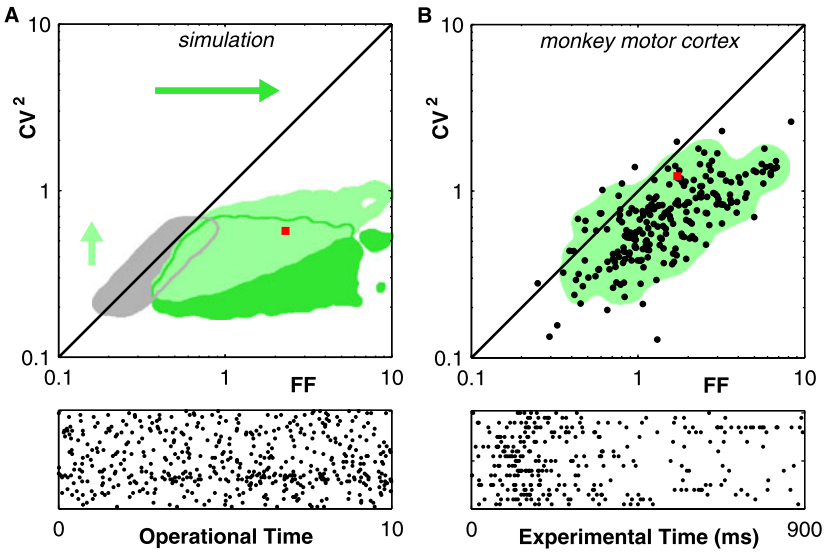

Fig. 3.4 Slow rate modulations can boost count variance. (**A**) Simulation. The *gray shading* represents the confidence interval for gamma processes with $\alpha \in [2, 5]$. The *green shadings* demonstrate the effect of slow modulations of the process intensity (MA log-normal noise; $\sigma = 200$, $\tau' = 20$). The FF is strongly increased. The empiric CV^2 (*light green*) was estimated by pooling intervals from all trials. The $\overline{\mathsf{CV}}^2$ (*dark green*) was estimated in each trial separately and then averaged. *Bottom panel* shows spike raster for one ensemble (*red square*). (**B**) Experiment. In vivo estimates from 56 motor cortical single units, each recorded in 6 directions (see text). The FF strongly exceeds the CV^2. The CV^2 was estimated from intervals pooled across trials. For each ensemble the number of trials was ≥ 15 (to limit the variance) and the observation window comprised $T' \geq 10$ intervals (to avoid a strong bias). This included a total of 223 samples. *Bottom panel* shows one example (*red square*). Modified from (Nawrot et al. 2001; Nawrot 2003). Data courtesy of Alexa Riehle, CNRS, Marseille, France

of variability, namely the trial-by-trial variation of the response onset time (Nawrot et al. 2003). This further impairs the trial-averaged estimate of the rate profile. Both factors will lead to an imperfect demodulation of the single-trial spike trains and, thus, to an increased dispersion of the inter-spike interval distribution and an increased empiric CV^2. An in-depth analysis of interval and count variability for this data set and a second monkey is provided in (Nawrot 2003). A time-resolved analysis of variability for this monkey (monkey 1) is provided in (Rickert et al. 2009).

In the model of slow rate fluctuations, we introduced a single time scale τ' for the temporal modulation. How does this time scale interact with the length T' of the empiric observation interval? In Fig. 3.5A, B the Fano-time curve $\mathsf{FF}(T')$ displays a nonmonotonic behavior resulting from two independent factors. For small observation times $T' \leq \mathsf{E}[X]$, the bias effect dominates, and the FF tends to unity as $T' \to 0$. With increasing $T' > \mathsf{E}[X]$, the slow intensity fluctuations cause a strong increase in count variance. As the positive serial interval correlations introduced by the rate fluctuation vanish for large correlation lags $i \gg \tau'$ (Fig. 3.5C), the FF saturates for large $T' \gg \tau'$ (Fig. 3.5B) because the spike count averages across the

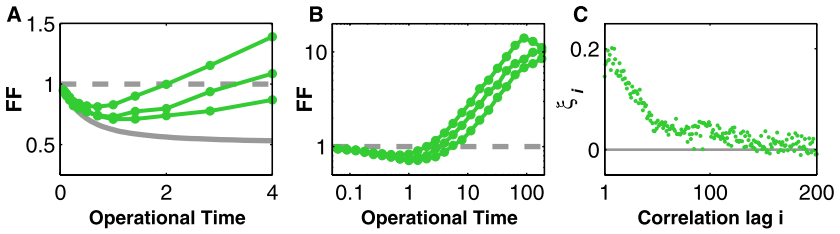

Fig. 3.5 Effect of slow rate modulations on the Fano-time curve and serial interval correlation. (**A**) $\mathsf{FF}(T')$ estimated from three individual realizations of a rate-modulated gamma process of order $\alpha = 2$. The process intensity $\phi(t)$ was modulated according to an MA process with log-normal noise ($\sigma = 200$, $\tau' = 20$). The *gray curve* represents the expectation value for the stationary gamma process. (**B**) For large observation intervals $T' \gg \tau'$, the Fano factor saturates. (**C**) Serial interval correlation coefficients diminish only for large serial correlation lags $i \gg \tau'$

stochastic fluctuations within the observation interval. Importantly, the trial-by-trial variability assumes a *minimum* for observation times $T' \approx \mathsf{E}[X]$, which is even more pronounced for a nonrenewal process with short-lived negative serial interval correlation (not shown; see Subsect. 3.3.3).

3.3.4.2 Task-Related Variability Dynamics

In a next step we extended the previous model for slow-rate modulation by adding a task-related response profile $\psi(t)$ during repeated trials that represents task-related activation of a neuron (or neural ensemble), e.g., in response to a stimulus. We model $\psi(t)$ with a Gaussian profile as in Subsect. 3.2.2. Now we have the situation that the same intensity profile repeats identically in each trial and adds to a fluctuating background $\phi(t)$. How does this affect the time-resolved variability? Figure 3.6 shows the result: The time-resolved Fano factor (blue curve) expresses a task-related dynamics. It is high during the initial phase of the trial before the response is triggered at $t = 0$ and again at the end of the trial. During the response, the FF strongly decreases and almost reaches the expected value for a stationary process with $\mathsf{FF} = 1/\alpha = 0.5$. This modulation can be easily understood: Whenever the firing rate is dominated by the task-related component ψ, the relative trial-by-trial fluctuations of the point process intensity, and thus of the empiric Fano factor, are minimal. Conversely, at times when the task-related component ψ is essentially zero, the spike count variability is dominated by the trial-to-trial variations due to the fluctuating intensity $\phi(t)$. The trial-based estimate of the $\overline{\mathsf{CV}^2}$ (dark green curve in Fig. 3.6D) does not show any apparent modulation. It correctly signifies the "true" stochasticity of the underlying gamma process except for a small bias that underestimates the expected value CV^2_∞ (see Subsect. 3.2.1). The ratio of $\mathsf{FF}/\mathsf{CV}^2$ in Fig. 3.6F combines both analyses and reveals dynamic deviations from the renewal hypothesis for which $\mathsf{FF} = \mathsf{CV}^2$.

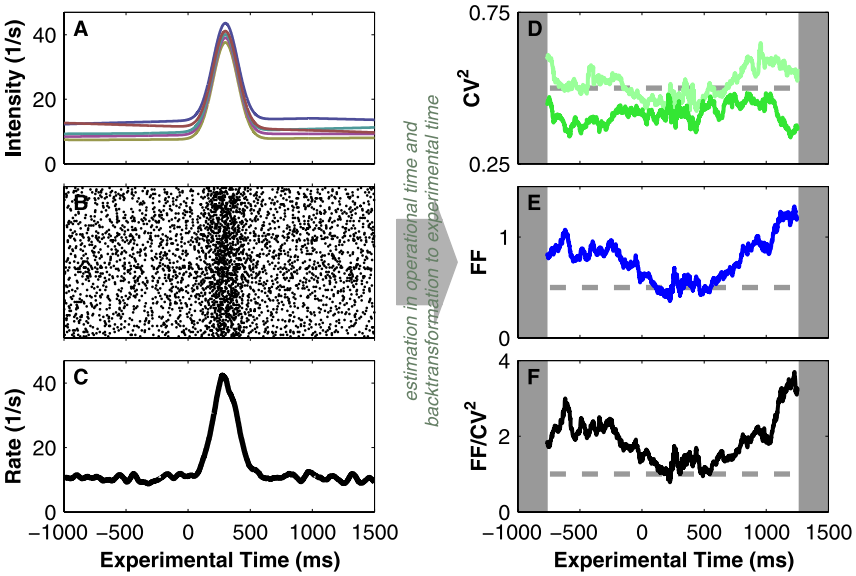

Fig. 3.6 Temporal modulation of spike count variability. (**A**) Five individual single trial intensities resulting from the superposition of the slow fluctuating background rate $\phi(t)$ and the task-related Gaussian modulation $\psi(t)$ (width $\sigma = 100$ ms, amplitude 30/s). (**B**) Spike raster from rate-modulated gamma process ($\alpha = 2$) realizations during 100 trials. (**C**) Kernel estimate of firing rate. (**D**) The time-resolved CV^2 (*light green*) is slightly modulated. The $\overline{\mathsf{CV}^2}$ (*dark green*) expresses an expected residual bias. (**E**) Time-resolved FF. (**F**) The ratio of $\mathsf{FF}/\overline{\mathsf{CV}^2}$ combines both analyses. Renewal prediction $\mathsf{FF}_\infty = \mathsf{CV}^2_\infty$ is indicated by the dashed line. Length of observation interval in **D**–**F** was $T' = 5$

Slow modulations of the output firing rate can be modeled by a nonstationary point process with time-varying intensity $\phi(t)$ and large time constant of modulation $\tau \gg \mathsf{E}[X]$. Such a modulation introduces a positive serial interval correlation ($\xi > 0$) and can *strongly increase the count variance*. The CV^2 is less sensitive to the nonstationarity. As a result, we observe that $\mathsf{FF} > \mathsf{CV}^2$ (see Fig. 3.4A). When this model is combined with a task-related intensity profile $\psi(t)$ which is identical in each trial, we observe a task-related modulation of the $\mathsf{FF}(t)$ (see Fig. 3.6).

3.4 Interpretations

3.4.1 Components of Single Neuron Variability

We may coarsely distinguish two components of single-neuron output variability (DeWeese and Zador 2004). The first component is attributable to neuron-*intrinsic*

sources such as synaptic failures and variability of synaptic event amplitude (e.g., DeWeese and Zador 2004; Nawrot et al. 2009; Zador 1998), noise caused by dendritic integration (e.g., Nawrot et al. 2009; Ariav et al. 2003; Shoham et al. 2005), and the reliability of spike initiation (e.g., Mainen and Sejnowski 1995; Nowak et al. 1997). The second, neuron-*extrinsic* component results from the spatial and temporal statistics of the synaptic input, i.e., the spiking statistics of the presynaptic excitatory, inhibitory, and neuromodulatory networks. Biological neurons are nonlinear, complex input–output devices that translate synaptic input into an output comprising a sequence of action potentials. When we analyze a neuron's output, we cannot distinguish between the different sources that caused the observed variability. Also, the concept of an "intensity" that we use in the framework of stochastic point process theory and that we like to interpret as "underlying rate" of neural firing has no physical equivalent in biological neurons. Therefore, we must base our interpretations on additional numerical studies of biophysical neuron models and experimental studies that focus on basic neural mechanisms in reduced preparations, which allow for highly controlled experimental conditions.

3.4.2 Serial Interval Correlations

Negative serial correlations have been reported for various neuron types in invertebrate and vertebrate systems (for review, see Farkhooi et al. 2009). These correlations are short-ranged, typically extending over only a few intervals, and they are of intermediate strength (e.g., $\xi \approx -0.2$ for cortical neurons) which results in a considerable reduction of the Fano factor of up to 50%. A plausible physiological explanation for this phenomenon are neuron-*intrinsic* mechanisms of spike frequency adaptation (SFA) (Benda and Herz 2003), which can introduce negative interval correlations in the output spike train when the neuron is in a steady state (i.e., for constant output rate), a result that has been established in various types of biophysical single neuron models (e.g., Wang 1998; Prescott and Sejnowski 2008; Muller et al. 2007). The reduction of the Fano factor implies that SFA neurons have an improved signal-to-noise ratio which increases the coding capacity of a rate code on slow time scales. On fast time scales, i.e., for very short observation windows, however, the Fano factor tends to unity (see Subsect. 3.3.1). In the frequency domain this results in a reduction of the low frequency noise (noise shaping; Chacron et al. 2001, 2005; Lindner et al. 2005; Chacron et al. 2007).

Systematic reports of negative serial correlations in experimental data are rare, in particular, in central brain structures such as the neocortex or the central insect brain. We briefly discuss two factors that may impair their empiric observation. First, serial correlation analysis assumes stationarity of the spike train. Any modulation of the firing rate will introduce positive serial correlations, which may conceal the negative correlations and increase the Fano factor (see Subsect. 3.3.4). The second issue is of technical nature. At extracellular electrodes we measure spikes that stem from

multiple neurons. Subsequent spike sorting procedures are error prone. Therefore the resulting single unit spike trains, to some extend, represent the activity of multiple neurons. From surrogate data we estimated that only 10–15% falsely assigned spikes can impair the detection of realistic negative serial correlations in recordings that comprise $\sim 1{,}000$ spikes (unpublished observation).

In the context of cross-correlation analysis of two (or more) simultaneously recorded neurons, renewal models are typically used to calculate the expected joint count distribution under the assumption that the neurons' activity is independent. Serial interval correlations affect the joint count distribution, and the renewal statistics may thus not be appropriate to test for deviations from independent spiking in SFA neurons (Grün et al. 2008).

3.4.3 Nonstationary Conditions in the Living Brain

There have been frequent reports on a large trial-by-trial variability in in vivo single-unit recordings, notably in the mammalian cortex where, with few exceptions, large average values of the Fano factor ($\mathsf{FF} \geq 1$) have been measured (e.g., Shadlen and Newsome 1998; for review, see Nawrot et al. 2008). This has lead to the dogma that the activity of cortical neurons is well characterized by Poisson statistics, which has subsequently become a benchmark for cortical network models. However, the large variability in vivo is contrasted by a series of in vitro studies that have quantified the output variability of pyramidal neurons for stationary input conditions. They used intracellular injection of currents generated by stochastic trains of excitatory and inhibitory synaptic inputs. It showed that the interval and count variability is in the range of $\mathsf{CV}^2 \approx \mathsf{FF} \in [0.1, 0.8]$, depending mostly on the relative fractions of excitation and inhibition (for review, see Nawrot et al. 2008). Negative serial interval correlations may further reduce the count variance such that $\mathsf{FF} < \mathsf{CV}^2$ (Fig. 3.3B; Nawrot et al. 2007). From these studies we may conclude that—for stationary input conditions—cortical neurons are more regular and less variable than the Poisson process.

What could be the reason for the discrepancy between the in vivo and in vitro results? One possibility is that in the living brain, stationary input conditions do not exist for neurons that are embedded in a recurrent and permanently active network. Local networks may be exposed to global changes of their activation state, e.g., due to homeostatic regulation, changes in the general state of arousal, plasticity, adaptation, etc., and they may be subject to top–down influences such as attentional modulation.

The simple stochastic model outlined in Subsect. 3.3.4 generates a random fluctuation of the firing intensity that underlies the stochastic production of spike events. We found that slow modulations of the intensity on time scales $\tau \gg \mathsf{E}[X]$ can strongly increase the count variance across independent observations, leading to large values of $\mathsf{FF} \gg 1$ as observed in vivo. The more important result, however, are expressed in the relation of count and interval variability. The CV^2 was only

slightly increased so that $\mathsf{FF} \gg \mathsf{CV}^2$, indicative of positive serial interval correlations due to the slow-rate modulations. This is what we also observed in the single-unit recordings from M1 in the behaving monkey (Fig. 3.4B; Nawrot et al. 2001; Nawrot 2003). These results suggest nonstationary input conditions in vivo, and they may indicate that the large in vivo variability does not characterize the stochastic nature of the individual neuron. Experimental studies (Nowak et al. 1997; Carandini 2004; Nawrot 2003) suggest that even mild fluctuations in the neuron's input are sufficient to cause a strong variability in the neuron's output. This is explained by the nonlinear transfer function of synaptic input drive and output firing rate. Mechanistically, such modulation of the presynaptic network input may be achieved by various means, e.g., through unbalancing of (presynaptic) excitatory and inhibitory networks, or through neuromodulatory regulation.

A number of theoretical models have investigated the effect of long-ranged temporal correlations in the driving noise of biophysical model neurons. These studies established the result of positive serial interval correlations in the output spike train and of the nonmonotonic behavior of the Fano-time curve (Chacron et al. 2001; Middleton et al. 2003; Schwalger and Schimansky-Geier 2008; Farkhooi et al. 2009). The strong increase of the Fano factor with increasing observation window, and in some cases also the characteristic of a nonmonotonic Fano-time dependence, has been reported in several experimental studies, e.g., in the cat striate (Teich et al. 1996) and in the monkey motor cortex (Nawrot 2003), in the LGN (Teich et al. 1997), in the retina (Teich et al. 1997), in medullary sympathetic neurons (Lewis et al. 2001), and most pronounced in the electrosensory afferents of the weakly electric fish (Ratnam and Nelson 2000). The fact that experimental Fano-time curves can express a minimum for a certain range of observation times may indicate that there exists an optimal temporal scale for information processing in these systems.

In Fig. 3.6A we added to the spontaneous intensity fluctuation $\phi(t)$ a task-related phasic component $\psi(t)$, which repeats identically in each trial. As a direct consequence, we observe a task-related modulation of the Fano factor. Indeed, this behavior has been repeatedly observed in motor cortical single-unit activity (Nawrot et al. 2001, 2003; Nawrot 2003; Churchland et al. 2006; Nawrot et al. 2008; Rickert et al. 2009) and, more recently, also in other cortical areas (Churchland et al. 2010). Thus, we may hypothesize that individual neurons or neural populations are specifically recruited for a computational task, e.g., the processing of a sensory stimulus, through a task-specific and dynamic input that overrides the background input, which represents ongoing activity and/or global changes of the network activation state.

Acknowledgements I am grateful to my collaborators Alexa Riehle and Clemens Boucsein for providing the experimental data that were reanalyzed in Figs. 3.3 and 3.4. I thank Stefan Rotter for valuable discussions on all theoretical aspects of this work and for helpful comments on the original version of this manuscript. I also thank Farzad Farkhooi for fruitful discussions. Funding was received from the German Federal Ministry of Education and Research (BMBF) through grant 01GQ0413 to the Bernstein Center Berlin and from the German Research Council (DFG) to the Collaborative Research Center *Theoretical Biology* (SFB 618).

Appendix

3.4.4 Matlab Tools for Simulation and Analysis

The following functions are available online in the FIND open source Matlab toolbox (Meier et al. 2008); http://find.bccn.uni-freiburg.de/.

makeKernel	builds simple kernel functions of predefined shape and normalized temporal width (Nawrot et al. 1999; Meier et al. 2008); used in Subsect. 3.2.2.
optimizeKernelWidth	estimates the optimal kernel width from spike train data according to a heuristic method (Nawrot et al. 1999; Meier et al. 2008); used in Subsect. 3.2.2.
sskernel	optimizes kernel width from spike train data according to the method by Shimazaki and Shinomoto (Shimazaki and Shinomoto 2010).
unWarpTime	demodulation of point or counting process according to a monotonic warp function. For details, see (Nawrot et al. 2008; Meier et al. 2008).
warpTime	inverse modulation of point or counting process.
gamprc/simulateGamma	simulates constant rate/rate-modulated gamma process using time rescaling.
arlogn/simSCP	simulates autoregressive log-normal point processes; used in Sect. 3.2. For details, see (Farkhooi et al. 2009).

3.4.5 Point Process Models

Chapter 1 of this book provides a formal introduction to stochastic point process theory, covering a number of issues that have been addressed in the present chapter. Chapter 16 deals in more detail with the simulation of stochastic point processes.

For any point process, *interval and count statistics are related*. Define the kth-order interval as $\tau_k = \sum_{i=1}^{k} X_i$. For an ordinary process, $\tau_k \leq t \iff N_{[0,t)} \geq k$. The distribution of τ_k relates to the distribution of event count N by

$$P\{\tau_k \leq t\} = P\{N_{[0,t)} \geq k\}.$$

The class of *renewal point processes* is widely used for the simulation of neural spiking. The renewal process is defined as a process for which all inter-event intervals are independent and identically distributed. Thus, we can define a particular renewal process by specifying its interval distribution. For nonbursting neurons, there are a number of distributions that have been repeatedly used, in particular, the (centralized) gamma distribution which includes the special case of the Poisson process, the log-normal distribution, and the inverse Gaussian distribution.

The interval distribution of the (centralized) *gamma process* is defined as

$$f_{\alpha,\rho}(x) = \begin{cases} \frac{1}{\Gamma(\alpha)}\,\rho(\rho x)^{\alpha-1}e^{-\rho x}, & x \geq 0, \\ 0, & x < 0, \end{cases}$$

where Γ denotes the gamma function, and $\alpha > 0$ and $\rho > 0$ are its two parameters. The mean interval is α/ρ, and the variance is α/ρ^2. For $\alpha = 1$, we obtain the Poisson process. For $\alpha > 1$, the gamma process is more regular and, for $0 < \alpha < 1$, more irregular than the Poisson process.

We used an *autoregressive model* to generate serially correlated interval series. A generalization of this model is described in detail elsewhere (Farkhooi et al. 2009). Assume that a series of random variables $Y_s = \beta Y_{s-1} + \varepsilon_s$, where ε_s is assumed to be normally distributed with mean μ and variance σ_N^2. β describes the serial dependence of the series Y_s. Then, the series

$$X_s = \exp(Y_s) = \exp(\beta Y_{s-1} + \varepsilon_s)$$

is asymptotically log-normal distributed. For parameterization according to definitions of $\mathsf{E}[Y]$ and CV, we used the following relations:

$$\sigma_N = \sqrt{\log\bigl(\mathsf{CV}^2 + 1\bigr)\bigl(1 - \beta^2\bigr)},$$
$$\mu = \log\bigl(\mathsf{E}[Y]\bigr) * (1 - \beta) - \sigma^2\,(1 - \beta)/\bigl(2\bigl(1 - \beta^2\bigr)\bigr).$$

In Subsect. 3.3.4, we simulated a *moving-average noise process* to generate a modulated rate function $\phi(t)$. To this end we drew random noise samples from a log-normal distribution with mean 1 (corresponding to unit rate) and standard deviation $\sigma = 200$ with a time resolution of 0.01 (operational time). In a second step we convolved the noise with a symmetric kernel of triangular shape and standard deviation $\sigma_k = 20$ (operational time). The resulting rate function fluctuates on a time scale that is 20 times larger than the mean interval.

References

Ariav G, Polsky A, Schiller J (2003) Submillisecond precision of the input–output transformation function mediated by fast sodium dendritic spikes in basal dendrites of CA1 pyramidal neurons. J Neurosci 23:7750–7758

Benda J (2002) Single neuron dynamics-models linking theory and experiment. Ph.D. thesis, Humboldt Universität zu Berlin. Ph.D. Dissertation

Benda J, Herz AVM (2003) A universal model for spike-frequency adaptation. Neural Comput 15(11):2523–2564. doi:10.1162/089976603322385063

Brown EN, Barbieri R, Ventura V, Kaas RE, Frank LM (2002) The time-rescaling theorem and its application to neural spike train data analysis. Neural Comput 14:325–346

Carandini M (2004) Amplification of trial-to-trial response variability by neurons in visual cortex. PLoS Biology 2(9):1483–1493

Chacron MJ, Lindner B, Longtin A (2007 Dec) Threshold fatigue and information transfer. J Comput Neurosci 23(3):301–311

Chacron MJ, Longtin A, Maler L (2001) Negative interspike interval correlations increase the neuronal capacity for encoding time-dependent stimuli. J Neurosci 21(14):5328–5343

Chacron MJ, Maler L, Bastian J (2005 May) Electroreceptor neuron dynamics shape information transmission. Nat Neurosci 8(5):673–678

Churchland M, Yu B, Cunningham J, Sugrue L, Cohen M, Corrado G, Newsome W, Clark A, Hosseini P, Scott B, Bradley D, Smith M, Kohn A, Movshon J, Armstrong K, Moore T, Chang S, Snyder L, Lisberger S, Priebe N, Finn I, Ferster D, Ryu S, Santhanam G, Sahani M, Shenoy K (2010) Stimulus onset quenches neural variability: a widespread cortical phenomenon. Nat Neurosci 13(3):369–378

Churchland MM, Yu BM, Ryu SI, Santhanam G, Shenoy KV (2006) Neural variability in premotor cortex provides a signature of motor preparation. J Neurosci 26(14):3697–3712

Cox DR, Lewis PAW (1966) The statistical analysis of series of events. Methuen's monographs on applied probability and statistics. Methuen, London

Davies RM, Gerstein GL, Baker SN (2006) Measurement of time-dependent changes in the irregularity of neural spiking. J Neurophysiol 96:906–918

DeWeese MR, Zador AM (2004) Shared and private variability in the auditory cortex. J Neurophysiol 92:1840–1855

Farkhooi F, Strube-Bloss M, Nawrot MP (2009) Serial correlation in neural spike trains: experimental evidence, stochastic modelling, and single neuron variability. Phys Rev E 79:021905

Grün S, Farkhooi F, Nawrot MP (2008) Significance of coincident spiking considering inter-spike interval variability and serial interval correlation. In: Frontiers comp neurosci conf abstr: neuroinformatics 2008. doi:10.3389/conf.neuro.11.2008.01.021

Holt GR, Softky WR, Koch C, Douglas RJ (1996) Comparison of discharge variability in vitro and in vivo in cat visual cortex neurons. J Neurophysiol 75(5):1806–1814

Knoblauch A, Palm G (2005) What is signal and what is noise in the brain? Biosystems 79:83–90

Lewis CD, Gebber GL, Larsen PD, Barman SM (2001) Long-term correlations in the spike trains of medullary sympathetic neurons. J Neurophysiol 85(4):1614–1622

Lindner B, Chacron MJ, Longtin A (2005 Aug) Integrate-and-fire neurons with threshold noise: a tractable model of how interspike interval correlations affect neuronal signal transmission. Phys Rev E Stat Nonlin Soft Matter Phys 72(2 Pt 1):021911

Mainen ZF, Sejnowski TJ (1995) Reliability of spike timing in neocortical neurons. Science 268:1503–1506

McFadden J (1962) On the lengths of intervals in stationary point processes. J Roy Stat Soc B 24:364–382

Meier R, Egert U, Aertsen A, Nawrot MP (2008) Find – a unified framework for neural data analysis. Neural Networks 21:1085–1093. http://find.bccn.uni-freiburg.de/

Middleton JW, Chacron MJ, Lindner B, Longtin A (2003 Aug) Firing statistics of a neuron model driven by long-range correlated noise. Phys Rev E Stat Nonlin Soft Matter Phys 68(2 Pt 1):021920

Miura K, Okada M, Amari S (2006) Estimating spiking irregularities under changing environments. Neural Comput 18:2359–2386

Muller E, Buesing L, Schemmel J, Meier K (2007) Spike-frequency adapting neural ensembles: Beyond mean adaptation and renewal theories. Neural Comput 19(11):2958–3010

Nawrot M, Aertsen A, Rotter S (1999) Single-trial estimation of neuronal firing rates: from single-neuron spike trains to population activity. J Neurosci Meth 94:81–92

Nawrot M, Aertsen A, Rotter S (2003) Elimination of response latency variability in neuronal spike trains. Biol Cybern 5(88):321–334

Nawrot MP (2003) Ongoing activity in cortical networks: noise, variability and context. Ph.D. thesis, Faculty of Biology, Albert-Ludwigs-University Freiburg, Germany. URN: nbn:de:bsz:25-opus-73426. http://www.freidok.uni-freiburg.de/volltexte/7342/

Nawrot MP, Benda J (2006) Two methods for time-resolved inter-spike interval analysis. In: Berlin neuroforum abstr, p 62

Nawrot MP, Boucsein C, Rodriguez-Molina V, Aertsen A, Grün S, Rotter S (2007) Serial interval statistics of spontaneous activity in cortical neurons in vivo and in vitro. Neurocomputing 70:1717–1722

Nawrot MP, Boucsein C, Rodriguez Molina V, Riehle A, Aertsen A, Rotter S (2008) Measurement of variability dynamics in cortical spike trains. J Neurosci Meth 169:374–390

Nawrot MP, Rodriguez V, Heck D, Riehle A, Aertsen A, Rotter S (2001) Trial-by-trial variability of spike trains in vivo and in vitro. Soc Neurosci Abstr 27:64.9

Nawrot MP, Schnepel P, Aertsen A, Boucsein C (2009) Precisely timed signal transmission in neocortical networks with reliable intermediate-range projections. Frontiers Neural Circ 3:1. doi:103389/neurv.04.001.2009

Nowak LG, Sanchez-Vives MV, McCormick DA (1997) Influence of low and high frequency inputs on spike timing in visual cortical neurons. Cereb Cortex 7:487–501

Parzen E (1962) On estimation of a probability density function and mode. Ann Math Statist 33:1065–1076

Ponce-Alvarez A, Kilavik B, Riehle A (2009) Comparison of local measures of spike time irregularity and relating variability to firing rate in motor cortical neurons. J Comput Neurosci. doi:10.1007/s10827-009-0158-2

Prescott SA, Sejnowski TJ (2008) Spike-rate coding and spike-time coding are affected oppositely by different adaptation mechanisms. J Neurosci 28:13649–13661

Ratnam R, Nelson M (2000) Nonrenewal statistics of electrosensory afferent spike trains: implications for the detection of weak sensory signals. J Neurosci 20(17):6672–6683

Reich DS, Victor JD, Knight BW (1998) The power ratio and the interval map: Spiking models and extracellular recordings. J Neurosci 18:10090–10104

Rickert J, Riehle A, Aertsen A, Rotter S, Nawrot MP (2009) Dynamic encoding of movement direction in motor cortical neurons. J Neurosci 29:13870–13882

Schwalger T, Schimansky-Geier L (2008) Interspike interval statistics of a leaky integrate-and-fire neuron driven by Gaussian noise with large correlation times. Phys Rev E Stat Nonlin Soft Matter Phys 77(3 Pt 1):031914

Shadlen MN, Newsome WT (1998) The variable discharge of cortical neurons: Implications for connectivity, computation, and information coding. J Neurosci 18(10):3870–3896

Shimazaki H, Shinomoto S (2010) Kernel bandwidth optimization in spike rate estimation. J Comput Neurosci. doi:10.1007/s10827-009-0180-4

Shimokawa T, Shinomoto S (2009) Estimating instantaneous irregularity of neuronal firing. Neural Comput 21:1931–1951

Shinomoto S, Miura K, Koyama S (2005) A measure of local variation of inter-spike intervals. Biosystems 79:67–72

Shoham S, O'Connor DH, Sarkisov DV, Wang SSH (2005) Rapid neurotransmitter uncaging in spatially defined patterns. Nature Meth 2:837–843

Teich MC, Heneghan C, Lowen SB, Ozaki T, Kaplan E (1997) Fractal character of the neural spike train in the visual system of the cat. J Opt Soc Am A Opt Image Sci Vis 14(3):529–546

Teich MC, Turcott RG, Siegel RM (1996) Temporal correlation in cat striate-cortex neural spike trains. IEEE Eng Med Biol Mag 15(5):79–87

Wang XJ (1998) Calcium coding and adaptive temporal computation in cortical pyramidal neurons. J Neurophysiol 79:1549–1566

Wiener MC (2003) An adjustment of the time-rescaling method for application to short-trial spike train data. Neural Comput 15:2565–2576

Zador A (1998) Impact of synaptic unreliability on the information transmitted by spiking neurons. J Neurophysiol 79:1219–1229

Chapter 4
Processing of Phase-Locked Spikes and Periodic Signals

Go Ashida, Hermann Wagner, and Catherine E. Carr

Abstract Studies of synchrony in the nervous system have revealed circuits specialized for the encoding and processing of temporal information. Periodic signals are generally coded by phase-locked action potentials and often processed in a dedicated pathway in parallel with other stimulus variables. We discuss circular statistics and current data analysis tools to quantify phase locking such as vector strength.

4.1 Introduction

Accurate coding of temporal information is widespread in the nervous system, and many theories in sensory biology depend upon the detection of signals that are correlated in time (see also Chap. 9). This chapter specifically deals with periodic signals, where some of the component variables are angular. A famous early example of angular data in biology deals with bird migrations, which are tracked by compass direction, yielding a distribution on a unit circle. Originally, the work tracked caged migrants, which were observed to orient in their normal migratory direction, with southerly headings in autumn and northerly headings in spring in the northern hemisphere. A vector sum of all headings yielded a mean direction of migration (for review, see Mouritsen and Ritz 2005). Other biological examples include events that are periodic in time, such as those dependent on time of day (Liu et al. 2006), or upon a repeating, periodic stimulus, such as breathing. Circular statistics have been developed to analyze this class of data (Batschelet 1981) and later adapted to measure neural data and relate spike timing to a number of different periodic oscillatory signals (Goldberg and Brown 1969).

Typical periodic phenomena discussed in this chapter include phase-locked responses to pure-tone sound stimuli (Köppl 1997) and the periodic discharges of the

G. Ashida (✉)
Department of Biology, University of Maryland, 1210 Biology–Psychology Building, College Park, MD 20742, USA
e-mail: ashida@umd.edu

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_4, © Springer Science+Business Media, LLC 2010

electric organ of weakly electric fish (Kawasaki and Guo 1996). Other notable examples discussed in other chapters include cortical oscillations (Engel et al. 1990) and the analysis of the oscillations underlying rhythmic movement (Rybak et al. 2006). Readers are also directed to a review of temporal processing in the nervous system (Mauk and Buonomano 2004). In the auditory and electrosensory examples used in this chapter, receptors encode the phase of signals with microsecond precision, and central circuits compute relevant features of the stimulus, like the phase difference between two ears (Carr 1993). These periodic signals are generally coded by phase-locked action potentials and processed in a dedicated pathway in parallel with other stimulus variables. In this chapter, we will first discuss circular statistics and the analysis of angular data, then the data analysis tools used to quantify phase-locking, followed by modeling of phase-locked spike sequences.

4.2 Analysis of Angular Data

Analyses of circular data differ from the application of conventional statistics. For example, consider $\alpha = 30°$ and $\beta = 70°$, their mean $\mu = 50°$, both geometrically and arithmetically. However, consider $\alpha' = \alpha = 30°$ and $\beta' = \beta + 360° = 430°$; their arithmetical mean $\mu' = 230°$ is geometrically incorrect. α, β are geometrically the same as α', β', although their arithmetical means μ, μ' are different. Thus conventional analyses cannot be directly applied to derive the mean angle. An appropriate approach to compute the mean angle converts all angles into corresponding points on a unit circle, i.e., α can be converted to a vector $\overrightarrow{a} = (\cos\alpha, \sin\alpha)$, and β to $\overrightarrow{b} = (\cos\beta, \sin\beta)$. The mean vector $\overrightarrow{m}$ of the two vectors $\overrightarrow{a}$ and $\overrightarrow{b}$ can then be obtained as

$$\overrightarrow{m} = \frac{\overrightarrow{a} + \overrightarrow{b}}{2} = \left(\frac{\cos\alpha + \cos\beta}{2}, \frac{\sin\alpha + \sin\beta}{2} \right).$$

The value of the mean angle μ can be determined as the angle of $\overrightarrow{m}$ and is calculated as

$$\mu = \operatorname{atan}2\left(\frac{\sin\alpha + \sin\beta}{2}, \frac{\cos\alpha + \cos\beta}{2} \right),$$

where

$$\operatorname{atan2}(y, x) = \begin{cases} \arctan(y/x), & x > 0, \\ 180° + \arctan(y/x), & x < 0,\ y \geq 0, \\ -180° + \arctan(y/x), & x < 0,\ y < 0, \\ 90°, & x = 0,\ y > 0, \\ -90°, & x = 0,\ y < 0, \\ \text{undefined}, & x = 0,\ y = 0. \end{cases}$$

The function $\operatorname{atan}2(y, x)$ gives the angle between the positive x-axis and the vector (x, y) in the range of $(-180°, 180°]$.

This approach can also be generalized to determine the average of many angles. Consider, for instance, angles $\theta_1, \theta_2, \ldots, \theta_n$. Their average, also referred to as mean phase, can be calculated as

$$\theta = \operatorname{atan2}\left(\frac{1}{n}\sum_{j=1}^{n}\sin\theta_j, \frac{1}{n}\sum_{j=1}^{n}\cos\theta_j\right) = \operatorname{atan2}\left(\sum_{j=1}^{n}\sin\theta_j, \sum_{j=1}^{n}\cos\theta_j\right).$$

Another important quantity to compute when analyzing directional data is the strength of the vector-field. Consider vectors $\overrightarrow{v}$ and $\overrightarrow{w}$ of unit length subtending angles of 45° and 225° on the positive x-axis. The angle subtended by $\overrightarrow{w}$ on the negative x-axis would be 45°. Since they are two vectors that have equal magnitude and opposite directions, they would effectively cancel out each other, and the resultant (= sum) vector would therefore be a zero vector. While this problem itself might seem trivial, the calculation of the resultant vector length benefits from analysis when considering a larger set of vectors with different angles. The resultant vector length is resolved by calculating the projections of the vectors on the x-axis and y-axis of a unit circle. Having computed the Abscissas x_j and Ordinates y_j for all the individual vectors $\overrightarrow{v_j}$, we sum them to obtain the Abscissa x and Ordinate y of the resultant vector, so we can compute the length L of the resultant vector as $L = \sqrt{x^2 + y^2}$. In the example being discussed here, the pair (Abscissa, Ordinate) for $\overrightarrow{v}$ and $\overrightarrow{w}$ would be $(\cos 45°, \sin 45°)$ and $(\cos 225°, \sin 225°)$. The resultant vector would be $(\cos 45° + \cos 225°, \sin 45° + \sin 225°) = (0, 0)$, and thus its length $L = 0$.

Generalizing the above explanation for a set of n unit vectors that subtend angles $\theta_1, \theta_2, \ldots, \theta_n$, the (Abscissa x, Ordinate y) pair of the resultant vector would be

$$(x, y) = \left(\sum_{j=1}^{n} x_j, \sum_{j=1}^{n} y_j\right) = \left(\sum_{j=1}^{n}\cos\theta_j, \sum_{j=1}^{n}\sin\theta_j\right).$$

The resultant vector length L would be

$$L = \sqrt{\left(\sum_{j=1}^{n}\cos\theta_j\right)^2 + \left(\sum_{j=1}^{n}\sin\theta_j\right)^2}.$$

If all the individual vectors are unidirectional, then the length of the resulting vector would simply be the arithmetic sum of individual vector lengths, i.e., $L = n$. This property leads us to derive the quantity termed vector strength, which is defined as the length of the resultant vector L divided by the number of vectors n. Namely, vector strength r is the ratio

$$r = \frac{L}{n} = \frac{1}{n}\sqrt{\left(\sum_{j=1}^{n}\cos\theta_j\right)^2 + \left(\sum_{j=1}^{n}\sin\theta_j\right)^2}.$$

Note that vector strength r can also be regarded as the length of the mean vector

$$(\bar{x}, \bar{y}) = \left(\frac{1}{n} \sum_{j=1}^{n} x_j, \frac{1}{n} \sum_{j=1}^{n} y_j \right) = \left(\frac{1}{n} \sum_{j=1}^{n} \cos\theta_j, \frac{1}{n} \sum_{j=1}^{n} \sin\theta_j \right).$$

If all the vectors cancel out each other, the resultant vector length is 0, and consequently vector strength is also 0. Vector strength becomes 1 if and only if all the vectors are unidirectional. In other cases, vector strength takes values between 0 and 1.

In this section we have used degrees for angles and phases so that beginners can readily understand. In the following text, we will mainly use radians and cycles rather than degrees so that theoretical considerations are straightforward. Note that the definition of vector strength does not depend on the choice of degrees or radians.

4.3 Calculating Vector Strength

In 1969, Goldberg and Brown (1969) introduced circular statistics and vector strength to the analysis of neural data. They quantified the synchronization or phase-locking of the discharges of auditory neurons to low-frequency sound stimuli. In measuring the degree of phase-locking (Fig. 4.1A), each spike is considered as a vector of unit length with a phase angle θ_j $(0 \leq \theta_j < 2\pi)$ by taking

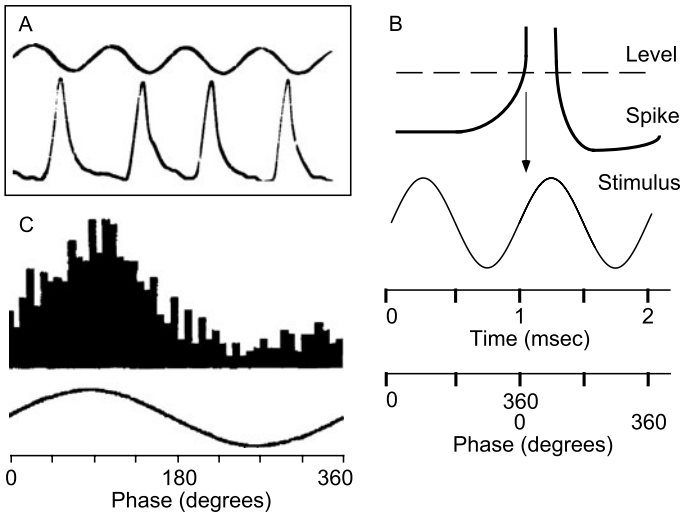

Fig. 4.1 Phase-locked action potentials. **A**. Timing information encoded by phase-locked action potentials. This example shows an intracellular recording from a phase-coding afferent from the electrosensory lateral line lobe (*bottom*) and a 300-Hz sinusoidal stimulus (*top*). Taken from Carr et al. (1986). **B**. Spike timing measured with respect to the phase of a 1 kHz stimulus. Typically a zero-crossing detector or Schmitt trigger circuit creates time stamps of the spikes. **C**. Spike phase plotted in a period histogram. Example from an electrosensory lateral line neuron in *Gymnarchus niloticus*. Taken from Kawasaki and Guo (1996)

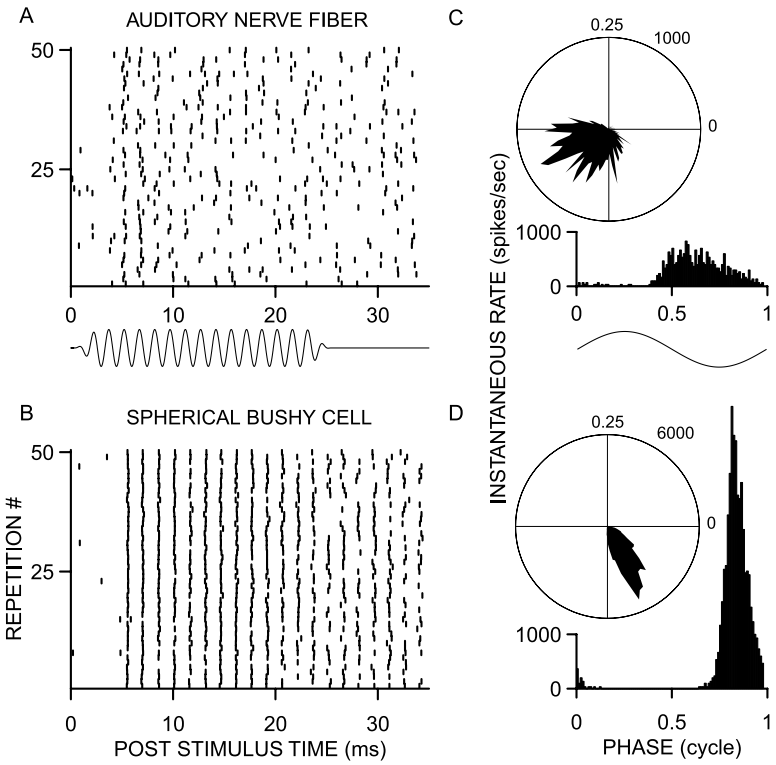

Fig. 4.2 Variation in phase locking. **A–B**. Phase-locking recorded in an auditory nerve fiber (**A**) and in a bushy cell in the cat cochlear nucleus (**B**). The plots show the spike occurrence in *dot rasters* to 50 repetitions of short (25 ms) tones at the characteristic frequency (670 Hz). **C–D**. The same data are shown as both period and cycle histograms of responses to 100 repetitions. The vector strength is 0.64 for the auditory nerve fiber and 0.93 for the spherical bushy cell (mean phase is 0.64 and 0.84 cycles, respectively). The total number of spikes in these histograms is 667 (*nerve fiber*) and 1775 (*bushy cell*). Note that the period histograms are on the same scale, but not the polar plots (*cycle histograms*), where the outer circle indicates instantaneous rates of 1,000 and 6,000 spikes/sec. Taken from Joris and Smith (2008)

$\theta_j = 2\pi f t_j \pmod{2\pi}$, where t_j is the timing of jth spike, and f is the stimulus frequency. The n vectors characterizing a spike train are treated as a distribution on a unit circle (Greenwood and Durand 1955), and θ_j gives the phase relation between the sinusoidal stimulus waveform and the unit's discharge (Fig. 4.1B). The mean vector, mean phase, and vector strength can then be calculated according to the formulae shown in the previous section. The mean phase, or the direction of the mean vector, gives the average phase relation between the stimulus and discharge (see also Fig. 4.2). The vector strength, sometimes called the "synchronization index", provides information of spike synchrony with respect to the stimulus. A vector strength of 1 implies perfect phase-locking; a value of 0, a random relationship between stimulus and response.

In order to visually examine phase-locked spiking activities, period histograms are often used. A period histogram shows a single stimulus cycle, with spikes plotted with respect to stimulus phase angle (Fig. 4.1C). Spike phase may be plotted with any radial measure (radians, degrees, cycles, etc.). If most action potentials fall within a small range of phase angles, vector strength is expected to be high (near 1). If action potentials are evenly distributed in the histogram, vector strength would be close to 0. Cycle histograms, which show spike phase distribution on a circle, are sometimes preferred because periodicity can be understood more clearly (Fig. 4.2C–D).

4.4 Sources of Variation in Phase-Locking, Frequency Dependence and Temporal Dispersion

Variation in phase-locking can emerge from various sources in biological systems. Most prominently, phase-locking in the auditory system is limited by stimulus frequency. Vector strength generally decreases with increasing frequency (Fig. 4.3). Recordings from auditory nerve fibers have a statistical tendency to phase-lock to the waveform of the acoustic stimulus up to a certain frequency (Kiang et al. 1966). Spikes occur most frequently at a particular phase of the tone, although not necessarily in every tonal cycle (see raster plots in Figs. 4.2A–B and 4.5D). Thus, the discharge pattern of a cochlear nerve fiber can encode the phase of a tone with a frequency considerably above 1,000 Hz, even though the average discharge rate is low. Eventually biophysical constraints limit the ability of biological systems to phase-lock, presumably when neurotransmitter release and membrane time constants are too slow to modulate at the stimulus frequency. The upper limit of phase-locking quality in barn owl auditory nerve fibers appears to be the highest measured, with vector strength measures of 0.2 at 9 kHz (Köppl 1997) (Fig. 4.3).

There are a number of adaptations for precise temporal coding (for review, see Carr and Soares 2002). For example, the auditory nerve forms large endbulb or caliciform synapses with multiple sites of synaptic contact on the cell body to provide the substrate for the preservation of the phase-locked action potentials between the auditory nerve fibers and first-order auditory neurons (Carr et al. 2001; Trussell 2008). This timing information is degraded for high-frequency sounds, presumably because of low pass filter effects in the hair cell, since electrical stimulation of the auditory nerve increases the high-frequency cutoff of phase-locking (Dynes and Delgutte 1992). A comparison of maximal vector strength as a function of frequency in the auditory nerve of several species is provided in Fig. 4.3.

In the ear, sensory hair cells phase-lock to sound frequencies that are higher than maximal firing rates of auditory nerve fibers, which are about 300 Hz. As discussed above, auditory nerve axons can skip cycles (see Figs. 4.2A–B, 4.5D), and thus the upper limit of phase-locking is not imposed by refractoriness (Joris and Smith 2008; Wever and Bray 1930). Instead the upper limit of phase-locking appears to depend on biophysical features of the inner ear (Kidd and Weiss 1990) and to be about 4–5 kHz in cats and about 9 kHz in the barn owl (Fig. 4.3).

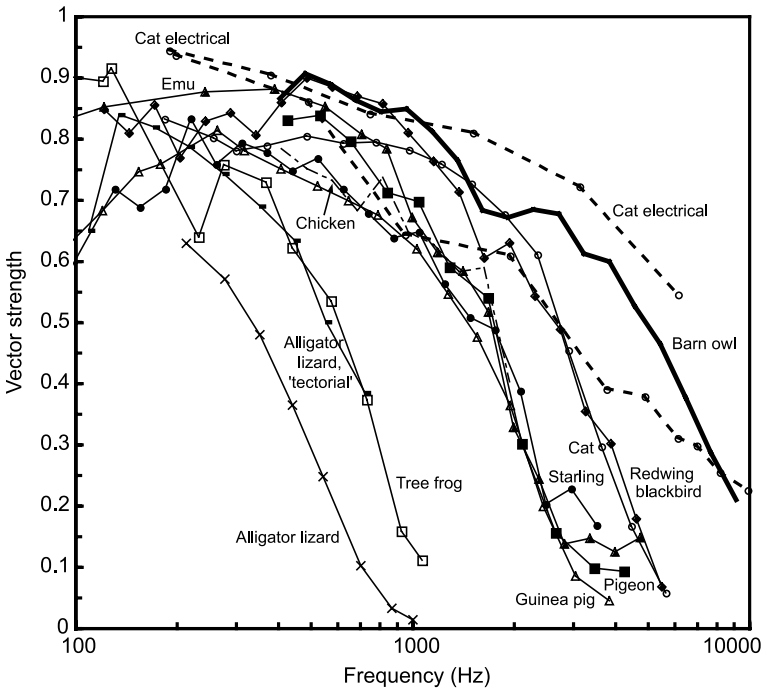

Fig. 4.3 Limits of phase locking. Köppl (1997) assembled data on phase locking for a number of birds and mammals. *Barn owl* (Köppl 1997), *emu* (Manley et al. 1997), *redwing blackbird* (Sachs and Sinnott 1978), *starling* (Gleich and Narins 1988), *pigeon* (Hill et al. 1989), *chicken* (Salvi et al. 1992). For the *cat* and *guinea pig*, average values in 0.1 decade bins as determined by (Weiss and Rose 1988) from the original data (Johnson 1980; Palmer and Russell 1986) were used. Two different data sets for electrical stimulation in the *cat* (Dynes and Delgutte 1992; Hartmann 1987) are also plotted. Taken from Köppl (1997). Data for the *alligator lizard* and *tree frog* are added

Vector strength increases with increasing sound loudness. Phase-locking often begins below rate threshold, then increases. Individual units vary in the rate at which their vector strength increases, reaching a plateau of vector strength somewhere between near rate threshold and about 20 dB above it. These features are illustrated in an example of phase locking in pigeon auditory nerve in Fig. 4.4A. For each frequency, vector strength increases with level, while vector strength declines with increasing frequency (compare along rows in Fig. 4.4A).

Even though vector strength deteriorates with increasing frequency (Fig. 4.4B), the precision of spike timing actually increases over a certain frequency range. This decrease in jitter or increase in precision is measured as temporal dispersion (Hill et al. 1989). Phase-locking at increasingly higher frequencies requires an increasingly better temporal precision of the neuron because the stimulus period is steadily decreasing on an absolute time scale. For example, to achieve the same vector strength at 10 kHz (0.1 msec period) as at 1 kHz (1 msec period), the temporal jitter or dis-

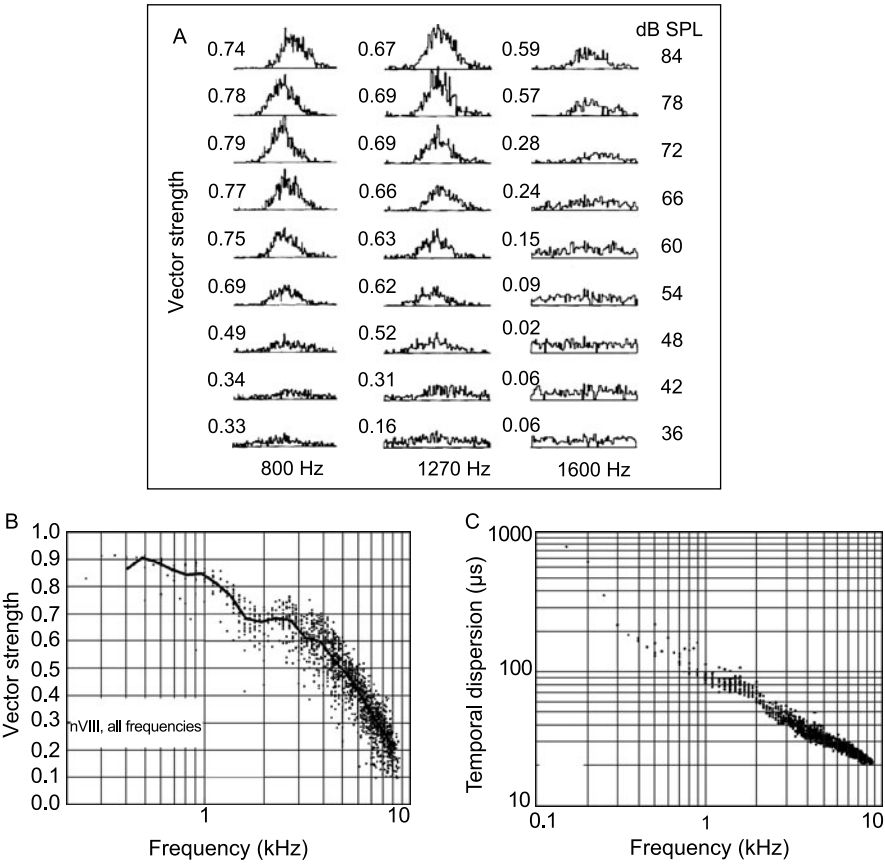

Fig. 4.4 Changes in phase locking with sound frequency and intensity. **A**. Period histograms obtained from a pigeon auditory nerve fiber for a range of sound frequencies and intensities. *Columns* show histograms for different sound pressure levels (dB SPL), and *rows* show histograms for different sound frequencies. Vector strengths are shown at the *left* of each column, and each histogram has been adjusted to align its mode to π radians. Modified from Hill et al. (1989). **B**. Vector strength as a function of stimulus frequency. The *solid lines* connect the median values calculated for quarter octave bins for barn owl auditory nerve fibers. **C**. Temporal dispersion as a function of stimulus frequency for the same data set of auditory nerve fibers. **B** and **C** from Köppl (1997)

persion of spikes must decrease by one order of magnitude. Temporal dispersion d (in seconds) can be related to vector strength r according to

$$d = \frac{\sqrt{2(1-r)}}{2\pi f},$$

where f is the frequency (in Hz) (Hill et al. 1989; Köppl 1997) (see also Paolini et al. 2001 for a slightly different formula). A vector strength of 0 would indicate that there is so much dispersion that a mean angle cannot be determined, while a vector strength of 1 means that all the data are concentrated at one direction. Köppl

(1997) derived values for temporal dispersion for a large data set from barn owl auditory nerve (Fig. 4.4C). Temporal dispersion was highest, up to 1 msec, at the lowest frequencies tested and fell with increasing frequency, approaching 22 μsec at 9–10 kHz. The overall decline was well described by a power law (Figs. 4.3, 4.4B–C).

4.5 Rayleigh Test and Statistical Tests of Vector Strength

It is important to know what confidence can be placed in the vector strength calculated from the spike data. To assess the significance of any periodicity found, the chance of getting a vector strength r from a list of random numbers needs to be found. This can be done by generating a set of random vectors uniformly distributed on a unit circle, as many as the original spike dataset, and computing vector strength. This is then repeated many times. For example, if on 10 occasions out of the 1,000, the vector strength from the list of random numbers was equal to or exceeded that obtained from the real dataset, then there is a 1% chance of obtaining this result from a random dataset or a dataset with no signal.

Another way to examine the statistical significance of vector strength r is to perform a test derived from Rayleigh's studies of periodic motion (Rayleigh 1894). The Rayleigh test asks how large a sample size n must be to indicate confidently a nonuniform population distribution. The null hypothesis H_0: The sampled distribution is uniformly distributed around the circle. The alternative hypothesis H_A: The population is not a uniform circular distribution. "Rayleigh's R" defined as $R = nr$ and "Rayleigh's z" defined as $z = R^2/n = nr^2$ have been used for testing the null hypothesis of no population mean direction (Batschelet 1981; Fisher 1993; Mardia 1972). The quantity $2z$ is known to obey the chi-square distribution with 2 degrees of freedom in the case of large samples (typically $n > 15$). Thus a table of the chi-square distribution can be used for testing. Or, more directly, the significance probability p of vector strength r can be approximated as

$$p \simeq \exp(-z) = \exp\left(-nr^2\right)$$

for a larger set of samples (typically $n > 50$) (Fisher 1993). Therefore, for example, a vector strength of $r = 0.1$ with $n = 500$ samples yields statistical significance of $p < 0.01$ (Köppl 1997). For more information on the Rayleigh test, readers are referred to texts on circular statistics (Batschelet 1981; Fisher 1993; Mardia 1972).

4.6 Relationship to Fourier Analysis

In this section we give a brief description on the relationship between vector strength and the Fourier analysis. By regarding each spike as a delta function, a train containing n spikes can be written as

$$s(t) = \sum_{j=1}^{n} \delta(t - t_j).$$

The Fourier transform of $s(t)$ is

$$\begin{aligned}\hat{s} = \hat{s}(f) &= \int_{-\infty}^{\infty} s(t)\mathrm{e}^{-2\pi i f t}\,dt \\ &= \int_{-\infty}^{\infty} \sum_{j=1}^{n} \delta(t - t_j)\big(\cos(2\pi f t) - i\sin(2\pi f t)\big)\,dt \\ &= \sum_{j=1}^{n} \cos(2\pi f t_j) - i\sum_{j=1}^{n} \sin(2\pi f t_j).\end{aligned}$$

Then we have

$$|\hat{s}|^2 = \left(\sum_{j=1}^{n} \cos(2\pi f t_j)\right)^2 + \left(\sum_{j=1}^{n} \sin(2\pi f t_j)\right)^2.$$

Using

$$\int_{-\infty}^{\infty} \delta(t - t_j)\,dt = 1 \quad (\text{for every } t_j),$$

we obtain

$$\int_{-\infty}^{\infty} s(t)\,dt = \sum_{j=1}^{n} \int_{-\infty}^{\infty} \delta(t - t_j)\,dt = \sum_{j=1}^{n} 1 = n.$$

Therefore, vector strength r can be rewritten as

$$r = \frac{|\hat{s}|}{n} = \left|\int_{-\infty}^{\infty} s(t)\mathrm{e}^{-2\pi i f t}\,dt\right| \Big/ \int_{-\infty}^{\infty} s(t)\,dt.$$

From this equation relating vector strength with the Fourier transform, we can summarize: "Vector strength is the Fourier component of the spike sequence at the stimulus frequency normalized by the total number of spikes".

4.7 Additional Measures and Some Technical Issues

An additional metric, the correlation index, was developed by Joris et al. (2006). The correlation index is the peak value of the normalized shuffled autocorrelogram, and it provides a quantitative summary of temporal structure in the neural response (see Chap. 6 for correlation analyses). It quantifies the degree to which spikes are generated at the same time with repeated stimulus presentations, with both precision and consistency of spikes required for large values. The correlation index is normalized for average firing rate, and the same research group have developed an unnormalized metric (maximal coincidence rate) (Louage et al. 2005). Technically, the correlation index and correlograms are easier to measure than vector strength or measures of reverse correlation because they can be measured more quickly (vector

strength measures require many stimulus presentations). The correlation index can also be used for aperiodic stimuli and does not require knowledge of the stimulus. However, there are some disadvantages to the use of the correlation index, because phase information is discarded.

A technical issue, raised by Joris et al. (2006), and germane to all measures of temporal precision, relates to the measure of spike timing. From recorded neural responses, the timing of each spike is usually determined by a level detector or a Schmitt trigger circuit and converted to a standard pulse (Fig. 4.1B). A Schmitt trigger switches its output when the input passes upward through a reference voltage, then prevents switching back to the other state until the input passes through a lower threshold voltage. This stabilizes the switching against rapid triggering by noise as it passes the trigger point. However, Joris et al. (2006) found that changes in the voltage level of triggering relative to the spike maximum could produce noticeable shifts, and therefore spike triggering should be done at a consistent point of the spike waveform throughout the recording. This problem may be eliminated by using a peak-detection circuit instead of a Schmitt trigger (see also Sullivan and Konishi 1984 for discussion).

There are two more considerations to be noted when measuring phase-locking: sampling frequency and good signal-to-noise ratio. First, timing of every spike needs to be recorded with high temporal resolution, typically around 5–10 μsec for the auditory system responding to signals from 1 to 10 kHz. In cases where microsecond precision is required, it is possible for measurement error to interfere with real variation. For example, assuming that the signal frequency is 2 kHz (the length of one cycle is 500 μsec) and the sampling frequency is 200 kHz (the length of one cycle is 5 μsec), estimation error of spike timing will be less than 1%, and vector strength is expected to be calculated with similar accuracy. However, if the sampling frequency is 10 kHz (the length of one cycle is 100 μsec), estimation error of spike timing could be as much as 20% and lead to a significant error in the calculation of vector strength. Systematic investigation needs to be conducted to more accurately estimate the influence of measurement errors on vector strength calculation. Second, to obtain good vector strength estimation, good signal-to-noise is also important. Since spike timing is recorded with such high temporal resolution, low signal-to-noise in the recording can produce a temporal jitter in zero crossing detection, degrading phase-locking, while the presence of a stimulus artifact in the recording can artificially increase phase-locking (the Johnson effect, Johnson 1980).

4.8 Modeling Phase-Locked Spike Trains

In the modeling of neuronal spike trains, the Poisson process is one of the simplest and most common computational tools. Let $N(t)$ be the number of spikes that occurred before time t. $N(t)$ is called a Poisson process with mean rate m if

$$\text{Prob}\big(N(t+\Delta t) - N(t) = k\big) = \frac{(m\Delta t)^k}{k!}\exp(-m\Delta t).$$

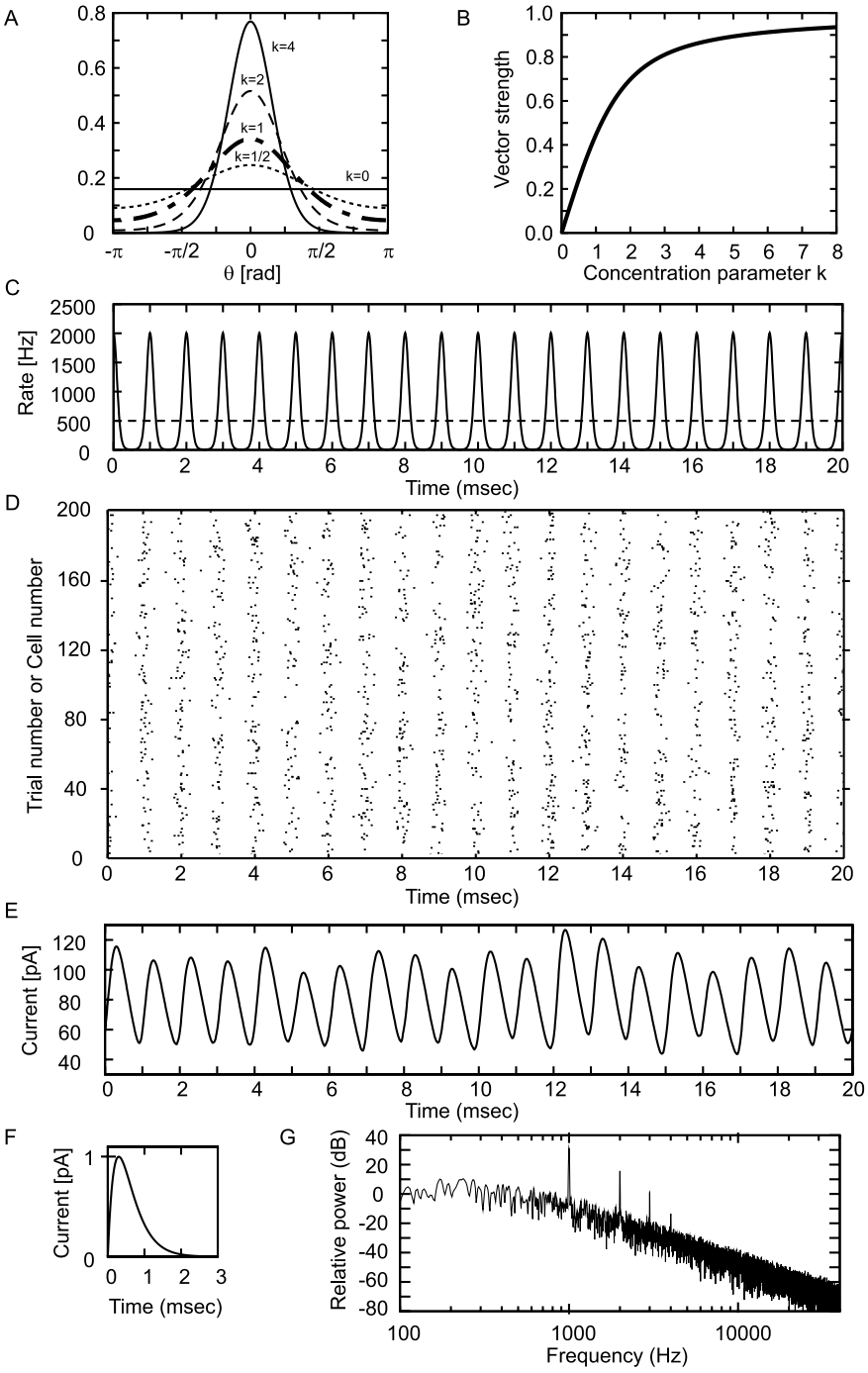

Fig. 4.5 Continued

Note that $N(t+\Delta t)-N(t)$ means the number of spikes that occurred between time t and time $t+\Delta t$. If the length of the time interval Δt is small enough, namely $m\Delta t \ll 1$, probabilities of spike counts in the interval can be approximated as:

$$\begin{aligned}
&\mathrm{Prob}\big(N(t+\Delta t)-N(t)=0\big)=\exp(-m\Delta t)\simeq 1-m\Delta t,\\
&\mathrm{Prob}\big(N(t+\Delta t)-N(t)=1\big)=(m\Delta t)\exp(-m\Delta t)\simeq m\Delta t,\\
&\mathrm{Prob}\big(N(t+\Delta t)-N(t)>1\big)\simeq 0.
\end{aligned}$$

The mean rate m, which is also called the intensity, indicates the average number of spikes per unit time. If m changes with time t, $N(t)$ is called an inhomogeneous Poisson process (Gerstner et al. 1996) (see, for example, Kempter et al. 1998 for more detailed formulation). In generating periodic (phase-locked) spike trains, periodic functions are adopted as $m=m(t)$. The von Mises distribution, which is also called the circular normal distribution, is one of the most commonly used probability distributions for this purpose. Its probability density function is defined as

$$p_{a,k}(x)=\frac{1}{2\pi I_0(k)}\exp\big(k\cos(x-a)\big).$$

The period of this function is 2π. a is a parameter determining the "mean direction" of the distribution, and k is a parameter determining the "concentration" (Fig. 4.5A). The distribution is uniform for $k=0$ and becomes steep around a as k increases. $I_0(k)$ is the modified Bessel function of order 0 satisfying

$$I_0(k)=\frac{1}{2\pi}\int_{-\pi}^{\pi}\exp(k\cos x)\,dx,$$

and thus $1/2\pi I_0(k)$ is the normalization constant assuring

$$\int_{-\pi}^{\pi}p_{a,k}(x)\,dx=1.$$

By using the von Mises distribution, periodic signal can be modeled as an inhomogeneous Poisson process with rate

$$m(t)=2\pi\bar{R}\times p_{a,k}(2\pi f t),$$

where $\bar{R}$ is the average spiking rate over time, and f is the signal frequency (see Fig. 4.5C for example). Collection of many periodic spikes (typically several hundred to several thousand) gives a period histogram with the same shape as the distribution function used for the inhomogeneous Poisson process.

Fig. 4.5 Modeling phase-locked spikes. **A**. Probability density function of the von Mises distribution (mean direction $a=0$). **B**. Concentration parameter k of the von Mises distribution and vector strength. **C**. Rate function of the inhomogeneous Poisson spike trains (*solid line*) oscillating around the temporal average (*dashed line*). Signal frequency is 1 kHz. The concentration parameter $k=2.8713$ so that vector strength $r=0.8$. **D**. Raster plot of simulated spike trains. Each *dot* indicates a spike. 200 trials are shown. **E**. Simulated synaptic input oscillating with a frequency of 1 kHz. **F**. Synaptic input modeled as an alpha-function ($\tau=0.3$ msec). **G**. Power spectrum calculated from the Fourier transform of the simulated synaptic input (**E**). Note the main peak at 1 kHz

We can calculate vector strength from the distribution function by the continuous extension of the definition of vector strength. Let $q(x)$ be a probability density function over $[-\pi, \pi]$. The mean vector $(\bar{x}, \bar{y})$ of many spikes generated by the inhomogeneous Poisson process with rate $m(t) = 2\pi \bar{R} q(2\pi f t)$ can be obtained as

$$(\bar{x}, \bar{y}) = \left(\int_{-\pi}^{\pi} \cos(x) q(x)\, dx, \int_{-\pi}^{\pi} \sin(x) q(x)\, dx \right),$$

and then vector strength $r = \sqrt{\bar{x}^2 + \bar{y}^2}$ can be calculated.

For the von Mises distribution, probability density $q(x) = p_{a,k}(x)$. We can assume the mean direction $a = 0$ in calculating vector strength because $p_{a,k}(x)$ is a periodic function with period 2π. Since $p_{0,k}(x)$ is symmetric about $x = 0$, vector strength r can simply be calculated as

$$r = \int_{-\pi}^{\pi} \cos(x) p_{0,k}(x)\, dx = \frac{1}{2\pi I_0(k)} \int_{-\pi}^{\pi} \exp\big(k \cos(x)\big) \cdot \cos(x)\, dx.$$

Vector strength r is zero for $k = 0$ and monotonically increases with the concentration parameter k (Fig. 4.5B).

An example of simulated periodic spike trains is shown in Fig. 4.5C–D. Spiking rate changes periodically around the average rate over time (Fig. 4.5C). Since signal frequency was set at 1 kHz and average spiking rate was 500 Hz, spikes occur not in every single cycle but about every two cycles on average. The raster plot shows that spikes are concentrated around a particular phase (Fig. 4.5D, compare with Fig. 4.2A–B).

We here assume that all the spike trains shown in Fig. 4.5D converge onto one target neuron and that every single spike gives a synaptic current modeled as an alpha-function (Fig. 4.5F)

$$I_{\mathrm{syn}}(t) = \frac{Ht}{\tau} \exp\left(1 - \frac{t}{\tau}\right).$$

Setting the time constant $\tau = 0.3$ msec and the peak current of single synaptic input $H = 1$ pA, we can calculate the total synaptic input into the target neuron (Fig. 4.5E). Since the spikes are phase-locked, the simulated synaptic input oscillates at the signal frequency. This is confirmed by the Fourier transform. Power spectrum density of the synaptic input has a main peak at the signal frequency of 1 kHz and smaller peaks at higher harmonics (Fig. 4.5G). Thus the periodic signal can be "reconstructed" with accumulation of periodic spike sequences (Ashida et al. 2007).

In this section we have discussed how to model phase-locked spike trains. In the real nervous system, temporal information coded by phase-locked spikes would be processed in higher-order centers in the brain and finally lead to behavioral actions. Since the motor output generally operates on the mean activity rate of the motor efferents, temporal codes should be "decoded" or "translated" into the rate code somewhere in the brain. This translator is expected to act as a coincidence detector that senses temporal structure of the input and modulates spiking activity. Thus it is important to examine how and to what extent a coincidence detector can extract

information from oscillatory inputs. It has been proposed that membrane time constant, spiking threshold, and the number of inputs per period are key parameters that determine the performance of a coincidence detector (Kempter et al. 1998).

Acknowledgements We acknowledge advice and assistance from Krishna Prasad Rajaram, Philip Joris, and Christine Köppl. This work was supported by NIH DC00436 to CEC, by NIH P30 DC04664 to the University of Maryland Center for the Evolutionary Biology of Hearing, by the German Research Foundation (DFG, Wa-606/12, Ke-788/1-3, 4), and by the Bundesministerium für Bildung und Forschung (BMBF, Bernstein Collaboration Temporal Precision, 01GQ07101 to HW).

References

Ashida G, Abe K, Funabiki K, Konishi M (2007) Passive soma facilitates submillisecond coincidence detection in the owl's auditory system. J Neurophysiol 97:2267–2282

Batschelet E (1981) Circular statistics in biology. Academic Press, London

Carr CE (1993) Processing of temporal information in the brain. Annu Rev Neurosci 16:223–243

Carr CE, Heiligenberg W, Rose GJ (1986) A time-comparison circuit in the electric fish midbrain. I. Behavior and physiology. J Neurosci 6:107–119

Carr CE, Soares D (2002) Evolutionary convergence and shared computational principles in the auditory system. Brain Behav Evol 59:294–311

Carr CE, Soares D, Parameshwaran S, Perney T (2001) Evolution and development of time coding systems. Curr Opin Neurobiol 11:727–733

Dynes SB, Delgutte B (1992) Phase-locking of auditory-nerve discharges to sinusoidal electric stimulation of the cochlea. Hear Res 58:79–90

Engel AK, König P, Gray CM, Singer W (1990) Stimulus-dependent neuronal oscillations in cat visual cortex: inter-columnar interaction as determined by cross-correlation analysis. Eur J Neurosci 2:588–606

Fisher NI (1993) Statistical analysis of circular data. Cambridge University Press, Cambridge

Gerstner W, Kempter R, van Hemmen JL, Wagner H (1996) A neuronal learning rule for submillisecond temporal coding. Nature 383:76–78

Gleich O, Narins PM (1988) The phase response of primary auditory afferents in a songbird (*Sturnus vulgaris L.*). Hear Res 32:81–92

Goldberg JM, Brown PB (1969) Response of binaural neurons of dog superior olivary complex to dichotic tonal stimuli: some physiological mechanisms of sound localization. J Neurophysiol 32:613–636

Greenwood JA, Durand D (1955) The distribution of length and components of the sum of *n* random unit vectors. Ann Math Statist 26:233–246

Hartmann G (1987) Recognition of hierarchically encoded images by technical and biological systems. Biol Cybern 57:73–84

Hill KG, Stange G, Mo J (1989) Temporal synchronization in the primary auditory response in the pigeon. Hear Res 39:63–74

Johnson DH (1980) The relationship between spike rate and synchrony in responses of auditory-nerve fibers to single tones. J Acoust Soc Am 68:1115–1122

Joris PX, Louage DH, Cardoen L, van der Heijden M (2006) Correlation index: a new metric to quantify temporal coding. Hear Res 216–217:19–30

Joris PX, Smith PH (2008) The volley theory and the spherical cell puzzle. Neuroscience 154:65–76

Kawasaki M, Guo YX (1996) Neuronal circuitry for comparison of timing in the electrosensory lateral line lobe of the African wave-type electric fish *Gymnarchus niloticus*. J Neurosci 16:380–391

Kempter R, Gerstner W, van Hemmen JL, Wagner H (1998) Extracting oscillations. Neuronal coincidence detection with noisy periodic spike input. Neural Comput 10:1987–2017

Kiang NYS, Watanabe T, Thomas EC, Clark EF (1966) Discharge patterns of single fibers in the cat's auditory nerve. MIT Press, Cambridge

Kidd RC, Weiss TF (1990) Mechanisms that degrade timing information in the cochlea. Hear Res 49:181–208

Köppl C (1997) Phase locking to high frequencies in the auditory nerve and cochlear nucleus magnocellularis of the barn owl, Tyto alba. J Neurosci 17:3312–3321

Liu D, Peddada S, Li L, Weinberg C (2006) Phase analysis of circadian-related genes in two tissues. BMC Bioinformatics 7:87

Louage DH, van der Heijden M, Joris PX (2005) Enhanced temporal response properties of anteroventral cochlear nucleus neurons to broadband noise. J Neurosci 25:1560–1570

Manley GA, Köppl C, Yates GK (1997) Activity of primary auditory neurons in the cochlear ganglion of the emu *dromaius novaehollandiae*: spontaneous discharge, frequency tuning, and phase locking. J Acoust Soc Am 101:1560–1573

Mardia KV (1972) Probability and mathematical statistics: statistics of directional data. Academic Press, London

Mauk M, Buonomano D (2004) The neural basis of temporal processing. Annu Rev Neurosci 27:307–340

Mouritsen H, Ritz T (2005) Magnetoreception and its use in bird navigation. Curr Opin Neurobiol 15:406–414

Palmer AR, Russell IJ (1986) Phase-locking in the cochlear nerve of the guinea-pig and its relation to the receptor potential of inner hair-cells. Hear Res 24:1–15

Paolini AG, FitzGerald JV, Burkitt AN, Clark GM (2001) Temporal processing from the auditory nerve to the medial nucleus of the trapezoid body in the rat. Hear Res 159:101–116

Rayleigh JWS (1894) Theory of sound, vol 2, 2nd edn. Macmillan, London, pp 230–223

Rybak I, Stecina K, Shevtsova N, McCrea D (2006) Modelling spinal circuitry involved in locomotor pattern generation: insights from the effects of afferent stimulation. J Physiol 577:641–658

Sachs MB, Sinnott JM (1978) Responses to tones of single cells in nucleus magnocellularis and nucleus angularis of the redwing blackbird (*Agelaius phoeniceus*). J Comp Physiol 126:347–361

Salvi RJ, Saunders SS, Powers NL, Boettcher FA (1992) Discharge patterns of cochlear ganglion neurons in the chicken. J Comp Physiol 170:227–241

Sullivan WE, Konishi M (1984) Segregation of stimulus phase and intensity coding in the cochlear nucleus of the barn owl. J Neurosci 4:1787–1799

Trussell LO (2008) Central synapses that preserve auditory timing. In: Oertel D (ed) The senses: a comprehensive reference. Elsevier, New York, pp 587–602.

Weiss TF, Rose C (1988) A comparison of synchronization filters in different auditory receptor organs. Hear Res 33:175–180

Wever E, Bray C (1930) The nature of acoustic response: the relation between sound frequency and frequency of impulses in the auditory nerve. J Exp Psychol 13:373–387

Part II
Pairwise Comparison of Spike Trains

Chapter 5
Pair-Correlation in the Time and Frequency Domain

Jos J. Eggermont

Abstract Neural pair-correlation can be analyzed and represented in both the time and frequency domains. Sometimes it is easier to see the effects in the time domain correlograms, sometimes the frequency representation in the form of the coherence function gives more insight, for instance, about which frequency regions contribute to the correlation. Regardless the preference one might have of the domain for representing the interactions, calculations are generally easier and faster when performed in the frequency domain.

5.1 Dual Worlds: Time and Frequency Domain Representations are Related by Fourier Transforms

Any time signal $x(t)$, be it a densely sampled train of action potentials or an abstracted version in the form of unit pulses (spikes), has an equivalent description in the frequency domain, $S_x(f)$. This spectrum can be obtained by Fourier transformation of the spike train $x(t)$,

$$S_x(f) = \int_0^T x(t).e^{-i2\pi ft}\,dt \quad \text{or, in sampled form,}$$

$$S_x(f) = \sum_{n=1}^{N} x(n)e^{-i2\pi fn}, \tag{5.1}$$

where $n \in [1, N]$ indicates the bin number. The spectrum is complex valued, i.e., consists of magnitude and phase spectra.

The autocorrelation function $R_{xx}(\tau)$ of $x(t)$ is defined as the average lagged cross-product of the signal with itself over signal duration T,

J.J. Eggermont (✉)
Department of Physiology and Pharmacology, Department of Psychology, University of Calgary, Calgary, Alberta, Canada
e-mail: eggermon@ucalgary.ca

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_5, © Springer Science+Business Media, LLC 2010

$$R_{xx}(\tau) = \frac{1}{T}\int_0^T x(t)x(t+\tau)\,dt \quad \text{or, in sampled form,}$$

$$R_{xx}(k) = \frac{1}{K}\sum_{n=1}^{K} x(n)x(n+k). \tag{5.2}$$

The autocorrelation function reflects the self-similarity of the signal as a function of the delay, expressed as a time delay τ or as the number of bins k. We only have to delay in one direction because we are dealing here with only one signal, and therefore the autocorrelation function is symmetric around zero lag time. For a periodic signal, the autocorrelation function will indicate a complete self-similarity after each repetition period. For all signals, the value of $R_{xx}(\tau)$ is never larger than its value for $\tau = 0$, i.e., the correlation of the signal with itself. In practice the autocorrelation function is obtained in the form of a correlation histogram. A given bin size is selected, small enough so that there are no two spikes in the same bin, and coincidences between the original spike train and the time-shifted one are counted in each bin k starting anew from every spike time n.

The power spectrum $S_{xx}(f) = S_x(f)S_x^*(f)$. $S_x^*(f)$ is the complex conjugate of $S_x(f)$,

$$S_x^*(f) = \sum_{n=1}^{N} x(n)e^{i2\pi fn}.$$

The power spectrum of the signal $x(t)$ also results from the Fourier transform of $R_{xx}(\tau)$:

$$S_{xx}(f) = \frac{1}{T}\int_0^T R_{xx}(\tau).e^{-i2\pi f\tau}\,d\tau \quad \text{or, in sampled form,}$$

$$S_{xx}(f) = \frac{1}{K}\sum_{k=0}^{K-1} R_{xx}(k)e^{-i2\pi fk}. \tag{5.3}$$

The information present in the autocorrelation function and the power spectrum is complementary. In theory the Wiener–Khintchine theorem expressed in $S_{xx}(f) = \frac{1}{T}\int_0^T R_{xx}(\tau).e^{-i2\pi f\tau}\,d\tau$ is only valid when the integration boundaries extend from $-\infty$ to ∞, i.e., $S_{xx}(f) = \frac{1}{T}\int_{-\infty}^{\infty} R_{xx}(\tau).e^{-i2\pi f\tau}\,d\tau$.

5.1.1 Cross-Correlation and Cross-Spectrum: FFT Pairs

When we have two different spike trains, their correspondence in firing times is given by the cross-correlation function

$$R_{xy}(\tau) = \frac{1}{T}\int_0^T x(t)y(t+\tau)\,dt \quad \text{or, in sampled form,}$$

$$R_{xy}(k) = \frac{1}{N}\sum_{n=1}^{N} x(n)y(n+k). \tag{5.4}$$

Note that here we take both lag and lead times; in practice, $\tau \in [-0.5T, 0.5T]$ and $k \in [-0.5N, 0.5N]$, respectively, to calculate the cross-correlation function, since there is no reason to assume that there is symmetry between the two signals or spike trains as there was in case of the autocorrelation.

The Fourier transform of the cross-correlation function is called the cross-spectrum:

$$S_{xy}(f) = \frac{1}{T}\int_0^T R_{xy}(\tau).e^{-i2\pi f\tau}\,d\tau \quad \text{or, in sampled form,}$$

$$S_{xy}(f) = \frac{1}{K}\sum_{k=0}^{K-1} R_{xy}(k)e^{-i2\pi fk}. \tag{5.5}$$

The cross-spectrum is complex, because $R_{xy}(\tau)$ is not an even function, i.e., it is not symmetric around $\tau = 0$.

5.1.2 Practical Issues in Estimating Correlations and Spectra

In practice, correlation functions and coherence functions are not estimated from $[-\infty\ \infty]$ for lag-time and frequency, respectively, but a window $w(n)$ is typically imposed. Ponomarenko et al. (2004) have shown that when the upper integration boundary T is $\gg$ the duration of the window filter, one can in fact extend it to ∞. By changing the original integration window from $[0\ T]$ to $[-0.5T\ 0.5T]$, the integral boundaries can thus be extended to $[-\infty, \infty]$. However, the autocorrelation function in (5.2) has to be computed for windowed signals as well.

It is noted that all correlation functions and spectra used here and in the following are considered as averaged values ($y = x$ for autocorrelations and power spectra) and including window functions (omitted here):

$$R_{xy}(\tau) = E\left[\frac{1}{T}\int_{-0.5T}^{0.5T} x(t)y(t+\tau)\,dt\right] \quad \text{and} \quad S_{xy}(f) = E\left[S_x(f)S_y^*(f)\right].$$

Generally, the averaging is over consecutive time segments of length T, but it could also be ensemble averaging over limited time segments following a repeated stimulus.

5.1.3 Impulse Response and Cross-Correlation

If one considers the spike train $x(t)$ as the input to a system and spike train $y(t)$ as the output of that system, then the frequency response function of that system is given by $H(f) = \frac{S_{xy}(f)}{S_{xx}(f)}$, and its inverse Fourier transform is, in the case of a linear system, equal to the impulse response $h(\tau)$. In a sense, $h(\tau)$ is the cross-correlation function $R_{xy}(\tau)$ normalized on the input. Note that since $h(\tau)$ captures only the linear aspects of the system, the same holds for $R_{xy}(\tau)$ and $S_{xy}(f)$.

5.1.4 Correlation and Coherence: Symmetry Breaking Through Normalization

In the form we have presented the auto- and cross-correlation functions, there was no normalization, so the values increase when the number of spikes in the spike trains increases. For an analog signal, the signal power is also known as the variance σ_x^2, defined as $\langle (x(t) - \mu_x)\rangle^2$ with μ_x the mean of $x(t)$, and a standard normalization for the autocorrelation function is to subtract the mean signal value and divide by the variance to obtain the autocorrelation coefficient function:

$$\rho_{xx}(\tau) = \left(\frac{1}{2T} \int_{-T}^{T} \big(x(t) - \mu_x\big)\big(x(t+\tau) - \mu_x\big)\, dt \right) \Big/ \sigma_x^2. \tag{5.6}$$

This is equal to the autocovariance function divided by the variance. One obtains the cross-correlation coefficient function in the same way:

$$\rho_{xy}(\tau) = \left(\frac{1}{2T} \int_{-T}^{T} \big(x(t) - \mu_x\big)\big(y(t+\tau) - \mu_y\big)\, dt \right) \Big/ \sqrt{\sigma_x^2 \sigma_y^2}. \tag{5.7}$$

In this case one obtains the cross-covariance divided by the square root of the product of the variances. Similarly, a normalization is carried out in the frequency domain to give the complex coherence function (or coherency), excluding frequencies for which the denominator equals 0:

$$\gamma_{xy}(f) = \frac{S_{xy}(f)}{\sqrt{S_{xx}(f) S_{yy}(f)}}. \tag{5.8}$$

However, the Fourier transform of $\rho_{xy}(\tau)$ is bound to be different from $\gamma_{xy}(f)$ since the Fourier transform of $\rho_{xy}(\tau) = \frac{R_{xy}(\tau) - \mu_x \mu_y}{\sqrt{\sigma_x^2 \sigma_y^2}}$ is equal to $\frac{S_{xy}(f) - \mu_x \mu_y \delta f}{\sqrt{\int S_{xx}(f)\, df \int S_{yy}(f)\, df}}$. Thus the symmetry between time and frequency domains breaks down after normalization.

Quite often squared correlation coefficients are used that indicate the percentage of variance in $y(t)$ predicted in a linear way from knowing $x(t)$. Analogously, the squared coherency (commonly known as coherence) is defined as

$$\gamma_{xy}^2(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f) S_{yy}(f)} \tag{5.9a}$$

and indicates the linearly explained power in the spectrum of $y(t)$ at each frequency f on the basis of knowing the spectrum of signal $x(t)$ at each frequency f. Note that because of the linearity, frequencies $f_2 \neq f_1$ in $x(t)$ have no effect on frequency f_1 in $y(t)$. The squared coherence can also be expressed as the product of the forward and backward transfer functions:

$$\gamma_{xy}^2(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f) S_{yy}(f)} = \left| \frac{S_{xy}(f)}{S_{xx}(f)} \right| \left| \frac{S_{xy}(f)}{S_{yy}(f)} \right| = \big|H_{xy}(f)\big| \big|H_{yx}(f)\big|. \tag{5.9b}$$

Note that for the practical calculation of this function, one has to use average power- and cross-spectra; otherwise the function always turns out to be equal to 1. Again,

the correlation coefficients and coherence reflect only linear relationships between the two spike trains. Any deviation from a value of 1 for both the squared correlation coefficient and the squared coherence is due to (1) nonlinearity of the system, (2) to the presence of noise in the system, and (3) to the presence of other inputs to the system that contribute to $y(t)$ but are not accounted for by $x(t)$.

In a simple example of a system representing a static nonlinearity such as $y = x^2$, and given a sinusoidal $x(t)$ with frequency f, the output of the system $y(t)$ will be a raised cosine with the double frequency $2f$. The cross-spectrum between $x(t)$ and $y(t)$ will be zero, and so will be the coherence. The same holds, of course, for the cross-correlation function because the product of a sine and a cosine squared is zero on average. A general practical reference for the topics reviewed above is Van Drongelen (2007) and Chap. 20 of this book.

5.2 Practical Pair-Correlation Estimation for Neural Spike Trains Under Spontaneous Firing Conditions

5.2.1 Representation of the Cross-Correlogram in Terms of Coincidences

In case of Poisson spike trains, the variance is equal to the number of spikes in the train (see also Chap. 6 by Tetzlaff and Diesmann), and the cross-correlation coefficient becomes (Eggermont 1992a)

$$\rho_{xy}(\tau) = \left[R_{xy}(\tau) - \frac{N_x N_y}{N} \right] \Bigg/ \sqrt{\left(N_x - \frac{N_x^2}{N} \right) \left(N_y - \frac{N_y^2}{N} \right)}. \tag{5.10}$$

Here, N_x and N_y are the number of spikes in spike trains $x(t)$ and $y(t)$, respectively, and N is the number of bins in the total spike train duration. $\frac{N_x N_y}{N}$ is the expected number of coincidences in case of independence. For small numbers of spikes compared to the number of bins, i.e., low firing rates, in the record (N) this can be approximated by

$$\rho_{xy}(\tau) = \left[R_{xy}(\tau) - \frac{N_x N_y}{N} \right] \Big/ \sqrt{(N_x N_y)}. \tag{5.11}$$

For statistical purposes, this can also be converted (holds for all firing rates) to a z-score, i.e., the number of standard deviations above the mean-corrected R, which is equal to zero:

$$z_{xy}(\tau) = \left[R_{xy}(\tau) - \frac{N_x N_y}{N} \right] \Bigg/ \sqrt{\frac{N_x N_y}{N}}. \tag{5.12}$$

Statistical significance should be set conservatively, e.g., at $z \geq 4$.

The standard normal distribution that is the basis for the interpretation of z, as the number of standard deviations between a value x and the mean μ of the distribution,

is based on the z-score $z = (x - \mu)/\sigma$. For our cross-correlograms, we do not have normal (i.e., Gaussian) distributions, but we assume that the bin filling with coincidences follows a Poisson process (so bin size should be so small that the probability of >1 coincidence is vanishingly small). For a Poisson process (see Chap. 1), the standard deviation is the square root of the mean, leading to the standard deviation of the number of coincidences $\sigma_{xy} = \sqrt{\frac{N_x N_y}{N}}$.

If one wants to assess significance based on the cross-correlation coefficient, the standard deviation thereof is equal to $\mathrm{SD}(\rho_{xy}) = \sqrt{\frac{1}{N}(1 - \frac{N_x N_y}{N^2})}$. For large numbers of bins in the record and relatively low firing rates, this can be approximated as $\mathrm{SD}(\rho_{xy}) = \sqrt{\frac{1}{N}}$. So for a 900-s record length and 2-ms bins, the SD $= 0.0015$. So taking 4 SD would make ρ_{xy} values above 0.006 significant. For larger numbers of spikes, the SD decreases compared to this estimate.

5.2.2 Representations of the Cross-Correlogram in Terms of Firing Rate

$R_{xy}(\tau)$ is sometimes defined as a first-order conditional rate function $R_{x|y}(\tau) = R_{xy}(\tau)/\mu_y$ that measures, for one cell y and close to any particular time t, the average instantaneous rate or the likelihood of generating a spike, conditional on an x spike τ time units away (Brillinger et al. 1976). If the trains are independent, then $R_{x|y}(\tau) = \mu_y$, where μ_y is the mean firing rate for spike train y, for all τ. If y spikes are independent of "later" x spikes (causality), then $R_{x|y}(\tau) = \mu_y$ for all $\tau < 0$. This "causality" may apparently be violated if both x and y spikes are caused with some jitter by a third, unobserved, input. Deviation of $R_{x|y}(\tau)$ from μ_y is suggestive of dependency of the y-train on what happened in the x-train τ time units earlier. For all natural spike trains, $R_{x|y}(\tau)$ will be flat and essentially equal to μ_y at large $|\tau|$ values because inevitably any influence of x on y will have vanished. Assuming a Poisson distribution, one can plot $\sqrt{R_{x|y}(\tau)}$. The variance of $\sqrt{R_{x|y}(\tau)}$ is approximately constant for all τ at a level of $(4\Delta T \mu_x)^{-1}$, so SD lines can be drawn at $\pm(4\Delta T \mu_x)^{-2}$ and become smaller in proportion to $\sqrt{T}$. This provides a good approximation if Δ is not too large, and the correlation width of the process is not too large (Voigt and Young 1990).

5.3 Testing for Stationarity of Individual Spike Trains

5.3.1 Time-Dependent Cross-Correlation Functions

Visual inspection of the time-dependent cross-correlation function $R_{xy}(t, \tau)$, with t the running time in the spike trains and τ as the lag-lead time between spikes

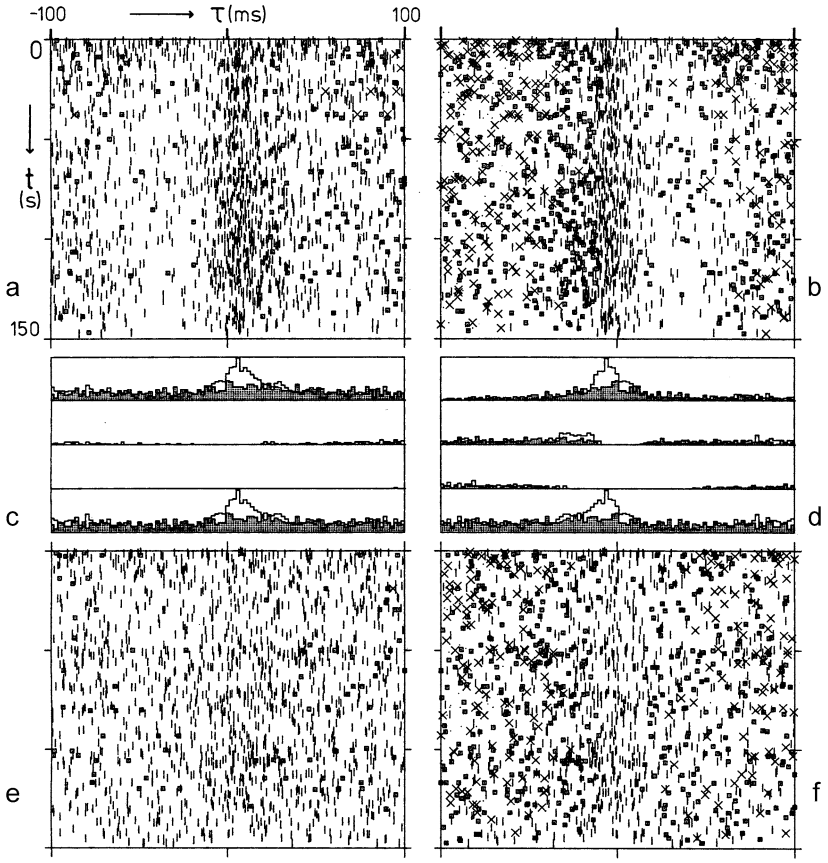

Fig. 5.1 Time-dependent cross-correlation diagrams of spike trains. The different symbols used indicate the order of the spikes following the reference spike, "|" indicates the first-order recurrence time, i.e., the first spike in the y-train following or preceding a spike in the x-train (for the *left-hand column*), "■" indicates the second-order recurrence time, i.e., the second spike in the y-train following or preceding a spike in the x-train. Finally "×" indicates the third-order recurrence time (Perkel et al. 1967). **a** Simultaneous time-dependent cross-correlation diagram, spikes from unit x (2188 events) are used as triggers; **b** as **a**, but now with spikes from unit y (528 events) as triggers; **c** simultaneous and nonsimultaneous (*shaded*) recurrence-time histograms of order 1, 2, and 3, and cross-correlation histogram derived from **a** and **e**, respectively; **d** as **c**, but now derived from **b** and **f**; **e** nonsimultaneous time-dependent cross-correlation diagram, spikes from unit x are used as triggers; **f** as **e**, but now with spikes from unit y as triggers. From Van Stokkum et al. (1986)

in trains x and y, shown here in an example (Fig. 5.1) taken from Van Stokkum et al. (1986), is often the best first step in judging stationarity. The example reflects simultaneous recordings from two neurons in the auditory midbrain of the grassfrog in response to species-specific vocalizations. The left-hand and right-hand columns refer to the same data, but on the left-hand side the correlation shown is $R_{xy}(t, \tau)$, whereas the right-hand side shows $R_{yx}(t, \tau)$, i.e., the reference unit has changed.

Vertically, the time over the first 150 seconds since stimulus onset, t, is shown, and, horizontally, the lag-lead times, τ, between the x and y spikes. Different symbols indicate different order recurrence times (see Chaps. 1 and 3 in this book). One observes that, with the exception of the first 10 seconds or so, the occurrence of the recurrence times is stationary. The most important events that determine the various order cross-correlograms here (shown in parts c, d) are the first-order recurrences, i.e., there are hardly any bursts in the y-train and only some in the x-train, which had the higher firing rate. The top correlogram is that based on only first-order recurrence times, the second one on second-order, and the third one on third-order recurrence times, and the fourth shows the all-order cross-correlogram as commonly used. All correlograms are on the same scale. The shaded correlograms are the shift-predictors (see next section), calculated from the time-dependent shift correlograms shown in the bottom row that show only modest stimulus locking to the stimulus. The shape of the cross-correlogram suggests common input as the main source of correlated spike times (see later in this chapter).

5.3.2 Stationarity Tests for Spike Trains

Several types of nonstationarity have been identified in spike trains, which can lead to misleading results; an important one being firing rate (FR) changes throughout the duration of the experiment (Grün et al. 2002; Chap. 10). Nonstationarities in firing rate generally have two causes: they are either preparation-related ones that are caused, e.g., by changes in anesthesia level, or stimulus-related ones that cause modulations in FR (Averbeck and Lee 2004). The latter will be discussed under stimulus correction procedures. Most signal analysis methods require that the data are at least weakly stationary for the FR (Brown et al. 2004). The concept of weak stationarity in neural spike trains is defined as a stability of average firing rate and variance of FR over time. Nonstationarity in FR leads to biased results in numerous cross-correlation techniques (Perkel et al. 1967) and may sometimes even be the only cause of correlation between spike trains (Brody 1999). In Chap. 6 by Tetzlaff and Diesmann, the inhomogeneous Poisson source is discussed as an example of nonstationarity. Gourévitch and Eggermont (2007) recently reviewed suitable tests for stationarity and also proposed a new stationarity test for use with spike trains. The reader is referred to that paper for details.

5.4 Pair-Correlation Under Stimulus Conditions

A peak in the cross-correlogram can be the result of a neural interaction between units ("noise correlation") and also the result of a coupling of the firings of the neurons to a stimulus ("signal correlation"). As we will see, the interpretation of the cross-correlogram will require both the autocorrelograms of the two units and the correlation due to stimulus coupling. Following our earlier approach (Eggermont et al. 1983), we consider two spike trains $x(t)$ and $y(t)$ of duration T, represented as a sequence of δ-functions, see Chap. 1:

Table 5.1 Overview of possible correlation histograms for a double unit recording with two presentations of the stimulus ensemble

	$x_1(t)$	$y_1(t)$	$x_2(t)$	$y_2(t)$
$x_1(t)$	$C_{x_1x_1}$	$C_{x_1y_1}$	$C_{x_1x_2}$	$C_{x_1y_2}$
$y_1(t)$		$C_{y_1y_1}$	$C_{x_2y_1}$	$C_{y_1y_2}$
$x_2(t)$			$C_{x_2x_2}$	$C_{x_2y_2}$
$y_2(t)$				$C_{y_2y_2}$

$$x(t) = \sum_{i=1}^{N_k} \delta(t - t_i) \quad \text{and} \quad y(t) = \sum_{j=1}^{M} \delta(t - t_j) \tag{5.13}$$

with cross-correlation function

$$R_{xy}(\tau) = \frac{1}{T} \sum_{i-1}^{N} \sum_{j=1}^{M} \delta\big[\tau - (t_i - t_j)\big]. \tag{5.14}$$

For a bin width Δ, the cross-correlation histogram

$$C_{xy}(\tau, \Delta) = \frac{1}{\Delta} \int_{\tau-\Delta/2}^{\tau+\Delta/2} R_{xy}(\sigma)\, d\sigma \tag{5.15}$$

can be defined. If one presents the stimulus ensemble twice, then in case of a double unit recording, the resulting spike trains are $x_1(t)$ and $y_1(t)$ for the first stimulus presentation and, respectively, $x_2(t)$ and $y_2(t)$ for the second one. For those four spike trains, one can compute ten correlation histograms (or correlograms) as shown in Table 5.1.

On the main diagonal two estimates each of the units' autocorrelation histograms are found. Under long-term stationary firing conditions, $C_{x_1x_1} = C_{x_2x_2}$, and one commonly uses $C_{xx} = (C_{x_1x_1} + C_{x_2x_2})/2$ as an estimator of the autocorrelation function for spike train $x(t)$.

For the off-diagonal elements, it is noted that, e.g., $C_{x_1y_1}(\tau) = C_{y_1x_1}(-\tau)$; therefore, the table is symmetric, provided that the sign of τ is adjusted. We define $C_{x_1y_1}(\tau)$ and $C_{x_2y_2}(\tau)$ as simultaneous cross-coincidence histograms, and again we may use their average as an estimate for the cross-correlation function. Following our previously introduced terminology (Eggermont et al. 1983; Epping and Eggermont 1987), cross-correlations obtained under stimulus conditions will be called *neural synchrony*, and they will be a mix of signal- and noise-correlations (e.g., Gawne and Richmond 1993). Cross-correlations obtained under spontaneous firing conditions or after appropriate correction for correlation resulting from stimulus locking will be called *neural correlation*; they exclusively are noise correlations. During stimulation, the "total" neural synchrony consists of a component due to stimulus synchrony and one that is a result of neural connectivity, the neural correlation.

The nonsimultaneous cross-correlation histograms between spike trains $x_1(t)$ and $y_2(t)$ and, respectively, $x_2(t)$ and $y_1(t)$ are considered to represent correlation

between spike trains $x(t)$ and $y(t)$ as a result of stimulus coupling. These stimulus-induced correlations are known as the shuffle- or shift-predictors (see also Chap. 17).

Another nonsimultaneous cross-correlation histogram is obtained by taking both spike trains for the same unit to obtain $C_{x_1x_2}$ and $C_{y_1y_2}$. These histograms represent the amount of stimulus coupling of the units individually and have been called the "existence functions" (Aertsen et al. 1979) because any significant peak around $\tau = 0$ indicates a stimulus coupling effect and hence the existence of a stimulus response relation. This shuffled autocorrelogram technique was rediscovered by Joris et al. (2006).

There is no model-free approach for the separation of correlations produced by direct neural interaction and those caused by stimulus coupling. Under the assumption that the effects of an external stimulus and the effects of neural connectivity to the neural synchrony are additive, Perkel et al. (1967) proposed a correction for the effects of the stimulus. Two formally identical stimulus predictors were suggested; one resulting from a cross-correlation of the two single-unit post-stimulus-time histograms (PSTHs) and the other from calculating the cross-correlogram between one spike train and a shifted (by 1 stimulus period) version of the second spike train. This latter procedure, resulting in the shift predictor, has been the most popular and has the advantage that it can be applied to very long stimuli (such as a set of animal vocalizations) that allow only one or a few repetitions; in such a case calculating the PSTH would be useless. The shift predictor was assumed to represent the amount of correlation that could be attributed to the locking of the neuron's firings to the stimulus. Subtracting the shift predictor from the neural synchrony then would result in the neural correlation.

The assumption of additivity of stimulus-induced correlation and neural correlation may well be the weakest link in the chain of reasoning that leads to an interpretation of stimulus-dependent neural correlations. A strong warning against the use of the shift predictor in estimates of the neural interaction strength came from studies using realistic model neurons (Melssen and Epping 1987). Because stimulus effects and neural connectivity effects on the neural synchrony are generally nonadditive, it was concluded that the use of the shift predictor to separate stimulus and neural effects was of limited value. Yang and Shamma (1990) underlined these objections to the use of correlation methods in identifying neural connectivity and arrived at conclusions similar to those of Melssen and Epping (1987) using simulations with similar model neurons. In case of strong stimulus-locking, even all synchrony can be due to stimulus-correlation (Eggermont 1994). Under "adequate" stimulus conditions that evoke strongly stimulus-driven and reproducible spike trains, especially for auditory stimuli, the neural synchrony becomes identical to the expected correlation based on the shift predictor, and consequently no inference can be made about possible underlying connectivity (changes). One may avoid this overcorrection effect due to stimulus locking by applying suboptimal stimulation. Thus a sufficient amount of "neural noise" in the record is required to allow conclusions about changes in functional neural connectivity.

Thus, when stimulus-dependent neural correlations are obtained, there are a number of possibilities: the interneuronal coupling itself changes due to application of

a stimulus (facilitation, depression, short-term learning), the finding is an artifact due to the selection of the wrong model, or the finding can be explained on the basis of a pool of neurons affecting the firing behavior of the pair under study in a stimulus-dependent way.

5.4.1 Practical Estimation of Stimulus Correlation

During spontaneous activity, one estimates the expected value of the cross-correlogram under the assumption of independence and subtracts this in order to obtain the neural correlation. During stimulation, the effect of a common input to both spike trains as a result of a correlation of the neuronal firing to the stimulus has to be taken into account if one wants an estimate of the true neural correlation. There are two ways to do this. The first can be used whenever the stimulus is truly periodic. One constructs the post-stimulus-time histograms, i.e., the cross-correlograms between stimulus onsets and spikes (i.e., the post-stimulus-time histograms), $R_{S_x}(\sigma)$ and $R_{S_y}(\sigma)$, and calculates their correlation integral modulo stimulus period P (here σ is the post-stimulus time):

$$R_{\mathrm{Scorr}}(\tau) = \int_0^P R_{S_x}(\sigma) R_{S_y}(\sigma + \tau)\, d\sigma. \tag{5.16}$$

One can also use the shift predictor obtained by calculating the cross-correlation between spike trains $x(t)$ and $y(t + P)$, i.e., between the original x-train and the y-train shifted over one stimulus period. This latter method is the only one that can be used if a long stimulus sequence, e.g., a sequence of vocalizations or other natural sounds, is only repeated once, as discussed above.

As an example of this latter stimulus correction procedure, we show recordings from the auditory midbrain to repeated one-second long 30-Hz amplitude-modulated noise-burst stimulation (Fig. 5.2). The shift predictor is shaded and incorporates the 30-Hz modulation in the firing rate that obviously is stimulus locked. The triangular shape of the predictor is the result of using an approximately rectangular envelope for the 500-ms-long noise burst. The neural correlation consists of a narrow zero-lag centered peak approximately 5 ms in width (W) at half peak amplitude (Fig. 5.2B, see Fig. 5.2C for detail).

Another example is based on recordings from the primary auditory cortex of the cat. Here the stimulus consisted of a 450-s-long tone pip ensemble with random presentation in frequency and time (Valentine and Eggermont 2004; Fig. 5.3, details in the legend) that was repeated once, thus resulting in an overall duration of 15 minutes. The functions shown are the cross-correlation coefficient function according to (5.10) as a solid line, the shift predictor (dashed line), and the stimulus-corrected neural correlation (the beaded solid line). One observes a very modest stimulus correlation (shift predictor) because these long continuous stimuli produce perstimulatory adaptation that reduces the degree of stimulus locking.

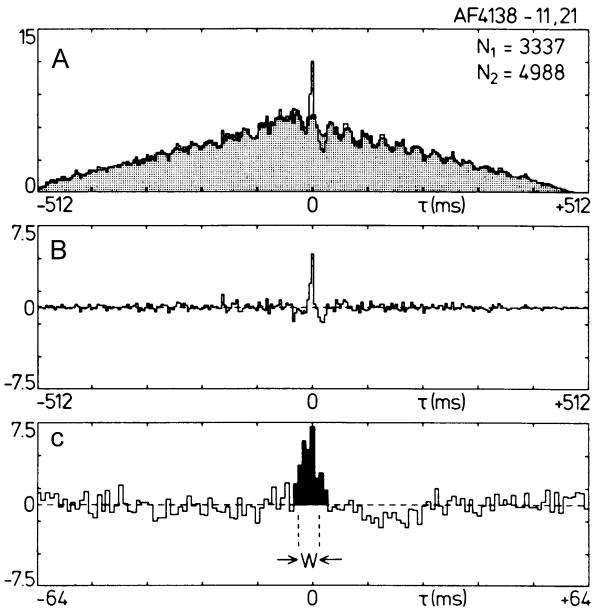

Fig. 5.2 **A**: Simultaneous and nonsimultaneous (i.e., the shift-predictor; *shaded*) cross-correlation histograms for two units recorded from the auditory midbrain in the grassfrog under amplitude-modulated noise burst stimulation of 500-ms duration. The triangular shift-predictor results from the approximately rectangular stimulus envelope. A 30-Hz stimulus-locked oscillation is visible as well. **B** and **C**: Difference histogram shown on different time scales; *W* is the width of the peak at half amplitude. Bin width was 4 ms in **A** and **B**, and 1 ms in **C**. The number of spikes for each unit is indicated in the *top right* of panel **A**. From Epping and Eggermont (1987)

5.4.2 Rate Correlation and Event Correlation

Cross-correlograms not infrequently show a narrow peak on a broader pedestal (e.g., Figs. 5.2A and 5.4E), especially when units are recorded on the same electrode (Eggermont 1992a, 1992b; Nowak et al. 1995). The broader pedestal is typically the result of covariance in firing rate between the two spike trains, e.g., caused by the stimulus, whereas the sharp peak indicates event synchrony (Neven and Aertsen 1992). The rate covariance under spontaneous conditions can easily be estimated by calculating a cross-correlogram for 50-ms bins that slide in 10-ms steps (Eggermont and Smith 1995). Subtracting the rate covariance from the overall correlogram will result in the event correlation.

This generally assumed model, even if correct, may result in an inaccurate estimation of the true event correlation. Staude et al. (2008) decomposed the raw cross-correlation as the sum of the modulation rate covariance and spike coordination based on the mean conditional covariance of the processes given the rates. Both, rate covariation and spike coordination alike, contribute to the raw correlation function by triangular peaks. This separation does not always have to result in a broad triangular peak for rate correlation and a narrow one for spike correlation; at least in their procedure the reverse can be possible as well. Staude et al. (2008) posit that "without prior knowledge of the underlying model class, rate estimation will most likely be suboptimal. The resulting predictor leaves a residual central peak in the corrected correlation function, which is likely to be wrongly interpreted as spike coordination".

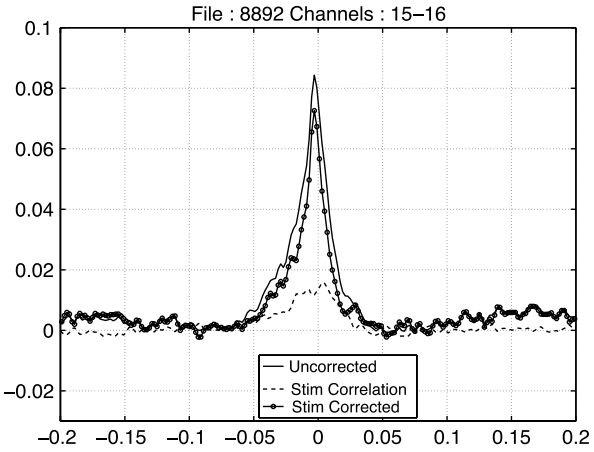

Fig. 5.3 This set of three cross-correlograms illustrates the stimulus-correction procedure applied to spike trains recorded during a 900-s-long random frequency stimulus where the second half was equal to the first. Stimuli were tone pips presented randomly in frequency and time with an overall rate of 28/s across 7 octaves (112 frequencies, so an average rate of 0.25 frequencies/s). The recording comprised two arrays of 16 electrodes each. Here we show the pair-correlation between two neighboring electrodes separated by 0.5 mm. The uncorrected cross-correlation coefficient function is shown in *solid line*, and the peak cross-correlation is approximately 0.082. The frequency tuning of the two recording sites was nearly identical, yet the stimulus correlation obtained as a shift predictor (*dashed line*) shows only modest stimulus locking. The stimulus-corrected cross-correlation coefficient function is shown in a *connected-dot line*

5.4.3 Stimulus-Dependent Neural Correlation

An example, exhibiting stimulus-dependent correlation, is provided in Fig. 5.4. As discussed above, this may potentially result from using the invalid assumption of "neural synchrony equals the sum of stimulus correlation and neural correlation". The two spontaneously active units shown were recorded in the auditory midbrain of the grass frog on electrodes with an estimated tip separation of 130 μm. The spontaneous rates were 0.34 (A) and 2.5 (B) spikes/s, respectively. Their spectro-temporal sensitivities, represented as dot displays of neural activity, as determined with tone pips presented once per second, are shown in Fig. 5.4A, B. The spontaneous activity of unit 1 was suppressed by frequencies below 1,000 Hz (Fig. 5.4A), whereas unit 2 was activated in this frequency range with a latency of 67 ms (Fig. 5.4B). Neural synchrony histograms and shift predictors (shaded) of spontaneous and stimulus evoked activity are shown in Fig. 5.4C–H. In case of spontaneous activity the neural synchrony and shift predictor are flat and hardly different (Fig. 5.4C). For tone pips, the shift predictor (Fig. 5.4D) shows a decrement due to the antagonistic (suppression vs. activation) stimulus influence on the two units. Just showing on top of the shift predictor a small, rather symmetric peak around the origin is visible, likely indicating shared neural input. The shift predictor to sinusoidally amplitude-modulated (AM) tone bursts is broad (Fig. 5.4E) with a sharp ($W = 6$ ms) peak of the neural synchrony visible on top. The cross-correlation histograms to a low-pass

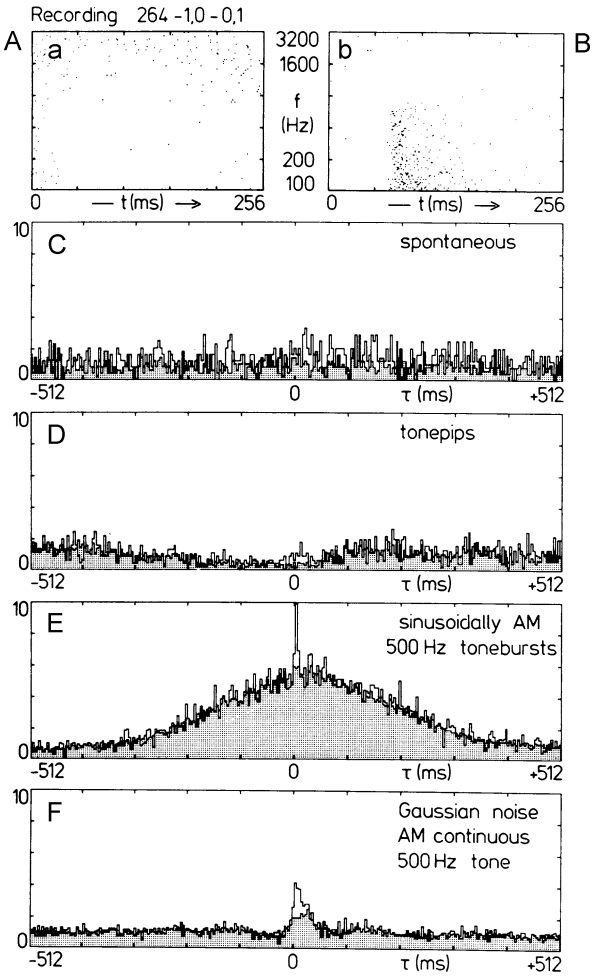

Fig. 5.4 Unit pair with a stimulus-dependent neural correlation. **A** and **B**: *dot* displays of single-unit activity to tone pips with carrier frequencies in the range 100–3,200 Hz. **C**–**F**: Simultaneous and nonsimultaneous (shift predictor; *shaded*) cross-correlation histograms for a variety of stimuli. Bin width is 4 ms. From Epping and Eggermont (1987)

noise AM tone (Fig. 5.4F), with an asymmetric peak of 16-ms half width. In addition to the primary peak flanking valleys are present due to the autocorrelation structure of the individual spike trains (Tetzlaff et al. 2008). The two units responded antagonistically to the tone-pip stimulus, but more alike to the other stimulus ensembles; e.g., when stimulated with sinusoidally AM tonebursts (Fig. 5.4E) with a carrier frequency of 500 Hz, the units both were excited by a band-limited range of AM rates around 100 Hz. So it can be concluded that this unit pair exhibits stimulus dependencies of both stimulus-induced cross-correlations as revealed by the shift predictor as well as on neurally mediated cross-correlations because of changes in form and width of peaks in the neural synchrony as compared with the shift predictor.

5.5 Pair Coherence for Neural Spike Trains

The spectrum of a spike train $x(t)$ of length T with spikes at times x_j is equal to

$$S_x(f) = \sum_{0<x_j\le T} w(n) e^{-i2\pi f x_j}.$$

If a long spike train of length U is divided into L disjoint sections of length T such that $LT = U$, then useful estimates of the power spectra, cross-spectrum, and coherence can be obtained that allow estimating statistical significances:

$$\begin{aligned}\hat{S}_{xy}(f) &= \frac{1}{LT}\sum_{1}^{L} S_x(f) S_y^*(f) \\ &= \frac{1}{LT}\sum_{l=1}^{L}\left(\sum_{(l-1)T<x_j\le lT} e^{-i2\pi f x_j} \sum_{(l-1)T<y_k\le lT} e^{i2\pi f y_k}\right), \qquad (5.17)\end{aligned}$$

$$\hat{\gamma}_{xy}^2(f) = \frac{|\hat{S}_{xy}(f)|^2}{\hat{S}_{xx}(f)\hat{S}_{yy}(f)} \qquad (5.18)$$

where the statistical significance boundary for $\hat{\gamma}_{xy}^2(f)$ is equal to $1-(1-\alpha)^{1/(L-1)}$ at all frequencies $f \neq 0$ (Brillinger et al. 1976). If the desired level of confidence, α, is, for instance, 0.95, then $(1-\alpha)$ gives the desired P-value of 0.05. Inserting the number of disjoint sections over which the averaging was done gives the value of $\hat{\gamma}_{xy}^2(f)$ that needs to be exceeded to get the desired significance level at all frequencies.

Figure 5.5 gives an example for the same recording for which we previously showed the pairwise cross-correlogram and stimulus-correction procedure (Fig. 5.3). An average over 90 segments of 10-s length results in a significance level of $P < 0.05$ to be reached when the squared coherence exceeds 0.03. In contrast, the same level of significant correlation in the time domain is already reached for a level of $\mathrm{SD}(\rho_{xy}) = \sqrt{\frac{1}{N}}$, which for 2 SD and 450,000 bins, would be 0.003, i.e., 10 times smaller as for the coherence. This is likely the result of the correlogram peak reflecting the coherence distributed over all frequencies.

5.5.1 Correcting for Common Input Firing Periodicities and Firing Rates

According to de la Rocha et al. (2007), the stimulus-corrected cross-correlation coefficient for common input depends on the geometric mean of the firing rates of the two neurons in cortical slice recordings and in their simulated data. This effect was also very clear for in vivo recordings from visual cortex (Krüger 1991), and it was used to argue that nearly all of the cortical cell correlations resulted from common retinal input. It is therefore prudent to correct for these dependencies.

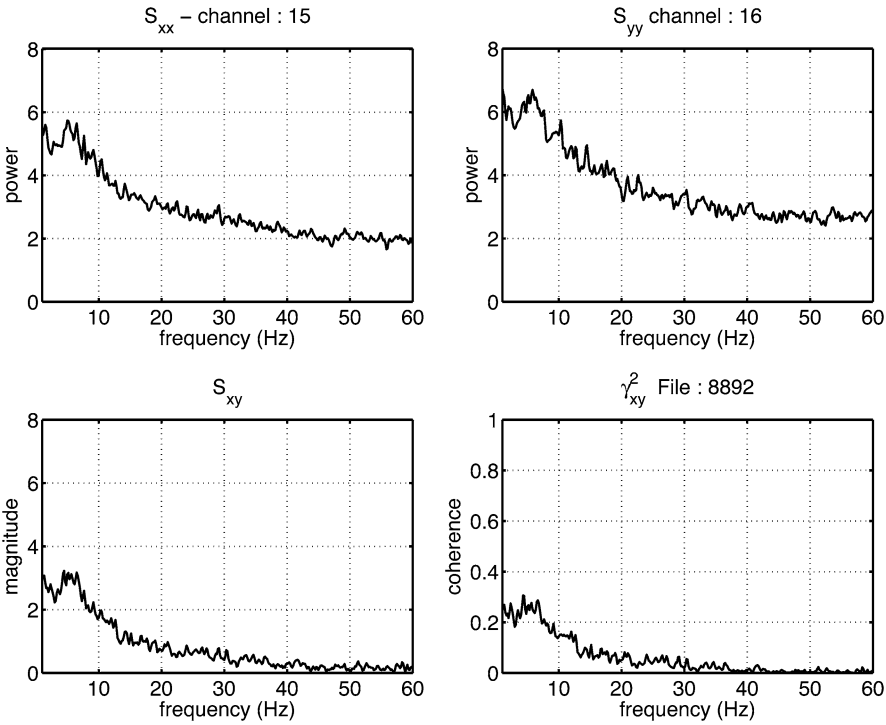

Fig. 5.5 Powerspectra, magnitude of the cross-spectrum and squared coherence. The coherence below 20 Hz is significant at all frequencies. Frequency zero is suppressed in all plots. The magnitudes are given in spikes/stimulus

The coherency $\gamma_{xy}(f)$ corrects for common periodicities (frequencies in the spectra) in the firing rates and also for the level of the unit-firing rates by virtue of the normalization based on the product of the powerspectra of the two spike trains. This does not mean that it eliminates the effects of common input (Tetzlaff et al. 2008). The example shown (Fig. 5.6) was recorded from primary auditory cortex in an anesthetized cat where the EEG showed spindle waves in the frequency range of 8–10 Hz. These periodic oscillations are reflected in the firing times of the spike trains and in the weak oscillatory character of the cross-correlogram (solid line). The coherency corrects for the common periodicity in both spike trains, as can be seen from its inverse Fourier transform (the dashed curve). This correction is akin to deconvolving the cross-correlogram by the geometric mean of the autocorrelation functions of the two spike trains (Eggermont and Smith 1996). One notices also the strong contribution ($>50\%$) of these spindle-related correlations on the peak value of the cross-correlogram.

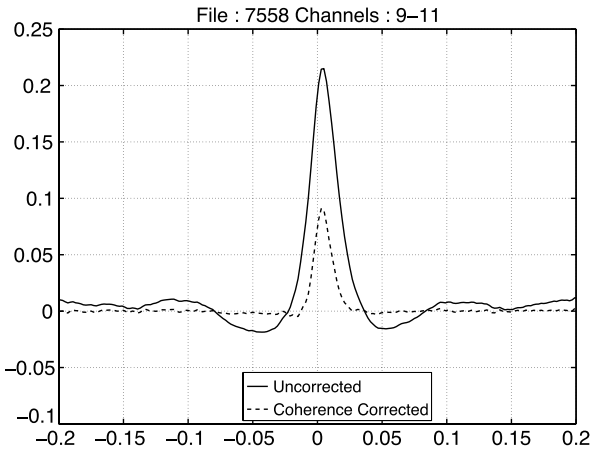

Fig. 5.6 The coherence-based-correction procedure eliminates the oscillatory aspects resulting from anesthesia-induced spindle oscillations in auditory cortex. The *dashed curve* represents the IFFT of the complex coherence. Since both units showed an oscillatory autocorrelation at the same frequency, the effect thereof is eliminated using the coherency. The *vertical axis* represents the cross-correlation coefficient

5.5.2 Oscillatory Cross-Correlograms

Especially in visual cortex, ketamine-anesthetized or awake, cross-correlograms with a pronounced oscillatory character (frequencies in the gamma-band) have been reported frequently (Singer and Gray 1995). In auditory cortex these oscillations have been observed in local field potentials (Barth and MacDonald 1996) but not in spiking activity (Eggermont 1992b; Horikawa et al. 1994). Consequently, these high-frequency oscillations are commonly absent from correlograms in the auditory system. In anesthetized animals spindle-related oscillations are frequently observed (Fig. 5.6), and correction procedures therefore have been described above. The high-frequency oscillations likely reflect common input as the correlograms typically appear symmetric around zero lag, so their effect should disappear in the coherence-corrected correlogram. Oscillatory symmetric correlation functions can also result from the dynamics of the local network (Brunel and Hakim 1999; Meyer and van Vreeswijk 2002), even without common input. Moreover, the coherence can reveal peaks at the network oscillation frequency (see, e.g., Fig. 9 in Tetzlaff et al. 2008, or Fig. 6 in Kriener et al. 2008).

5.5.3 Stimulus Corrections in the Frequency Domain

The effects of stimulus correlations can be incorporated in the frequency domain analysis and corrected for by using the partial coherency (Rosenberg et al. 1998) of spike trains $x(t)$ and $y(t)$ given stimulus $S(t)$, and is defined as

$$\gamma_{xy|S}(f) = \frac{\gamma_{xy}(f) - \gamma_{Sx}(f)\gamma_{yS}(f)}{\sqrt{(1 - |\gamma_{Sx}(f)|^2)(1 - |\gamma_{yS}(f)|^2)}}. \tag{5.19}$$

Here the effect of the stimulus S is incorporated as a common input to neurons x and y.

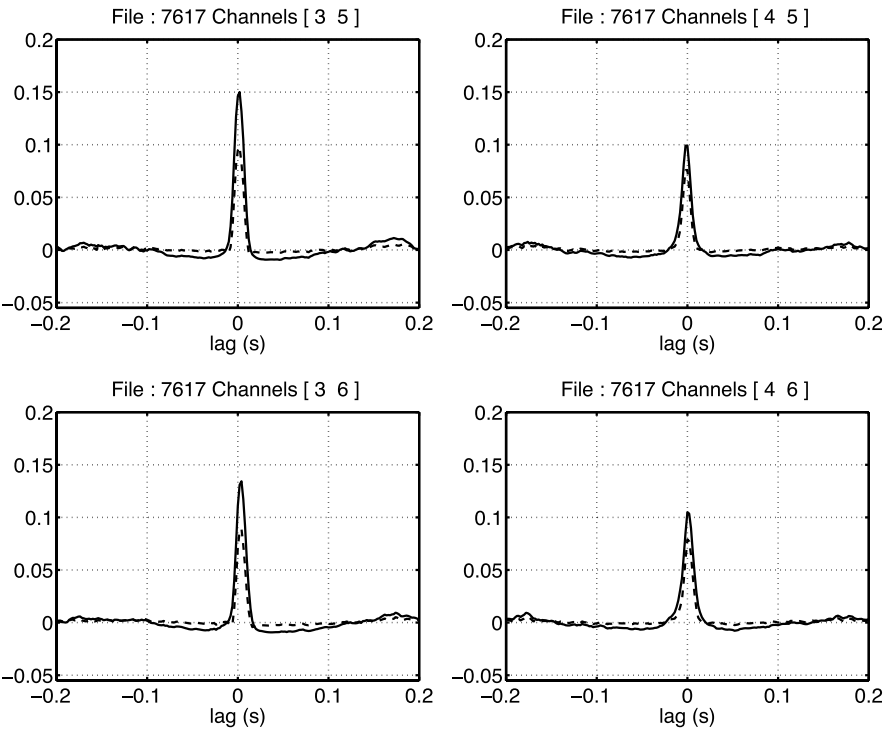

Fig. 5.7 Uncorrected cross-correlation coefficient functions (*solid line*) and coherency-corrected ones (*dashed line*) for stimulation with a Poisson-distributed click train (mean rate 8/s) of 15-minute duration. Four pairs among three simultaneously recorded sorted units are shown. The electrodes were 0.5 mm apart except for the [3 6] pair, which had a separation of 0.7 mm

For the example shown in Figs. 5.3 and 5.5, the partial coherency is smaller than the coherency but only by a few percent. It is clear that either the shift predictor correction is more sensitive or that it is overcorrecting. This overcorrection of the shift predictor is very clear for strong stimulus effects on the firing rate (Eggermont 1994). An example where the difference between the coherence and the partial coherence is somewhat larger is found in data obtained with 15-minute stimulation with a Poisson-distributed click train. In Fig. 5.7 we show the peak cross-correlation coefficient (solid lines) for four simultaneously recorded pairs (neighboring electrode separation 0.5 mm) from primary auditory cortex, as well as the inverse Fourier transform of the (complex) coherence (dashed lines).

For the same data set, we show the magnitude of the coherency (solid lines) and partial coherency (dashed lines) in Fig. 5.8. One observes that even for a fairly strong stimulus correlation as was observed here, the partial coherence is, except at frequency 0 representing the mean value, at most only 15% smaller than the coherence.

In order to compare stimulus correction in the time and frequency domains, we calculated the effect of stimulus-induced common input based on both the shift-

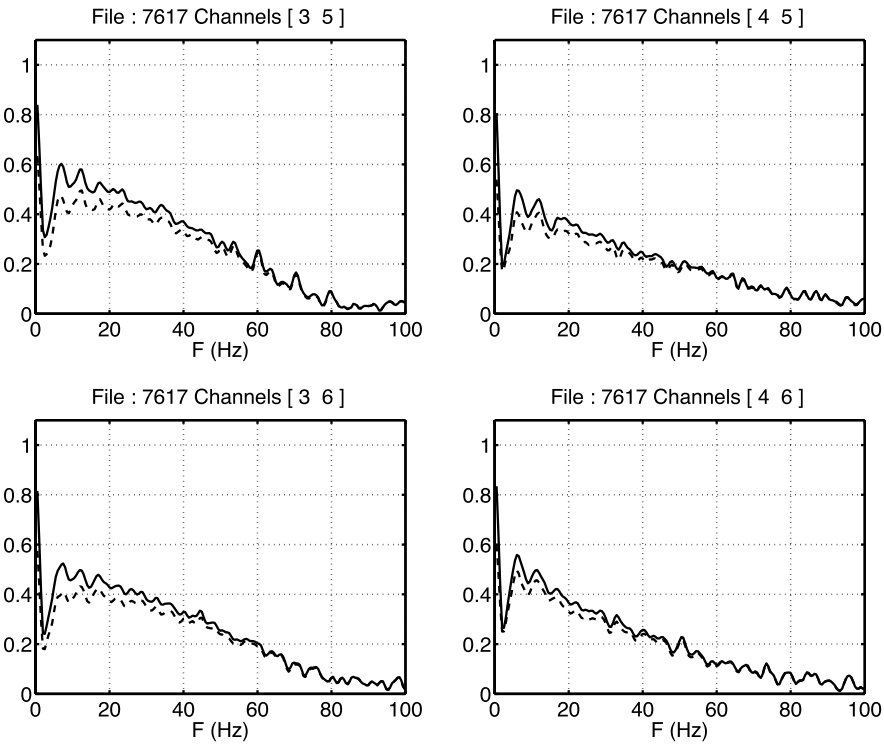

Fig. 5.8 Magnitude of the coherence (*solid line*) and partial coherence conditional upon the stimulus (*dashed line*) for the same pairs as in Fig. 5.7

predictor-corrected coherence and the partial coherence for a very simple model (see insert in Fig. 5.9). Neurons x and y have different and uncorrelated Poisson-distributed firing patterns with the same mean rate. The stimulus-induced activity is also Poisson distributed, is exactly the same for both units, and repeats once (so that a shift predictor can be calculated). The stimulus-evoked firing rate was fine tuned by $P(\text{firing})$ which was between 0 and 1. If we define the signal-to-noise ratio, $\text{SNR}_P = \text{Stimulus rate} * P(\text{firing})/\text{Spontaneous rate}$, then we obtain the following expressions (for this special simple model) for the coherence γ_{xy}, the shift-prediction-corrected coherence $\gamma_{xy|\text{Shift}}$, and the partial coherence $\gamma_{xy|S}$:

$$\begin{aligned}
\gamma_{xy} &= \text{SNR}_P/(1+\text{SNR}_P),\\
\gamma_{xy|\text{Shift}} &= \big[1-P(\text{firing})\big] * \text{SNR}_P/(1+\text{SNR}_P),\\
\gamma_{xy|S} &= \big[1-P(\text{firing})\big] * \text{SNR}_P/\big\{1+\text{SNR}_P * \big[1-P(\text{firing})\big]\big\}.
\end{aligned}$$

The ratio between the shift-predictor-corrected coherence $\gamma_{xy|\text{Shift}}$ and the partial coherence $\gamma_{xy|S}$ is $1-[\text{SNR}_P/(1+\text{SNR}_P)] * P(\text{firing})$. The contours in the figure result from a change in $P(\text{firing})$ from 0 to 1 in counter-clock wise fashion. For $P(\text{firing}) = 0$ and for $P(\text{firing}) = 1$, the shift-predictor-corrected coherence and the

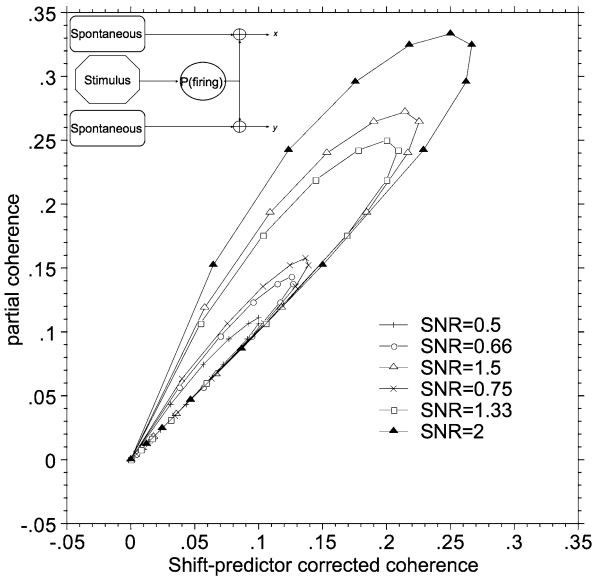

Fig. 5.9 Relationship between the partial coherence (correcting for common input induced by a stimulus) and the shift-predictor-corrected coherence (calculated from an FFT of the shift-predictor corrected cross-correlation function, and the powerspectra of $x(t)$ and $y(t)$) for the model shown in the *insert*. SNR values are representing Stimulus rate/Spontaneous rate, the P(firing) is the variable here (between 0.0125 and 1, anti-clockwise). Thus the smallest SNR_P = Stimulus rate $*$ P(firing)/Spontaneous rate (see text) used was 0.0062. The partial coherence is always highest at $P(\text{firing}) = 0.5$, because of the product $(1 - P(\text{firing}))(P(\text{firing}))$ in the numerator of the equation for $\gamma_{xy|S}$

partial coherence give the same result. For $0 < P(\text{firing}) < 1$, the shift-predictor-corrected coherence is always smaller than the partial coherence, and there is no one-to-one relationship between them. For $P(\text{firing}) < 0.2$, the relationship between shift-predictor-corrected coherence and partial coherence is approximately linear. For lower SNR, this is graphically shown in Fig. 5.9.

There is a time domain counterpart of the partial coherency in the form of the partial correlation optionally with different delays (Stark et al. 2006),

$$\rho_{xy(\tau_1)|S(\tau_2)} = \frac{\rho_{xy(\tau_1)} - \rho_{xS(\tau_2)}\rho_{y(\tau_1)S(\tau_2)}}{\sqrt{(1-\rho^2_{xS(\tau_2)})(1-\rho^2_{y(\tau_1)S(\tau_2)})}}; \tag{5.20}$$

here τ_1 is the delay between $x(t)$ and $y(t)$, and τ_2 is the delay between $S(t)$ onset and $x(t)$. This formulation has the advantage that delayed correlations, such as resulting from conduction delays in the common stimulus correlation, can be taken into account. When no delays are used, one obviously has the partial correlation at delay zero.

5.5.4 Time-Dependent Coherence

Most neurophysiological spike trains, e.g., under stimulus conditions, are nonstationary, although they are often periodic stationary. A simple extension of the coherence function allows visualization of the change in coherence as a function of time after stimulus onset. This can be very useful if long but repeated stimuli are used, such as vocalizations or changing natural scenes. It is also useful to inspect whether the coherence changes for all frequency ranges (e.g., in the delta, alpha, and gamma ranges) in the same way.

The time-dependent coherence is given as

$$\gamma_{xy}^2(\tau, f) = \frac{|S_{xy}(\tau, f)|^2}{S_{xx}(\tau, f) S_{yy}(\tau, f)}. \tag{5.21}$$

Here τ is the time since stimulus onset. If the spike trains are periodic stationary (interval T), the standard averaging procedure is done modulo T, and significance levels can be calculated (Zhan et al. 2006).

5.6 Correlation and Connectivity

In general, there is a unique relationship between known connectivity and observed cross-correlation. The reverse is unfortunately not true; a given cross-correlogram can result from many potential underlying neural connectivities. From the neural correlation one can, only under certain assumptions with respect to the integration of neural input and the shape on the neuron's response curve, estimate the strength of the neural interaction. Because the estimate of the peak neural correlation is very sensitive to the shape and working point of the neuron's response curve (Melssen and Epping 1987), it is generally not permitted to equate the strength of the neural correlation estimated from extracellular recordings with the strength of the neural interaction. When an actual stimulus is applied, one of its effects, especially for cortical neurons, will be to shift the neuron's working point, and apparent changes in neural correlation may be found without a concomitant change in the strength of the neural interaction (Aertsen and Gerstein 1985; Melssen and Epping 1987).

The simplest interactions shown by a pair of neurons are that one neuron makes a direct excitatory or inhibitory connection with the other, or that both neurons share a common excitatory or inhibitory input. It has been shown in simulation studies that common excitatory input and common inhibitory input result in a peak in the cross-correlogram around zero lag time. Reciprocal excitatory and inhibitory inputs to the neurons provided by the same source result in a central dip in the cross-correlogram (Moore et al. 1970). Unilateral excitation shows a peak shifted to positive lag times, and unilateral inhibition shows a dip at positive lag times.

In auditory cortex evidence for direct excitation or inhibition has been scant, unless the neuron pairs were recorded on the same electrode (Eggermont 1992a, 1992b), and all correlograms are of the common-input type. In visual cortex several interaction types have been shown for recordings with dual electrodes from

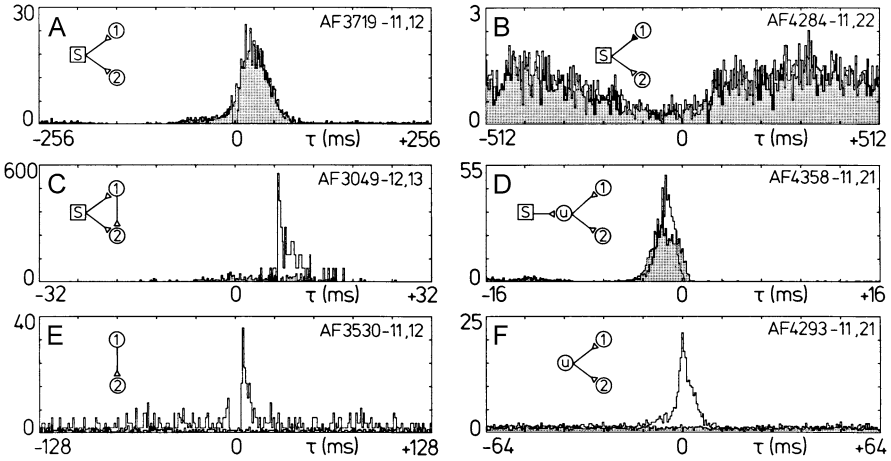

Fig. 5.10 Potential functional connectivity underlying cross-correlation functions. Simultaneous and nonsimultaneous (shift predictor; *shaded*) cross-correlation histograms are superimposed. **A**: neural synchrony, incremental correlogram. **B**: neural synchrony, decremental correlogram. **C**: unidirectional excitatory input. **D**: neural shared input. **E**: unidirectional excitatory input. **F**: neural shared input. Stimulus-driven activity in **A**–**D**, spontaneous activity in **E** and **F**. Note different time scales. Simple connectivity schemes that can account for the observed phenomena are shown as *insets*. *S*, stimulus; *1*, neuron 1; *2*, neuron 2; *u*, unobserved neuron. *Open* and *closed triangles* indicate excitatory and inhibitory connections, respectively. From Epping and Eggermont (1987)

cells at different depth in the same cortical column (Toyama et al. 1981). However, more variety in correlogram shapes was found in the auditory midbrain (Epping and Eggermont 1987). In the examples shown in Fig. 5.10 we show (A) stimulus-induced common excitatory effects without neural correlation for units recorded on the same electrode and (B) stimulus-induced common suppressive effects for a pair recorded on electrodes separated by 130 μm. In both cases the shift predictor is equal to the neural synchrony. Weak stimulus-induced activation combined with a direct excitatory effect from one neuron upon the other recorded on the same electrode (C) is inferred because the shift predictor is hardly visible and the peak of the correlogram is displaced by 6 ms from the origin and highly asymmetrical. This is interpreted as a unidirectional excitatory influence of unit 1 on unit 2. In these three examples the stimulus consisted of random frequency tone pips presented once per second. Part D shows stimulus-induced (Poisson-distributed clicks) common input for a pair recorded with electrodes separated by 290 μm putatively via an unobserved neuron (D), because it has a pronounced shift predictor. For two spontaneous recordings (E, F), reason for the flat shift predictors, unidirectional excitatory input was observed for a single electrode pair (E) and common input for a dual electrode pair (separation 160 μm; F). The highly asymmetrical peak of the neural synchrony displaced from the origin in part E indicates that unit 1 exerts a unidirectional excitatory influence on unit 2. However, there may be a complication due to the dead time of the spike sorting procedure, and it may well be a common input neural cor-

relation as well. This is a known drawback of using sorted units recorded on the same electrode.

5.7 Effects of Spike Sorting on Pair-Correlations

Multiunit (MU) recording is common in structures with high cell densities and especially using multielectrode arrays. Spike sorting can result in well-sorted units, but by no means one can be sure that one is dealing with single units (SU); that is, only the case when doing intracellular or patch-clamp recordings. Evidence from neural modeling suggests that interpretation of neural correlations from multiunit recordings may be ambiguous as it is not a linear combination of correlations for the various single-unit pairs (Bedenbaugh and Gerstein 1997; Gerstein 2000). However, changes in single-unit correlation strengths will be accompanied by comparable changes in the correlation between multiunit activity as shown in Eggermont (2000). Thus, if a number of single units contribute to each multiple-single-unit recording, the changes in the MU values will be positively, but nonlinearly, related to those for the corresponding SU ones. So if SU correlations go up, so will the MU correlation. The only difference is that the R-values are larger for MU than for the corresponding SU ones but smaller than the sum of all the relevant SU pair R-values. Trying to estimate functional connectivity from sorted multiunit activity is not encouraged.

5.8 Correlations and the Brain

Calculating a shift predictor or any other type of stimulus predictor is a way to extract the effects of stimulation on the effective neural connectivity despite the drawbacks that we discussed. However, as we have previously argued (Tomita and Eggermont 2005), it is unlikely that the nervous system performs a correction for stimulus-induced correlation as estimated by the various predictors. It is the actual spike coincidences, i.e., *the neural synchrony*, that are affecting the potential for firing in a target neuron the nonstimulus-corrected ones. Thus raw correlations may effectively estimate those coincident firings between neurons that could play a role in neural population coding of sound. The effect of removing common periodicities in spike firing and common bursting activity from the cross-correlation can be justified because these contributions typically do not occur, or occur much less, in awake animals and are often the result of using anesthesia. Nevertheless, in our data for long steady-state stimuli, the effects of a stimulus correction based on the shift predictor are minimal (Fig. 5.3), so using the stimulus-correction procedure has minimal effect on the interpretation of the data. However, for transient and optimal stimuli, the stimulus correction can be extensive and even remove all correlation (Eggermont 1994), so in this case one has to consider what the correlation estimate needs to support. If one is concerned with estimating connectivity

or deciding which wiring scheme is most likely, then both stimulus corrections and corrections for common, network-induced, activity need to be performed. If one is concerned with how the brain performs its task in vivo, then these corrections may obscure the very aspects one wants to understand.

Acknowledgements This research was supported by the Alberta Heritage Foundation for Medical Research, the Natural Sciences and Engineering Research Council, and by the Campbell McLaurin Chair of Hearing Deficiencies. Greg Shaw provided programming assistance. Tom Tetzlaff read a previous version and provided valuable comments.

References

Aertsen AM, Gerstein GL (1985 Aug 12) Evaluation of neuronal connectivity: sensitivity of cross-correlation. Brain Res 340(2):341–354

Aertsen AM, Smolders JW, Johannesma PI (1979 Mar 19) Neural representation of the acoustic biotope: on the existence of stimulus-event relations for sensory neurons. Biol Cybern 32(3):175–185

Averbeck BB, Lee D (2004 Apr) Coding and transmission of information by neural ensembles. Trends Neurosci 27(4):225–230

Barth DS, MacDonald KD (1996 Sep 5) Thalamic modulation of high-frequency oscillating potentials in auditory cortex. Nature 383(6595):78–81

Bedenbaugh P, Gerstein GL (1997 Aug 15) Multiunit normalized cross correlation differs from the average single-unit normalized correlation. Neural Comput 9(6):1265–1275

Brillinger DR, Bryant HL Jr, Segundo JP (1976 May 17) Identification of synaptic interactions. Biol Cybern 22(4):213–228

Brody CD (1999 Oct 1) Correlations without synchrony. Neural Comput 11(7):1537–1551

Brown EN, Kass RE, Mitra PP (2004 May) Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci 7(5):456–461

Brunel N, Hakim V (1999 Oct 1) Fast global oscillations in networks of integrate-and-fire neurons with low firing rates. Neural Comput 11(7):1621–1671

De la Rocha J, Doiron B, Shea-Brown E, Josić K, Reyes A (2007 Aug 16) Correlation between neural spike trains increases with firing rate. Nature 448(7155):802–806

Eggermont JJ (1992a Oct) Neural interaction in cat primary auditory cortex. Dependence on recording depth, electrode separation, and age. J Neurophysiol 68(4):1216–1228

Eggermont JJ (1992b Aug) Stimulus induced and spontaneous rhythmic firing of single units in cat primary auditory cortex. Hear Res 61(1–2):1–11

Eggermont JJ (1994 Jan) Neural interaction in cat primary auditory cortex II. Effects of sound stimulation. J Neurophysiol 71(1):246–270

Eggermont JJ (2000 May) Sound-induced synchronization of neural activity between and within three auditory cortical areas. J Neurophysiol 83(5):2708–2722

Eggermont JJ, Epping WJ, Aertsen AM (1983) Stimulus dependent neural correlations in the auditory midbrain of the grassfrog (*Rana temporaria L.*). Biol Cybern 47(2):103–117

Eggermont JJ, Smith GM (1995 Nov 13) Rate covariance dominates spontaneous cortical unit-pair correlograms. Neuroreport 6(16):2125–2128

Eggermont JJ, Smith GM (1996 Aug) Neural connectivity only accounts for a small part of neural correlation in auditory cortex. Exp Brain Res 110(3):379–391

Epping WJ, Eggermont JJ (1987 May) Coherent neural activity in the auditory midbrain of the grassfrog. J Neurophysiol 57(5):1464–1483

Gawne TJ, Richmond BJ (1993 Jul) How independent are the messages carried by adjacent inferior temporal cortical neurons?. J Neurosci 13(7):2758–2771

Gerstein GL (2000 Jul 31) Cross-correlation measures of unresolved multi-neuron recordings. J Neurosci Methods 100(1–2):41–51

Gourévitch B, Eggermont JJ (2007 Jun 15) A simple indicator of nonstationarity of firing rate in spike trains. J Neurosci Methods 163(1):181–187

Grün S, Diesmann M, Aertsen A (2002 Jan) Unitary events in multiple single-neuron spiking activity: II. Nonstationary data. Neural Comput 14(1):81–119

Horikawa J, Tanahashi A, Suga N (1994 Jun 1) After-discharges in the auditory cortex of the mustached bat: no oscillatory discharges for binding auditory information. Hear Res 76(1–2):45–52

Joris PX, Louage DH, Cardoen L, van der Heijden M (2006 Jun–Jul) Correlation index: a new metric to quantify temporal coding. Hear Res 216–217:19–30

Kriener B, Tetzlaff T, Aertsen A, Diesmann M, Rotter S (2008 Sep) Correlations and population dynamics in cortical networks. Neural Comput 20(9):2185–2226

Krüger J (1991) Spike train correlations on slow time scales in monkey visual cortex. In: Krüger J (Ed) Neuronal cooperativity. Springer-Verlag, Berlin, Heidelberg, pp 105–132

Melssen WJ, Epping WJ (1987) Detection and estimation of neural connectivity based on cross-correlation analysis. Biol Cybern 57(6):403–414

Meyer C, van Vreeswijk C (2002 Feb) Temporal correlations in stochastic networks of spiking neurons. Neural Comput 14(2):369–404

Moore GP, Segundo JP, Perkel DH, Levitan H (1970 Sep) Statistical signs of synaptic interaction in neurons. Biophys J 10(9):876–900

Neven H, Aertsen A (1992) Rate coherence and event coherence in the visual cortex: a neuronal model of object recognition. Biol Cybern 67(4):309–322

Nowak LG, Munk MH, Nelson JI, James AC, Bullier J (1995 Dec) Structural basis of cortical synchronization. I. Three types of interhemispheric coupling. J Neurophysiol 74(6):2379–2400

Perkel DH, Gerstein GL, Moore GP (1967 Jul) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophys J 7(4):419–440

Ponomarenko SA, Agrawal GP, Wolf E (2004) Energy spectrum of a nonstationary ensemble of pulses. Optics Lett 29:394–396

Rosenberg JR, Halliday DM, Breeze P, Conway BA (1998 Aug 31) Identification of patterns of neuronal connectivity: partial spectra, partial coherence, and neuronal interactions. J Neurosci Methods 83(1):57–72

Singer W, Gray CM (1995) Visual feature integration and the temporal correlation hypothesis. Annu Rev Neurosci 18:555–586

Stark E, Drori R, Abeles M (2006) Partial cross-correlation analysis resolves ambiguity in the encoding of multiple movement features. J Neurophysiol 95:1966–1975

Staude B, Rotter S, Grün S (2008) Can spike coordination be differentiated from rate covariation?. Neural Comput 20:1973–1999

Tomita M, Eggermon JJ (2005) Cross-correlation and joint spectro-temporal receptive field properties in auditory cortex. J Neorophysiol 93:378–392

Toyama K, Kimura M, Tanaka K (1981 Aug) Organization of cat visual cortex as investigated by cross-correlation technique. J Neurophysiol 46(2):202–214

Valentine PA, Eggermont JJ (2004 Oct) Stimulus dependence of spectro-temporal receptive fields in cat primary auditory cortex. Hear Res 196(1–2):119–133

Van Drongelen W (2007) Signal processing for neuroscientists. Academic Press, Burlington

Van Stokkum IH, Johannesma PI, Eggermont JJ (1986) Representation of time-dependent correlation and recurrence time functions. A new method to analyse non-stationary point processes. Biol Cybern 55(1):17–24

Tetzlaff T, Rotter S, Stark E, Abeles M, Aertsen A, Diesmann M (2008) Dependence of neuronal correlations on filter characteristics and marginal spike train statistics. Neural Comput 20(9):2133–2184

Yang XW, Shamma SA (1990 May) Identification of connectivity in neural networks. Biophys J 57(5):987–999

Voigt HF, Young ED (1990 Nov) Cross-correlation analysis of inhibitory interactions in dorsal cochlear nucleus. J Neurophysiol 64(5):1590–1610

Zhan Y, Halliday D, Jiang P, Liu X, Feng J (2006 Sep 30) Detecting time-dependent coherence between non-stationary electrophysiological signals—a combined statistical and time-frequency approach. J Neurosci Methods 156(12):322–332

Chapter 6
Dependence of Spike-Count Correlations on Spike-Train Statistics and Observation Time Scale

Tom Tetzlaff and Markus Diesmann

Abstract Spiking activity is typically measured by counting the number of spikes in a certain time interval. The length of this interval, the "bin size", varies considerably across studies. In this chapter, we provide a mathematical framework to relate the spike-count statistics to the statistics of the underlying point processes. We show that spike-count variances, covariances, and correlation coefficients generally depend in a nontrivial way on the bin size and on the spike-train auto- and cross-correlation structure. The spike-count coherence, in contrast, constitutes a correlation measure independent of bin size.

6.1 Introduction

Spiking activity is commonly considered as the computational basis of neural processing. Spike data, however, can be represented in many different forms. A standard measure of spiking activity is the *spike count*, i.e., the number of observed action potentials in a given time interval. Depending on the underlying question and method, the length of this time interval—the bin size—can considerably differ in different studies and preparations. Measured "spike trains", for example, are typically spike-count signals at small time scales in the millisecond or submillisecond range, usually resulting in binary sequences of zeros and ones. In many studies, however, spike counts are computed on larger time scales of several milliseconds, seconds, or even minutes (e.g., Bair et al. 2001; Kohn and Smith 2005). This chapter is particularly dedicated to the question to what extent the choice of the observation time scale affects standard correlation measures like correlation functions, correlation coefficients, or coherences. To this

T. Tetzlaff (✉)
Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences, Drøbakveien 31, 1432 Ås, Norway
e-mail: tom.tetzlaff@umb.no
url: http://arken.umb.no/~tomt

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_6, © Springer Science+Business Media, LLC 2010

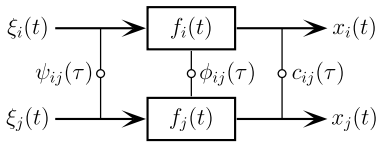

Fig. 6.1 Relation between shot-noise and spike-train correlation functions. Two spike trains $\xi_i(t)$ and $\xi_j(t)$ are transformed by the linear filters $f_i(t)$ and $f_j(t)$, respectively, into shot-noise signals $x_i(t)$ and $x_j(t)$. The resulting shot-noise correlation function $c_{ij}(\tau)$ follows from the spike-train correlation function $\psi_{ij}(\tau)$ by convolution with the filter correlation $\phi_{ij}(\tau)$ (see (6.6))

end, we will use a rather general approach and consider spike-count signals as linearly filtered point processes, i.e., as *shot-noise* signals. The results of this chapter can therefore be applied to a much broader class of neuronal signals: intracellular membrane potentials, membrane currents, and the resulting extracellular signals like local-field potentials, ECoG, or EEG signals can be considered as superpositions of spike trains filtered by components of the neural system (synapses, membranes) and the measurement process. In many cases, these filters can be approximated by linear filters. Here, we will mainly focus on spike-count signals. For a more general discussion, see Tetzlaff et al. (2008).

In Sect. 6.2, we will show that the second-order statistics of shot-noise signals, quantified, for example, by correlation functions, variances, covariances, or correlation coefficients, generally exhibits a complex dependence on the filter properties (e.g., the bin size of spike counts) and the statistics of the underlying point processes (spike trains)—even under simplifying conditions like stationarity. The shot-noise coherence, in contrast, is independent of a (unique) linear filter kernel and therefore provides a much less ambiguous correlation measure (as compared to correlation coefficients).

One of the major causes of correlated firing in neural networks is common presynaptic input. In Sect. 6.3, we will therefore consider a simple common-input model which allows us to generate correlated spike trains in a natural fashion and to illustrate how the spike-train statistics and the observation time scale alter measured spike-count correlations.

6.2 Shot-Noise Correlations

This section reviews how linear filtering of spike-trains affects the correlation structure of the resulting shot-noise signals both in the time and frequency domain (see Fig. 6.1).

6.2.1 Shot-Noise

In the following, we consider signals $x_i(t)$ constructed from spike trains[1] $\xi_i(t)$ by convolution ($*$) with some time-invariant kernel $f_i(t)$ (linear filtering):

$$x_i(t) := (\xi_i * f_i)(t) = \int_{-\infty}^{\infty} \mathrm{d}s\, \xi_i(s) f_i(t-s). \tag{6.1}$$

In the literature, $x_i(t)$ is commonly called *shot-noise* if $\xi_i(t)$ is a realization of a Poisson point process (e.g., Papoulis and Pillai 2002). Here, we adopt this term for general point processes. Various measures in theoretical and experimental neuroscience can be described as shot-noise. The subthreshold membrane potential of the linear Integrate-and-Fire neuron model with current-based synapses (Tuckwell 1988), for example, is a convolution of the incoming spike trains with the postsynaptic potentials (PSPs). It has been shown that population signals like local-field potentials (LFPs), EEG, and even fMRI (BOLD) signals exhibit a considerable correlation with linearly filtered spike data (e.g., Logothetis et al. 2001; Mukamel et al. 2005).

In spike-train analysis, we typically have to deal with *spike counts*

$$x_i^h(t) := \int_t^{t+h} \mathrm{d}s\, \xi_i(s), \tag{6.2}$$

i.e., the number of spikes observed in a time window $[t, t+h)$. The spike count (6.2) is a shot-noise process since it can be considered as resulting from the convolution of a spike train $\xi_i(t)$ with the rectangular kernel

$$f_i(t) = \begin{cases} 1, & -h < t \le 0, \\ 0, & \text{else.} \end{cases} \tag{6.3}$$

In many applications, the spike count is computed on a discrete-time grid $t \in \{k \cdot h \mid k \in \mathbb{N}\}$, whereas here we consider the general case of continuous time where spikes are counted in a sliding window (moving average). The results of this section are formally the same both for the continuous and discrete cases. In the latter, time integrals (convolutions) have to be replaced by sums over time steps.

6.2.2 Correlation Functions

We define the two-dimensional *spike-train* and *shot-noise correlation functions* $\psi_{ij}(t,t') := \mathrm{E}[\xi_i(t)\xi_j(t')]$ and $c_{ij}(t,t') := \mathrm{E}[x_i(t)x_j(t')]$, respectively, as the expected[2] products of the two spike trains $\xi_i(t)$, $\xi_j(t')$ and the two shot-noise signals

[1] Throughout this chapter, we refer to a specific spike-train realization $\xi_i(t)$ as a sum over delta-functions centered at the spike times t_i^k: $\xi_i(t) := \sum_k \delta(t - t_i^k)$.

[2] Here, $\mathrm{E}[\cdot]$ denotes an average over realizations (trials).

$x_i(t)$, $x_j(t')$ evaluated at times t and t' (Aertsen et al. 1989). Offset subtraction, $\tilde{\xi}_i(t) := \xi_i(t) - \nu_i(t)$ and $\tilde{x}_i(t) := x_i(t) - (\nu_i * f_i)(t)$, yields the corresponding *covariance functions*

$$\begin{aligned}\tilde{\psi}_{ij}(t,t') &:= \mathrm{E}\big[\tilde{\xi}_i(t)\tilde{\xi}_j(t')\big],\\ \tilde{c}_{ij}(t,t') &:= \mathrm{E}\big[\tilde{x}_i(t)\tilde{x}_j(t')\big].\end{aligned} \tag{6.4}$$

Here, $\nu_i(t) := \mathrm{E}[\xi_i(t)]$ denotes the instantaneous rate of the process $\xi_i(t)$ (see Chap. 2). According to (6.1) and (6.4), the shot-noise and spike-train covariance functions are linked by a two-dimensional convolution with the filter kernels $f_i(t)$ and $f_j(t)$:

$$\tilde{c}_{ij}(t,t') = \int_{-\infty}^{\infty} \mathrm{d}s \int_{-\infty}^{\infty} \mathrm{d}s'\, \tilde{\psi}_{ij}(s,s') f_i(t-s) f_j(t'-s'). \tag{6.5}$$

Assuming time invariance of $\tilde{\psi}_{ij}(t,t')$ (second-order stationarity), we obtain the one-dimensional shot-noise covariance function (Papoulis and Pillai 2002)

$$\tilde{c}_{ij}(\tau) := \tilde{c}_{ij}(t,t+\tau) = (\tilde{\psi}_{ij} * \phi_{ij})(\tau) \tag{6.6}$$

as a convolution between the one-dimensional spike-train covariance function

$$\tilde{\psi}_{ij}(\tau) := \tilde{\psi}_{ij}(t,t+\tau) = \tilde{\psi}_{ij}(0,\tau) \quad (\forall t,\tau) \tag{6.7}$$

and the (deterministic) correlation function $\phi_{ij}(\tau) := \int_{-\infty}^{\infty} \mathrm{d}t\, f_i(t) f_j(t+\tau)$ of the two filter kernels $f_i(t)$ and $f_j(t)$.

For the analysis of parallel spike-count signals $x_i^h(t)$ and $x_j^h(t)$, typically a unique bin size h is used. The filter kernels $f_i(t)$ and $f_j(t)$ given in (6.3) are therefore identical, i.e., $f(t) := f_i(t) = f_j(t)$. According to (6.6), the spike-count covariance function $\tilde{c}_{ij}(\tau)$ results from smoothing the spike-train covariance function $\tilde{\psi}_{ij}(\tau)$ with the triangular autocorrelation[3]

$$\phi(\tau) = \int_{-\infty}^{\infty} \mathrm{d}t\, f(t) f(t+\tau) = \begin{cases} h - |\tau|, & -h < \tau \le h,\\ 0, & \text{else,}\end{cases} \tag{6.9}$$

of the rectangular spike-count kernel $f(t)$.

[3] If $f_i(t) \neq f_j(t)$ and assuming $h_i < h_j$, $\phi(\tau)$ has to be replaced by the filter *cross*-correlation

$$\phi_{ij}(\tau) = \begin{cases} 0, & \tau \le -h_j,\\ h_j + \tau, & -h_j < \tau \le -(h_j - h_i),\\ h_i, & -(h_j - h_i) < \tau \le 0,\\ h_i - \tau, & 0 < \tau \le h_i,\\ 0, & \tau > h_i.\end{cases} \tag{6.8}$$

6.2.3 Variance, Covariance and Correlation Coefficient

The shot-noise *covariance* $\tilde{c}_{ij}$ (or *variance* for $i = j$) is obtained by evaluating the covariance function $\tilde{c}_{ij}(\tau)$ at zero lag $\tau = 0$. According to (6.6), this is the area of the product of the spike-train covariance function and the filter correlation function:

$$\tilde{c}_{ij} := \tilde{c}_{ij}(0) = \int_{-\infty}^{\infty} \mathrm{d}t\, \tilde{\psi}_{ij}(t)\phi_{ij}(-t). \tag{6.10}$$

For signals $x_i(t)$, $x_j(t)$ with finite variance, the covariance can be normalized by the fluctuations of the individual signals to the scale $[-1, 1]$. This defines Pearson's *correlation coefficient* (Perkel et al. 1967b; Hollander and Wolfe 1999; Feller 1971)

$$r_{ij} := \frac{\tilde{c}_{ij}}{\sqrt{\tilde{c}_{ii}\tilde{c}_{jj}}} = \frac{\int_{-\infty}^{\infty} \mathrm{d}t\, \tilde{\psi}_{ij}(t)\phi_{ij}(-t)}{\sqrt{\int_{-\infty}^{\infty} \mathrm{d}t\, \tilde{\psi}_{ii}(t)\phi_{ii}(t) \int_{-\infty}^{\infty} \mathrm{d}t'\, \tilde{\psi}_{jj}(t')\phi_{jj}(t')}}. \tag{6.11}$$

Note that r_{ij} generally depends on both the (joint and marginal) statistics of the underlying spike trains and the features of the filter kernels, even if the kernels $f_i(t)$ and $f_j(t)$ are identical, i.e., $\phi_{ij}(t) = \phi_{ii}(t) = \phi_{jj}(t)$. Only if, in addition, all spike-train covariance functions $\tilde{\psi}_{ii/jj/ij}(\tau)$ are delta-shaped (e.g., precisely correlated, stationary Poisson processes), the filter contributions cancel out.

With the autocorrelation (6.9) of the spike-count kernel (6.3), we obtain the spike-count covariance

$$\tilde{c}_{ij}^{h} = \int_{-h}^{h} \mathrm{d}\tau\, \big(h - |\tau|\big) \cdot \tilde{\psi}_{ij}(\tau) \tag{6.12}$$

for a given (unique) bin size h. For $i = j$, (6.12) resembles the result for the spike-count variance presented in Papoulis and Pillai (2002). The spike-count correlation coefficient

$$r_{ij}^{h} := \frac{\tilde{c}_{ij}^{h}}{\sqrt{\tilde{c}_{ii}^{h}\tilde{c}_{jj}^{h}}} = \frac{\int_{-h}^{h} \mathrm{d}\tau\, (h - |\tau|)\tilde{\psi}_{ij}(\tau)}{\sqrt{\int_{-h}^{h} \mathrm{d}\tau\, (h - |\tau|)\tilde{\psi}_{ii}(\tau) \int_{-h}^{h} \mathrm{d}\tau'\, (h - |\tau'|)\tilde{\psi}_{jj}(\tau')}} \tag{6.13}$$

is, up to the triangular prefactors $(h - |\tau|)$, the normalized area of the spike-train cross-covariance function in the interval $[-h, h]$. The dependence of the spike-count covariance $\tilde{c}_{ij}^{h}$ and correlation coefficient r_{ij}^{h} on the bin size is emphasized here by the superscript "h".

Consider the simple example of two stationary Poisson processes $\xi_i(t)$, $\xi_j(t)$ with constant rates ν_i, ν_j and autocovariance functions (Papoulis and Pillai 2002)

$$\tilde{\psi}_{ii/jj}(\tau) = \nu_{i/j}\delta(\tau). \tag{6.14}$$

Applying (6.12) immediately recovers the well-known result for the spike-count variance of a Poisson process,

$$\tilde{c}_{ii/jj}^{h} = \nu_{i/j} \int_{-h}^{h} \mathrm{d}\tau\, \big(h - |\tau|\big) \cdot \delta(\tau) = \nu_{i/j} h, \tag{6.15}$$

with a linear dependence on the bin size h. A similar result is obtained for the count covariance $\tilde{c}_{ij}^h$ of two processes with delta-shaped cross-covariance function $\tilde{\psi}_{ij}(\tau) \sim \delta(\tau)$. Only in this highly artificial case, the dependence on the bin size h disappears in the correlation coefficient r_{ij}^h. Natural spike trains typically exhibit structured covariance functions. It is a major objective of this chapter to point out that a non-delta-type correlation structure leads to a complex dependence of correlation coefficients on the properties of the filter kernels, in particular on the bin size h for spike-count signals.

6.2.4 Spectra and Coherence

Correlations in or between time series are often considered not only in the time but also in the Fourier (frequency) domain (see Chap. 5). If we denote $\Xi_i(\omega)$ and $F_i(\omega)$ as the Fourier transforms[4] of the spike-train $\xi_i(t)$ and the filter kernel $f_i(t)$, respectively, the Fourier transform of the shot-noise $x_i(t)$ defined in (6.1) reads $X_i(\omega) = \Xi_i(\omega)F_i(\omega)$. Given the one- and two-dimensional spike-train spectra[5]

$$\tilde{\Psi}_{ij}(\omega,\omega') := \mathfrak{F}\big[\tilde{\psi}_{ij}(t,t')\big](\omega,\omega') \quad \text{and} \quad \tilde{\Psi}_{ij}(\omega) := \mathfrak{F}\big[\tilde{\psi}_{ij}(\tau)\big](\omega), \quad (6.16)$$

we obtain

$$\tilde{C}_{ij}(\omega,\omega') = \tilde{\Psi}_{ij}(\omega,\omega')F_i(\omega)F_j(\omega') \quad \text{and} \quad \tilde{C}_{ij}(\omega) = \tilde{\Psi}_{ij}(\omega)\Phi_{ij}(\omega) \quad (6.17)$$

as the one- and two-dimensional power- ($i = j$) and cross-spectra ($i \neq j$) of the shot-noise signals by Fourier-transforming equations (6.5) and (6.6), respectively. $\Phi_{ij}(\omega) := F_i(\omega)F_j^*(\omega)$ in (6.17) denotes the power- ($i = j$) or cross-spectrum ($i \neq j$) of the filter kernels $f_i(t)$ and $f_j(t)$ (the superscript "$*$" represents the complex conjugate).

A normalized correlation measure in the frequency domain is the *complex coherence* (Priestley 1983; Jarvis and Mitra 2001)

$$\kappa'_{ij}(\omega) := \frac{\tilde{C}_{ij}(\omega)}{\sqrt{\tilde{C}_{ii}(\omega)\tilde{C}_{jj}(\omega)}} = \frac{\tilde{\Psi}_{ij}(\omega)\Phi_{ij}(\omega)}{\sqrt{\tilde{\Psi}_{ii}(\omega)\tilde{\Psi}_{jj}(\omega)\Phi_{ii}(\omega)\Phi_{jj}(\omega)}}, \quad (6.18)$$

which is defined as the ratio between the cross-spectrum $\tilde{C}_{ij}(\omega)$ and the geometric mean of the power spectra $\tilde{C}_{ii/jj}(\omega)$. Its modulus (amplitude) $\kappa(\omega) := |\kappa'(\omega)|$, restricted to the range $[0, 1]$, is called *coherence*. The phase of the complex coherence (6.18) contains information about the temporal alignment of the two signals $x_i(t)$ and $x_j(t)$ and can therefore be used to study delays or negative correlations (anti-correlations). Note, that the definition of the coherence is only meaningful at frequencies with nonvanishing power.

[4] Throughout this chapter, Fourier transforms will be represented by capital letters.

[5] $\mathfrak{F}[\cdot](\omega,\omega')$ and $\mathfrak{F}[\cdot](\omega)$ denote the two- and one-dimensional Fourier integrals, respectively.

With $\Phi_{ij}(\omega) = F_i(\omega)F_j^*(\omega)$, it turns out that the coherence $\kappa(\omega)$ is—in contrast to the correlation coefficient r_{ij} in (6.11)—independent of the linear filter kernels $f_{i/j}(t)$ and exclusively reflects the statistical properties of the spike trains (Brown et al. 2004):

$$\kappa(\omega) = \frac{|\tilde{\Psi}_{ij}(\omega)|}{\sqrt{\tilde{\Psi}_{ii}(\omega)\tilde{\Psi}_{jj}(\omega)}} \quad \big(\text{for any } f_i(t) \text{ and } f_j(t)\big). \tag{6.19}$$

This, however, does not hold if the shot-noise signals arise from superpositions $x_i(t) = \sum_{k=1}^{n}(\xi_k * f_{ik})(t)$ of n spike trains convolved with different kernels $f_{ik}(t)$. In this case, the spectra read $\tilde{C}_{ij}(\omega) = \sum_k \sum_l \tilde{\Psi}_{kl}(\omega)F_{ik}(\omega)F_{jl}^*(\omega)$. In general, the resulting coherence is filter independent only if the kernels $f_{ik}(t)$ $(k \in [1, n])$ are identical.

Note that $\kappa'(\omega)$ evaluated at frequency $\omega = 0$ is the area of the covariance function normalized by the areas of the autocovariance functions

$$\kappa'(0) = \frac{\int_{-\infty}^{\infty} \mathrm{d}\tau\, \tilde{c}_{ij}(\tau)}{\sqrt{\int_{-\infty}^{\infty} \mathrm{d}\tau\, \tilde{c}_{ii}(\tau) \int_{-\infty}^{\infty} \mathrm{d}\tau'\, \tilde{c}_{jj}(\tau')}}. \tag{6.20}$$

In the neuroscientific context, $\kappa'(0)$ is frequently called "correlation coefficient", too (e.g., Bair et al. 2001; Kohn and Smith 2005; Moreno-Bote and Parga 2006). The motivation to prefer $\kappa'(0)$ over r_{ij} in these works is the observation that peaks in neuronal correlation functions typically have some temporal extent and that the width of the peaks varies depending on the system and the experimental protocol. Therefore, an integrated measure appears adequate. Our considerations above demonstrate an additional advantage of $\kappa'(0)$: it is independent of a joint shot-noise kernel. Here, we reserve the term "correlation coefficient" for r_{ij} as defined in (6.11).

6.3 Spike-Count Correlations in a Simple Common-Input Model

Equipped with the formalism to study the correlation between two shot-noise signals developed in the previous section, we demonstrate in this section how non-Poissonian spike statistics and nonstationary firing rates affect pairwise correlations between spike-count signals generated by a simple common-input model. To this end, we define a minimal structural model describing two neurons sharing part of their inputs (Sect. 6.3.1) and derive the resulting spike-train correlation functions (Sect. 6.3.2). Section 6.3.3 exploits the results of Sect. 6.2 to calculate the variance, the covariance, and the correlation coefficient of spike-count signals for two examples introduced in Sects. 6.3.2.1 and 6.3.2.2. A Gamma process (Sect. 6.3.2.1) is considered to highlight how measured correlations depend on the autocorrelation of the common source. An inhomogeneous Poisson processes with sinusoidal rate modulation in time and random phase across trials (Sect. 6.3.2.2) is employed to clarify the notion of nonstationarity in the context of the correlation coefficient.

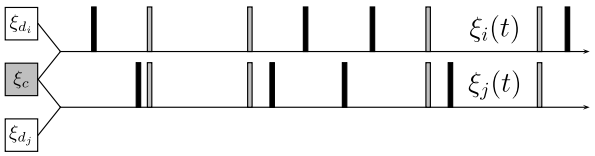

Fig. 6.2 Sketch of two correlated model spike trains $\xi_i(t)$, $\xi_j(t)$ constructed by merging spikes from a common source $\xi_c(t)$ (*gray bars*) and two disjoint independent Poissonian sources $\xi_{d_i}(t)$ and $\xi_{d_j}(t)$ (*black bars*). The common source $\xi_c(t)$ is modeled either as a Gamma process (Sect. 6.3.2.1) or as an inhomogeneous Poisson process with oscillating firing rate (Sect. 6.3.2.2). Figure modified from Tetzlaff et al. (2008)

The subsequent section Sect. 6.3.4 demonstrates that for a large class of processes, the high-frequency coherence reflects the common-input strength and therefore provides an unambiguous measure that depends neither on the filter nor on the marginal spike-train statistics. The final part (Sect. 6.3.5) of this section is concerned with the natural situation where spike cross-correlations exhibit a temporal dispersion as originating from heterogeneous delays, a finite rise time of the postsynaptic potentials, or other mechanisms.

6.3.1 Model Definition

Two spike trains $\xi_i(t)$ and $\xi_j(t)$ constituting the total presynaptic activity of two neurons i and j are constructed by superimposing two disjoint processes $\xi_{d_i}(t)$ and $\xi_{d_j}(t)$ with a process $\xi_c(t)$ shared by both neurons:

$$\xi_i(t) = \xi_{d_i}(t) + \xi_c(t), \qquad \xi_j(t) = \xi_{d_j}(t) + \xi_c(t) \tag{6.21}$$

(see Fig. 6.2 for an illustration of the architecture). To reduce the number of parameters, we assume that the rates of $\xi_i(t)$ and $\xi_j(t)$ are identical, $\nu(t) := \nu_i(t) = \nu_j(t)$. The strength of the common source is parameterized by the relative contribution $\alpha := \nu_c(t)/\nu(t)$ of its firing rate to the total rate. The rates of the background processes $\xi_{d_i}(t)$ and $\xi_{d_j}(t)$ can thus be expressed as $\nu_d(t) := \nu_{d_i}(t) = \nu_{d_j}(t) = (1-\alpha)\nu(t)$.

6.3.2 Correlation Functions

The (one-dimensional) auto- and cross-covariance functions of the two (centered) spike trains $\tilde{\xi}_i(t)$ and $\tilde{\xi}_j(t)$ are

$$\tilde{\psi}_{ii}(\tau) = \mathrm{E}\big[\tilde{\xi}_i(t)\tilde{\xi}_i(t+\tau)\big] = \tilde{\psi}_{d_i d_i}(\tau) + \tilde{\psi}_{d_i c}(\tau) + \tilde{\psi}_{c d_i}(\tau) + \tilde{\psi}_{cc}(\tau), \tag{6.22}$$

$$\tilde{\psi}_{ij}(\tau) = \mathrm{E}\big[\tilde{\xi}_i(t)\tilde{\xi}_j(t+\tau)\big] = \tilde{\psi}_{d_i d_j}(\tau) + \tilde{\psi}_{d_i c}(\tau) + \tilde{\psi}_{c d_j}(\tau) + \tilde{\psi}_{cc}(\tau). \tag{6.23}$$

In general, both are determined by the marginal and the joint second-order spike-train statistics. Let us, again for the sake of simplicity, assume that the disjoint and the common processes are mutually uncorrelated:

$$\tilde{\psi}_{d_i d_j}(\tau) = \tilde{\psi}_{d_i c}(\tau) = \tilde{\psi}_{d_j c}(\tau) = 0. \tag{6.24}$$

In this case, (6.22) and (6.23) reduce to

$$\tilde{\psi}_{ii}(\tau) = \tilde{\psi}_{d_i d_i}(\tau) + \tilde{\psi}_{cc}(\tau), \tag{6.25}$$

$$\tilde{\psi}_{ij}(\tau) = \tilde{\psi}_{cc}(\tau). \tag{6.26}$$

The disjoint inputs are modeled as stationary Poissonian sources with constant firing rate ν_d; hence,

$$\tilde{\psi}_{d_i d_i}(\tau) = \nu_d \delta(\tau). \tag{6.27}$$

In order to dissect the effect of different aspects of the common source process $\xi_c(t)$ on common-input correlations, we investigate two specific cases. The stationary Gamma process (Sect. 6.3.2.1) is discussed as a simple example of a non-Poissonian point processes in order to demonstrate how common-input correlations are altered by the interval distribution of the common source process. The effect of nonstationarity in time and across trials on common-input correlations is discussed by modeling the common source process as an inhomogeneous Poisson process with sinusoidal rate and random phase (Sect. 6.3.2.2).

6.3.2.1 Gamma Source

Let the common source emit spikes at intervals drawn from a Gamma distribution

$$p_1(\tau) = \nu_c \gamma \frac{(\nu_c \gamma \tau)^{\gamma - 1}}{(\gamma - 1)!} \exp(-\nu_c \gamma \tau) \tag{6.28}$$

with positive integer orders $\gamma \in \mathbb{N}^+$. The autocorrelation function of a general point process is determined by the sum over all kth-order interval distributions $p_k(\tau)$ (Perkel et al. 1967a, see also Chap. 1 of this book):

$$\psi_{cc}(\tau) = \nu_c \left(\delta(\tau) + \sum_{k=1}^{\infty} p_k\big(|\tau|\big) \right). \tag{6.29}$$

For any renewal process, consecutive intervals are independent (Cox 1962). Therefore, $p_k(\tau)$ is the k-fold convolution of the first-order density

$$p_k(\tau) = \underbrace{(p_1 * \cdots * p_1)}_{k}(\tau). \tag{6.30}$$

As (6.30) factorizes in the Fourier domain, i.e., $P_k(\omega) = P_1(\omega)^k$, the power-spectrum of a renewal process reads:

$$\begin{aligned}\Psi_{cc}(\omega) &= \mathfrak{F}\big[\psi_{cc}(\tau)\big](\omega)\\ &= \nu_c\left(1+\sum_{k=1}^{\infty}\big\{P_1(\omega)^k + P_1^*(\omega)^k\big\}\right)\\ &= \nu_c\left(1-2+\sum_{k=0}^{\infty}\big\{P_1(\omega)^k + P_1^*(\omega)^k\big\}\right)\\ &= \nu_c\big(\big[1-P_1(\omega)\big]^{-1} + \big[1-P_1^*(\omega)\big]^{-1} - 1\big). \end{aligned} \tag{6.31}$$

The Fourier transformed first-order interval density of the Gamma process is given by Cox (1962)

$$P_1(\omega) = \mathfrak{F}\big[p_1(t)\big](\omega) = \left(\frac{\gamma\nu_c}{\gamma\nu_c + \imath\omega}\right)^{\gamma}. \tag{6.32}$$

The autocorrelation function $\psi_{cc}(\tau)$ can now be obtained by (numerically) computing the inverse Fourier transform of the spectrum (6.31). A closed analytical expression for the autocovariance function can also be derived by a direct evaluation of (6.29)[6]:

$$\tilde{\psi}_{cc}(\tau) = \nu_c\delta(\tau) + \nu_c^2\sum_{l=1}^{\gamma-1} \mathrm{e}^{2\pi i l/\gamma}\exp\big(\gamma\nu_c\tau\big[\mathrm{e}^{2\pi i l/\gamma}-1\big]\big). \tag{6.33}$$

Gamma processes are frequently considered as models of neuronal firing since they can mimic the refractory behavior of neurons following spike emission. For $\gamma > 1$, short inter-spike intervals become more and more unlikely. This is reflected in the autocovariance functions which exhibit a trough around the central peak at $\tau = 0$ (see Fig. 6.3A, left). A limitation of the choice of Gamma processes is their tendency to become more and more regular with the coefficient of variation scaling as $1/\sqrt{\gamma}$ (Cox 1962). Observed coefficients of variation in cortical spike trains, however, are close to one (Softky and Koch 1993).

6.3.2.2 Inhomogeneous Poisson Source

In a second example, we model the common source as a doubly stochastic process ("Cox process"; see Daley and Vere-Jones 2005) where not only the spike-train realizations l but also the rate profiles are random. In the kth trial, $\xi_c^l(t|k)$ is considered as a realization of a Poisson process with a time-dependent rate function $\nu_c^k(t) := \mathrm{E}_l[\xi_c^l(t|k)]$ and autocorrelation (Papoulis and Pillai 2002)

$$\psi_{cc}^k(t,t') := \mathrm{E}_l\big[\xi_c^l(t|k)\xi_c^l(t'|k)\big] = \nu_c^k(t)\delta\big(t-t'\big) + \nu_c^k(t)\nu_c^k\big(t'\big). \tag{6.34}$$

[6]For a derivation, see Pipa et al. (2010).

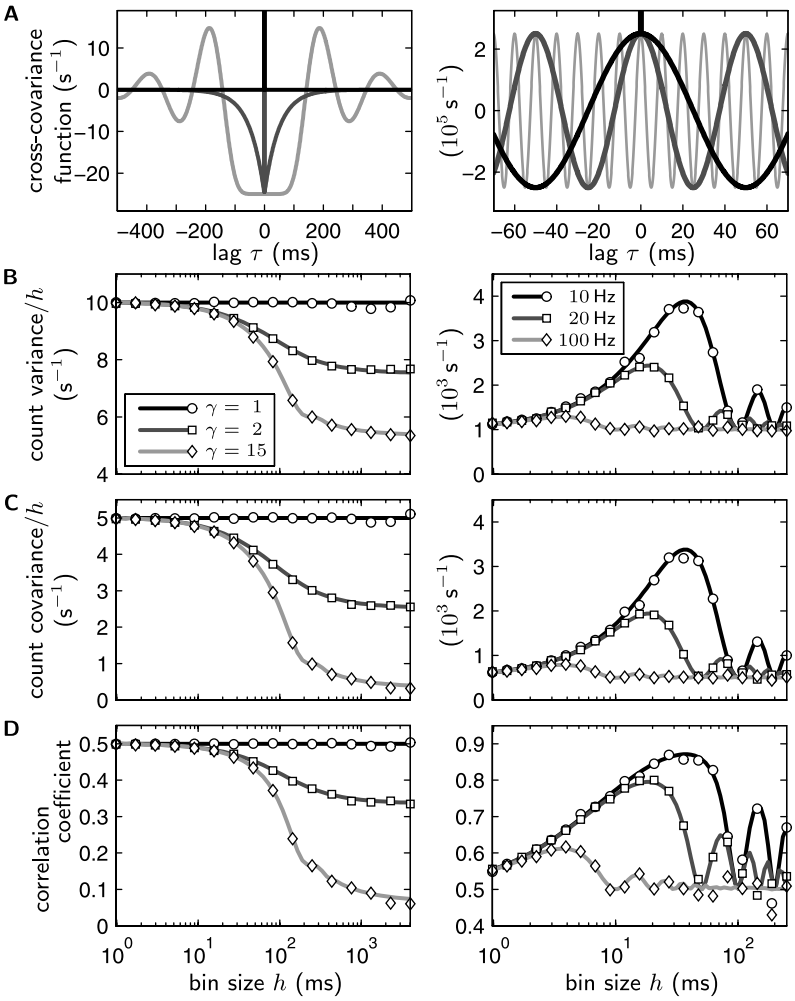

Fig. 6.3 Bin-size dependence of spike-count correlations (analytical results: *curves*, simulation: *symbols*) in the simple common-input model. *Left column*: common Gamma source of orders $\gamma = 1$ (*black, circles*), $\gamma = 2$ (*dark gray, squares*), and $\gamma = 15$ (*light gray, diamonds*) with common-input strength $\alpha = 0.5$ and total firing rate $\nu = 10\ \mathrm{s}^{-1}$ (40 trials, simulation time $T = 1000$ s). *Right column*: common Poissonian source with sinusoidal rate function of frequency $f_0 = 10$ Hz (*black, circles*), $f_0 = 20$ Hz (*dark gray, squares*), and $f_0 = 100$ Hz (*light gray, diamonds*; common-input strength $\alpha = 0.5$, total firing rate $\nu = 1000\ \mathrm{s}^{-1}$, 10 trials, simulation time $T = 2$ s). (**A**) Spike-train cross-covariance functions $\tilde{\psi}_{ij}(\tau)$ (zero-lag peaks truncated). (**B**) Normalized spike-count variances $\tilde{c}^h_{ii}/h$, (**C**) covariances $\tilde{c}^h_{ij}/h$, and (**D**) correlation coefficients r^h_{ij} as functions of the bin size h (log-scaled abscissa). Figure modified from Tetzlaff et al. (2008)

Across trials, the firing rate profiles $\nu^k_c(t)$ change randomly. In the following, all expectation values $\mathrm{E}[\cdot]$ therefore have to be interpreted as expectations over realiza-

tions l and over trials k, i.e., $\mathrm{E}[\cdot] = \mathrm{E}_k[\mathrm{E}_l[\cdot]]$. After averaging over k, the covariance function reads

$$\tilde{\psi}_{cc}(t,t') = \mathrm{E}_k\left[\nu_c^k(t)\right]\delta(t-t') + \tilde{\gamma}_{cc}(t,t'), \tag{6.35}$$

with

$$\tilde{\gamma}_{cc}(t,t') = \mathrm{E}_k\left[\nu_c^k(t)\nu_c^k(t')\right] - \mathrm{E}_k\left[\nu_c^k(t)\right]\mathrm{E}_k\left[\nu_c^k(t')\right] \tag{6.36}$$

being the autocovariance function of the firing rate. If the rate functions were identical in each trial, averaging over k would not have any effect; the two terms in (6.36) would cancel out, and in (6.35) only the delta peak would remain. In all other cases, however, the rate covariance function $\tilde{\gamma}_{cc}(t,t')$ determines the structure of the spike-train covariance function.

In order to study the interplay between nonstationarity in time and nonstationarity across trials, it is sufficient to restrict the discussion to processes where the average firing rate is constant in time,

$$\mathrm{E}_k\left[\nu_c^k(t)\right] =: \nu_c. \tag{6.37}$$

At first sight, this seems to imply simultaneous stationarity of the spike-generating process in time and across trials. Thus, for any given trial k, we would expect the firing rate describing the process to be constant in time, $\nu_c^k(t) = \nu_c^k$, and for any given point in time t, we would expect the firing rate to be constant across trials, $\nu_c^k(t) = \nu_c(t)$. In fact, however, stationarity in time only follows if, in addition to (6.37), the system is stationary across trials. Consider, as an example, a process with sinusoidal rate function and stationary frequency $f_0 = \omega_0/2\pi$, but phase $\phi_k \in [0, 2\pi)$ uniformly distributed across trials:

$$\nu_c^k(t) = \nu_c\left[1 + \cos(\omega_0 t + \phi_k)\right]. \tag{6.38}$$

Averaging over the ensemble of trials (k) results in a constant value $\nu_c = \mathrm{E}_k[\nu_c^k(t)]$—despite the nonstationary firing rate driving spike generation in each individual trial. Thus, by construction of the process, the trial averaged firing rate does not expose any underlying nonstationarity. In contrast, the trial averaged autocovariance function does. The rate covariance function is given by

$$\tilde{\gamma}_{cc}(t,t') = \frac{1}{2}\nu_c^2\cos\left(\omega_0\left[t-t'\right]\right) \tag{6.39}$$

and depends only on the time difference $\tau = t - t'$. Hence, we obtain for the spike-train covariance function of the common source,

$$\tilde{\psi}_{cc}(\tau) = \nu_c\delta(\tau) + \frac{1}{2}\nu_c^2\cos(\omega_0\tau) \tag{6.40}$$

(see Fig. 6.3A, right). The fact that the two-dimensional correlation function $\tilde{\psi}_{cc}(t,t')$ is time invariant allows us to compute the one-dimensional shot-noise correlations using (6.6) in the subsequent sections.

6.3.3 Bin-Size and Autocorrelation Dependence of Spike-Count Correlations

With a generic correlation model at hand, we can now discuss how measured spike-count correlations depend on the choice of the bin size and on the marginal spike-train statistics parameterized by the γ-order in the first example (Sect. 6.3.2.1) and by the oscillation frequency ω_0 in the Poisson example (Sect. 6.3.2.2).

With (6.12) and the spike-train autocovariance function $\tilde{\psi}_{cc}(\tau)$ of the common sources given by (6.33) and (6.40), we obtain for the spike-count covariance of the Gamma example,

$$\tilde{c}_{ij}^h = \nu_c h + 2 \sum_{l=1}^{\gamma-1} \frac{A_l}{B_l} \big(h - B_l^{-1} \big[1 - \mathrm{e}^{-B_l h} \big] \big) \tag{6.41}$$

with $A_l = \nu_c^2 \mathrm{e}^{2\pi i l/\gamma}$ and $B_l = \gamma \nu_c (1 - \mathrm{e}^{2\pi i l/\gamma})$, and for the oscillating Poisson example,

$$\tilde{c}_{ij}^h = \nu_c h - \frac{\nu_c^2}{\omega_0^2} \big(\cos(\omega_0 h) - 1 \big). \tag{6.42}$$

The variances are given by $\tilde{c}_{ii}^h = \nu_d h + \tilde{c}_{ij}^h$.

The bin-size dependence of spike-count variance $\tilde{c}_{ii}^h$, covariance $\tilde{c}_{ij}^h$, and correlation coefficient $r_{ij}^h = \tilde{c}_{ij}^h / \tilde{c}_{ii}^h$ determined by (6.41) and (6.42) is shown in Fig. 6.3B, C, and D for different γ-orders (left column) and oscillation frequencies (right column), respectively. Figure 6.3A depicts the corresponding spike-train cross-covariance functions (6.33) (left) and (6.40) (right). Note, that these cross-covariance functions $\tilde{\psi}_{ij}(\tau)$ are identical to the autocovariance functions $\tilde{\psi}_{cc}(\tau)$ of the common sources (see (6.26)). Only at short time scales, the count variances and covariances do not deviate from the Poisson case. In the Gamma example (Fig. 6.3, left), "short" means short compared to the mean inter-spike interval $1/\nu_c$ of the common source (here 200 ms). In the Poisson example (Fig. 6.3, right), the bin size must be considerably smaller than the oscillation period $2\pi/\omega_0$. The count variances $\tilde{c}_{ii}^h$ and covariances $\tilde{c}_{ij}^h$ exhibit a nontrivial dependence on the bin size h. The normalization of the covariance by the variances in the correlation coefficient r_{ij}^h does not remove this dependence. In the Gamma example, the deviations from the Poisson case become more pronounced with increasing γ-order and bin size. In the Poisson example, the count variances and covariances oscillate as functions of the bin size.

By comparing the results for the two examples shown in Fig. 6.3 with the case where the common process is a stationary Poisson process, we arrive at the following conclusion. For a given bin size h, spike-count variance, covariance, and correlation coefficient are generally decreased if the common process is a Gamma process; they are increased if the common process is a Poisson process with sinusoidal rate profile. To gain an intuitive understanding, consider the number of spikes $x^h(t)$ in a certain time window $[t, t+h)$ for a given realization of the point process.

In a Gamma process, the probability of spike generation immediately after a spike is reduced (reflected in the correlation functions shown in Fig. 6.3A, left). Hence, the number of possible spikes in a time window that is small compared to the mean inter-spike interval is decreased (compared to a stationary Poisson process with the same rate). Conversely, for an oscillatory Poisson process, the spiking probability after spike emission is enhanced for time intervals that are small compared to the oscillation period (see Fig. 6.3A, right). Therefore, the spike count increases. Recall that this does not affect the mean spike count obtained by averaging over realizations of the point process. It does affect, however, the expectation of the square of the spike count and therefore the spike-count variance. Studying the variances, we realize that in the common-input scenario each process results from a superposition of a common Gamma or inhomogeneous Poisson process, respectively, and a stationary Poissonian background process. In contrast, the spike-count covariances reflect the variances of the common process only. Thus, the normalization of the covariance by the variances does not remove the bin-size dependence.

A dependence of the count variance $\tilde{c}_{ii}^h$ and the Fano factor $F = \tilde{c}_{ii}^h / \nu_i h$ (Fano 1947) on the bin size for (non-Poissonian) renewal processes has already been reported by Rotter et al. (2005). In particular, the authors point out that the Fano factor of a Gamma process is biased towards 1 for small bin sizes (see Chap. 3). Our considerations demonstrate that this is the case for all point processes with a finite (or zero) interval density $p_1(\tau)$ (inter-spike interval distribution) at small time lags τ. The autocovariance function (Cox 1962; Perkel et al. 1967a)

$$\tilde{\psi}_{ii}(\tau) = \nu_i \left(\delta(\tau) + \sum_{k=1}^{\infty} p_k\big(|\tau|\big) \right) - \nu_i^2$$

of such a process is in the vicinity of $\tau = 0$ always dominated by the delta-peak with amplitude ν_i. According to (6.12), the count variance $\tilde{c}_{ii}^h$ therefore approaches the count mean $\nu_i h$ for small bin sizes h, resulting in a Fano factor close to one. In other words, at time scales which are small compared to the mean inter-spike interval a point process is not distinguishable from a Poisson process.

Figure 6.3 does not only illustrate that measured spike-count correlations depend on the choice of the bin size h but also demonstrates that the variance, the covariance, and the correlation coefficient are determined by the statistics of the common source. In the Gamma example (left column in Fig. 6.3), the correlation coefficient decreases as the γ-order increases (Fig. 6.3D, left). This dependence on the γ-order is made explicit in Fig. 6.4A. For common sources with oscillating Poisson statistics (right column in Fig. 6.3), the correlation coefficient decreases with increasing oscillation frequency f_0 (Fig. 6.3D, right).

6.3.4 Coherence

The dependence of the correlation coefficient on the filter properties and on the structure of the spike-train correlation functions (Fig. 6.3D, Fig. 6.4A) renders its

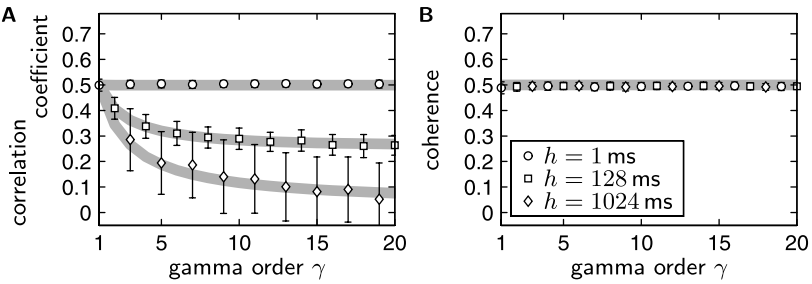

Fig. 6.4 Dependence of spike-count correlations on the order of a common Gamma source (common-input model). (**A**) Measured spike-count correlation coefficients r_{ij}^h and (**B**) averaged high-frequency (>10 Hz) coherences κ for three different bin sizes $h = 1$ ms (*circles*), $h = 128$ ms (*squares*), and $h = 1024$ ms (*diamonds*) as functions of the order γ of the common Gamma source (common-input strength $\alpha = 0.5$, total firing rate $\nu = 10\ \mathrm{s}^{-1}$). Symbols represent simulation results obtained from averaging over 1000 trials (simulation time per trial $T = 4.096$ s). *Error bars* indicate standard deviations resulting from 100 repetitions of the experiment, each with 10 trials (*error bars* in (**A**) for $h = 1$ ms and in (**B**) are too small to be visible). *Thick gray curves* in (**A**) show analytical results for the correlation coefficient. In (**B**), they depict the common-input strength $\alpha = 0.5$. Figure modified from Tetzlaff et al. (2008)

interpretation difficult and limits its usefulness for the comparison of data from different preparations and laboratories. In Sect. 6.2 we remarked that, in contrast to the correlation coefficient, the coherence $\kappa(\omega)$ (6.18) is independent of a joint linear filter kernel. This is illustrated for the Gamma example in Fig. 6.5C. However, the coherence still depends on the shape of the auto- and cross-spectra of the input spike trains. As we assumed mutual independence between the common and the disjoint input processes (6.24), the coherence is given by

$$\kappa(\omega) = \frac{\tilde{\Psi}_{cc}(\omega)}{\tilde{\Psi}_{dd}(\omega) + \tilde{\Psi}_{cc}(\omega)}, \tag{6.43}$$

where $\tilde{\Psi}_{dd}(\omega) := \tilde{\Psi}_{d_i d_i}(\omega) = \tilde{\Psi}_{d_j d_j}(\omega)$. For a large class of point processes, the power-spectrum becomes constant at high frequencies with its amplitude approaching the numerical value of the firing rate (Jarvis and Mitra 2001; Halliday 2000)

$$\begin{aligned} \lim_{\omega\to\infty} \tilde{\Psi}_{cc}(\omega) &= \nu_c, \\ \lim_{\omega\to\infty} \tilde{\Psi}_{dd}(\omega) &= \nu_d. \end{aligned} \tag{6.44}$$

This is trivial for the disjoint Poissonian processes with $\tilde{\Psi}_{dd}(\omega) = \nu_d$ $(\forall\omega)$. But also the Gamma process exhibits this property: both the real and imaginary parts of (6.32) approach zero in the limit $\omega \to \infty$. With (6.31), the first equation in (6.44) follows immediately. If the common source is an oscillating Poisson process, its power-spectrum, i.e., the Fourier transform of (6.40), is identical for all frequencies ω except ω_0:

$$\tilde{\Psi}_{cc}(\omega) = \nu_c \quad (\forall\omega \neq \omega_0). \tag{6.45}$$

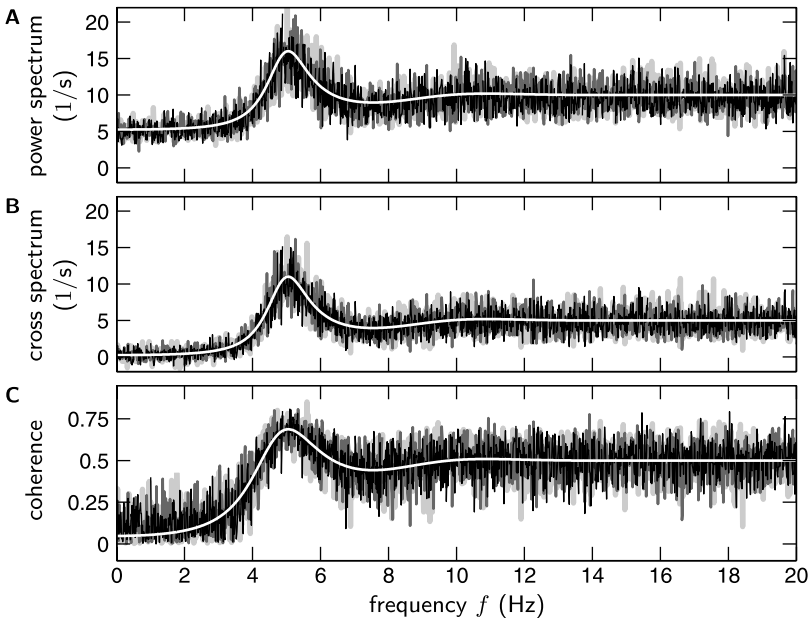

Fig. 6.5 Extraction of correlation from spike trains with structured autocorrelation generated by the simple common-input model (see Fig. 6.2). (**A**) Spike-train power- $\tilde{\Psi}_{ii}(\omega)$ and (**B**) cross-spectra $\tilde{\Psi}_{ij}(\omega)$ estimated from measured spike-count spectra $\tilde{C}_{ii/ij}(\omega)$ (total rate $\nu = 10\,\mathrm{s}^{-1}$, common Gamma source with $\gamma = 20$, common-input strength $\alpha = 0.5$, simulation time $T = 65.536$ s, averaged over 30 trials). The *panels* show data for three bin sizes superimposed (*light gray* to *black* and with decreasing thickness: $h = 1$, 128, and 1024 ms), no further smoothing of graphs. The *white curves* show analytical results. (**C**) Coherence $\kappa(\omega) = |\tilde{\Psi}_{ij}(\omega)|/\tilde{\Psi}_{ii}(\omega) = |\tilde{C}_{ij}(\omega)|/\tilde{C}_{ii}(\omega)$. Figure modified from Tetzlaff et al. (2008)

In general, (6.44) holds for all point processes with interval densities $p_k(\tau)$ absent of high-frequency components. This becomes apparent by inspection of (6.29): if $\mathfrak{F}[p_k(\tau)](\omega) = P_k(\omega)$ vanishes in the limit $\omega \to \infty$ for all orders k, the Fourier transform of the autocorrelation (6.29)—the power-spectrum—saturates at a constant level which is determined by the firing rate. Typically, natural point processes fulfill this condition. To mention an exception, consider a regular process with a constant inter-spike interval T, i.e., $p_1(\tau) = \delta(\tau - T)$.

Given the property (6.44) and the assumptions of Sect. 6.3.2, the coherence at high frequencies recaptures the common-input strength α:

$$\lim_{\omega\to\infty} \kappa(\omega) = \frac{\nu_c}{\nu_d + \nu_c} = \alpha. \tag{6.46}$$

Figure 6.5 compares the power-spectra $\tilde{\Psi}_{ii}(\omega) = \tilde{\Psi}_{cc}(\omega) + \tilde{\Psi}_{dd}(\omega)$ (A), the cross-spectra $\tilde{\Psi}_{ij}(\omega) = \tilde{\Psi}_{cc}(\omega)$ (B), and the coherences $\kappa(\omega)$ (C) for the Gamma example obtained from simulations with analytical expressions. The spike-train spectrum $\tilde{\Psi}_{ii/ij}(\omega)$ based on simulated spike trains is estimated by dividing the spike-count spectrum $\tilde{C}_{ii/ij}(\omega)$ by the spectrum $\Phi(\omega)$ of the spike-count filter (excluding fre-

quencies with $\Phi(\omega) = 0$). The figure compares results for three different bin sizes. The coherences in Fig. 6.5C are computed directly from the spike-count spectra. Both the spectra and the coherence become constant at frequencies above 10 Hz. The total rate $\nu = 10\ \mathrm{s}^{-1}$, the rate $\nu_c = 5\ \mathrm{s}^{-1}$ of the common process, and the common-input strength $\alpha = 0.5$ can be clearly identified as the limiting values of the three panels.

Figure 6.4B summarizes the results for measured high-frequency coherences for different bin sizes h and γ-orders. Here, the coherences are averaged over frequencies > 10 Hz. In contrast to the correlation coefficient (Fig. 6.4A), the high-frequency coherence depends neither on the bin size nor on the order of the common Gamma process.

The measurement of the common-input strength α is not limited to the case where the disjoint and the common processes are mutually uncorrelated. In the presence of mutual correlations between the source processes, the common-input strength can still be extracted from the coherence—provided that the spike-train correlation coefficient is known (see Appendix 6.5.1).

6.3.5 Jittered Correlations

In general (Sect. 6.2), the correlation coefficient observed for two shot-noise signals results from the interaction between the correlation functions of the underlying spike trains and the filters determining the shot-noise. In the foregoing, we have considered the case where the structure of the correlation functions is determined exclusively by the spike-train autocorrelations. However, even in the simplest case where the latter are stationary Poisson processes with delta-shaped autocorrelations, the resulting cross-correlations can be structured. If, for example, a presynaptic neuron consistently contributes spikes to the two input trains with different but static delays, the cross-correlation exhibits an off-center delta-peak. Assuming that joint contributions come from many different sources and that there is no bias in the distribution of delays, a temporally extended and centered peak results. The same would be true for sources which deliver spikes to the two input trains with dynamical delays described by an identical mean delay and nonvanishing uncorrelated temporal jitter.

Consider two processes $\xi_i(t)$ and $\xi_j(t)$ constructed as explained in Fig. 6.2. Assume that both the common $\xi_c(t)$ and the two disjoint sources $\xi_{d_{i/j}}(t)$ are homogeneous Poisson processes. If spikes of the common process $\xi_c(t)$ are precisely copied into both processes $\xi_i(t)$ and $\xi_j(t)$, the resulting cross-correlation function is delta-shaped. However, if one of the two processes receives a jittered version of the common process

$$\xi_c^{\mathrm{jit}}(t) = \sum_k \delta(t - t_k + \varepsilon_k) \tag{6.47}$$

such that each spike at time t_k is shifted by a random number ε_k, the shape of the cross-covariance function reflects the probability density function (pdf) $p_{\mathrm{jit}}(\varepsilon)$ of ε_k, i.e.,

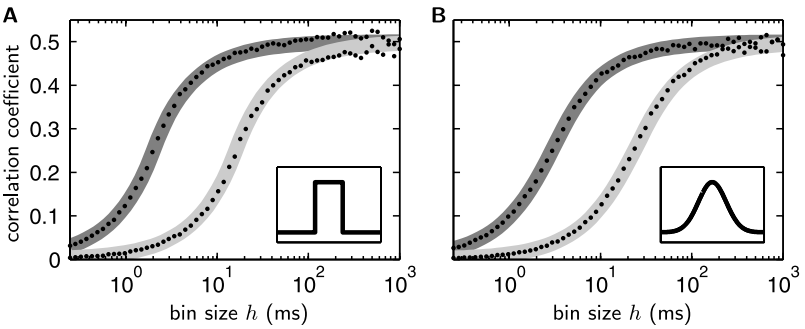

Fig. 6.6 Trivial bin-size dependence of the spike-count correlation coefficient for temporally extended cross-correlations. Two correlated Poisson spike trains ($\nu = 10\ \mathrm{s}^{-1}$, $\alpha = 0.5$) of duration $T = 8$ s are constructed as described in Fig. 6.2. In addition, in one of the spike trains the spikes from the common source are jittered by a random offset drawn from a symmetric distribution (*dark gray curves*: standard deviation $\sigma = 2$ ms; *light gray curves*: $\sigma = 16$ ms) with zero mean: (**A**) rectangular, (**B**) Gaussian distribution (see *insets*). In *both panels*, the spike-count correlation coefficient $\tilde{c}_{ij}/(\nu h)$ is shown (*gray curves*: analytical results, *dots*: simulations, averages over 200 trials) as a function of the bin size h (log-scaled abscissa). Figure modified from Tetzlaff et al. (2008)

$$\tilde{\psi}_{ij}(\tau) = \nu_c\, p_{\mathrm{jit}}(\tau). \tag{6.48}$$

Here, ν_c denotes the firing rate of the common source $\xi_c(t)$ which is of course not affected by the jittering procedure. Since we assumed that all involved processes are Poissonian, the autocovariance functions remain unaffected, too:

$$\tilde{\psi}_{ii/jj}(\tau) = \nu\delta(\tau). \tag{6.49}$$

According to (6.13), the count correlation coefficient reads

$$r_{ij}^h = \frac{\nu_c}{\nu h}\int_{-h}^{h} \mathrm{d}\tau\, \big(h - |\tau|\big)\, p_{\mathrm{jit}}(\tau). \tag{6.50}$$

Its bin-size dependence is illustrated in Fig. 6.6 for a rectangular (A) and a Gaussian (B) delay distribution $p_{\mathrm{jit}}(\tau)$ (see Appendix 6.5.2).

With $P_{\mathrm{jit}}(\omega)$ being the Fourier transform of $p_{\mathrm{jit}}(\tau)$, the coherence reads

$$\kappa(\omega) = \frac{\nu_c}{\nu}\big|P_{\mathrm{jit}}(\omega)\big|. \tag{6.51}$$

As $p_{\mathrm{jit}}(\tau)$ is a probability density with $P_{\mathrm{jit}}(0) = \int \mathrm{d}\tau\, p_{\mathrm{jit}}(\tau) = 1$, the coherence at frequency $\omega = 0$ recaptures the common-input strength α:

$$\kappa(0) = \frac{\nu_c}{\nu} = \alpha. \tag{6.52}$$

In contrast to the correlation coefficient, $\kappa(0)$ does not depend on the filter properties. It recaptures the strength of the common input α, however, only if the spike trains are stationary Poisson processes.

In a realistic setting, we must expect that both the spike-train statistics and the presence of (distributions of) delays will shape the resulting correlation functions. In this case, the coherence in our model reads

$$\kappa(\omega) = \frac{|P_{\text{jit}}(\omega)\tilde{\Psi}_{cc}(\omega)|}{\tilde{\Psi}_{cc}(\omega) + \tilde{\Psi}_{dd}(\omega)}. \tag{6.53}$$

In Sect. 6.3.4 we argued that in the neuroscientific context many processes exhibit power-spectra which approach a constant value, the firing rate, in the $\omega \to \infty$ limit. In the absence of temporal jitter this enables us to determine the common-input strength α from the coherence $\kappa(\omega)$ at large frequencies. In the presence of temporal jitter this approach is problematic if $P_{\text{jit}}(\omega)$ decays to zero at large frequencies, which seems natural for non-delta-type jitter distributions. According to (6.53), in this case the high-frequency coherence will become zero, too. This is, of course, not surprising because if the spread of $p_{\text{jit}}(\tau)$ is large, the spike trains become de facto uncorrelated on short time scale. Hope to recover α rests on the assumption that the temporal spread of $p_{\text{jit}}(\tau)$ is much smaller than the time scale of the structure of the autocovariance function $\tilde{\psi}_{cc}(\tau)$ such that $\tilde{\Psi}_{cc}(\omega)$ falls off faster than $P_{\text{jit}}(\omega)$.

6.4 Summary

For second-order stationary processes, the one-dimensional shot-noise correlation functions result from the spike-train correlation functions by convolution with the deterministic filter (auto)correlation. In consequence, standard second-order statistical measures like the variance and the covariance generally depend on the filter properties in a nontrivial way. The normalization of the covariance by the geometric mean of the variances does not compensate for this in general: the shot-noise correlation coefficient depends on the filter kernel, even under optimal conditions like stationarity across trials and time. In particular, the spike-count correlation coefficient depends on the choice of the bin size. Only for the highly artificial case of Poisson point processes with delta-shaped auto- and cross-correlation functions, the filter dependence of the correlation coefficient disappears.

The coherence, in contrast, is filter independent if each signal can be described as a simple shot-noise process arising from a linear convolution of a single (compound) spike-train with some filter kernel (as is the case for spike-count signals). Therefore, coherence measurements often constitute a less ambiguous quantification of neuronal interactions. In this light it is not surprising that the literature frequently refers to the zero-frequency coherence, i.e., the normalized cross-correlation area, also as the "correlation coefficient". Although this definition differs from that of standard textbooks (e.g., Hollander and Wolfe 1999; Feller 1971), it has the advantage that this measure is not sensitive to filtering by a joint linear kernel.

Both the correlation coefficient and the coherence generally depend not only on the joint spike-train statistics but also on the marginal second-order statistics of

the individual point processes. It is therefore questionable whether these quantities qualify as appropriate measures for the co-relation between two sources. This problem is difficult to address in a general context. We therefore focused on a specific type of correlations caused by common input. The relative *common-input strength* $\alpha = \nu_c/\nu$, i.e. the ratio between the compound firing rate ν_c of the common sources and the total rate ν, is a well-defined quantity which moreover corresponds to the (functional) connectivity of an underlying network (see Tetzlaff et al. 2008). We showed that the high-frequency coherence can—under simplifying assumptions—provide a direct measure of the common-input strength for a large class of point processes.

A drawback of using the coherence as a correlation measure is that it always requires some sort of time averaging (Fourier integral). Thus, time-resolved coherences can be computed only with a finite temporal resolution. As a result, correlations in the time domain may still be more appropriate (e.g., the joint-PSTH; see Aertsen et al. 1989) if a high temporal resolution is desired.

Spike train correlation functions and the corresponding cross-spectra are generally structured for various reasons. Depending on the measurement time scale, these correlations can be observed only in limited frequency bands. In most cases, this is a minor restriction for the experimenter measuring correlations between spike trains (spike counts) of individual neurons, because the average inter-spike interval of single-unit recordings typically exceeds the bin size used to compute spike counts (Aertsen et al. 1989: $h = 1.5$–50 ms, Vaadia et al. 1995: 30–70 ms, Lampl et al. 1999: 1 ms, Sakurai and Takahashi 2006: 0.1–1 ms). There are, however, examples in the literature where the bin size is much larger than the mean inter-spike interval (Zohary et al. 1994: $h = 2$ s, firing rates up to 50 s^{-1}). According to our results in Sects. 6.2 and 6.3.3, one can expect that the resulting spike-count correlation coefficients in these cases depend on the marginal spike-train statistics.

Acknowledgements We acknowledge partially support by the Research Council of Norway (eVITA [eNEURO], Notur), the Helmholtz Alliance on Systems Biology, the Next-Generation Supercomputer Project of MEXT, Japan, EU Grant 15879 (FACETS), DIP F1.2, and BMBF Grant 01GQ0420 to BCCN Freiburg.

Appendix

6.5.1 *Estimation of the Common-Input Strength for Correlated Spike Trains*

In Sect. 6.3.2 we assumed that common and disjoint processes are mutually uncorrelated. However, the results of Sect. 6.3.4 can be generalized to the case of correlated spiking, i.e., for $\tilde{\psi}_{pq}(\tau) \neq 0$ $(p, q \in \{c, d_i, d_j\})$. For Poisson processes with delta-type correlations, the covariance functions reads

$$\tilde{\psi}_{pq}(\tau) = r_{pq}\sqrt{\nu_p \nu_q}\delta(\tau). \tag{6.54}$$

Here, r_{pq} denotes the pairwise correlation coefficient between the processes $\xi_p(t)$ and $\xi_q(t)$. The corresponding spectra are constant:

$$\tilde{\Psi}_{pq}(\omega) = r_{pq}\sqrt{\nu_p \nu_q}. \tag{6.55}$$

We can generalize this to a broader class of processes for which (6.55) still holds for large frequencies, i.e.,

$$\lim_{\omega\to\infty} \tilde{\Psi}_{pq}(\omega) = r_{pq}\sqrt{\nu_p \nu_q}. \tag{6.56}$$

Similar to Sect. 6.3.1, let us assume that the disjoint processes have identical autocorrelations ($\tilde{\psi}_{d_i d_i}(\tau) = \tilde{\psi}_{d_j d_j}(\tau)$) and therefore identical rates ν_d. For simplicity, we further assume that all correlation coefficients are identical (i.e., $r = r_{pq}$ $\forall\{p, q\}$). In this case, the high-frequency coherence between the input signals

$$\xi_{i/j}(t) = \xi_c(t) + \xi_{d_{i/j}}(t) \tag{6.57}$$

becomes

$$\begin{aligned}\lim_{\omega\to\infty} \kappa(\omega) &= \lim_{\omega\to\infty} \frac{\tilde{\Psi}_{cc}(\omega) + \tilde{\Psi}_{d_i d_j}(\omega) + \tilde{\Psi}_{c d_j}(\omega) + \tilde{\Psi}_{d_i c}(\omega)}{\tilde{\Psi}_{cc}(\omega) + \tilde{\Psi}_{d_i d_i}(\omega) + \tilde{\Psi}_{c d_j}(\omega) + \tilde{\Psi}_{d_i c}(\omega)} \\ &= \frac{\nu_c + r\nu_d + 2r\sqrt{\nu_c \nu_d}}{\nu_c + \nu_d + 2r\sqrt{\nu_c \nu_d}}.\end{aligned} \tag{6.58}$$

With $\nu_c = \alpha\nu$ and $\nu_d = (1-\alpha)\nu$, this can be written as

$$\lim_{\omega\to\infty} \kappa(\omega) = \alpha \frac{1 + r\alpha^{-1}(1-\alpha) + 2r\sqrt{\alpha^{-1}(1-\alpha)}}{1 + 2r\sqrt{\alpha(1-\alpha)}}. \tag{6.59}$$

Thus, by solving (6.59) for α, the common-input strength can be estimated from the high-frequency coherence if the spike correlation coefficient r is known. Note that, according to (6.56), r can be determined by measuring the high-frequency coherences of the spike signals, i.e., $\lim_{\omega\to\infty} |\tilde{\Psi}_{pq}(\omega)|/\sqrt{\tilde{\Psi}_{pp}(\omega)\tilde{\Psi}_{qq}(\omega)}$.

6.5.2 *Count Covariances for Jittered Correlations*

Here we derive the spike-count correlation coefficient r_{ij} for two Poissonian spike trains with rectangular and Gaussian cross-covariance function. The results are visualized in Fig. 6.6.

6.5.2.1 Rectangular Cross-Correlations

For a rectangular covariance function

$$\tilde{\psi}_{ij}(\tau)(\tau) = \begin{cases} \frac{\nu_c}{2\sigma}, & -\sigma \le \tau \le \sigma, \\ 0, & \text{else}, \end{cases} \tag{6.60}$$

the count covariance is according to (6.12) given by

$$\tilde{c}_{ij} = \nu_c \begin{cases} \frac{1}{2\sigma} h^2, & h \leq \sigma, \\ (h - \frac{\sigma}{2}), & h > \sigma. \end{cases} \tag{6.61}$$

After normalizing $\tilde{c}_{ij}$ by the count variance $\tilde{c}_{ii} = \nu h$ and with $\alpha = \nu_c/\nu$, the correlation coefficient reads

$$r_{ij} = \alpha \begin{cases} \frac{1}{2\sigma} h, & h \leq \sigma, \\ (1 - \frac{\sigma}{2h}), & h > \sigma. \end{cases} \tag{6.62}$$

Fig. 6.6A shows how r_{ij} depends on the bin size h for $\sigma = 2$ ms and 16 ms and $\alpha = 0.5$.

6.5.2.2 Gaussian Cross-Correlations

If the spike covariance function has a Gaussian shape, i.e.,

$$\tilde{\psi}_{ij}(\tau) = n(\tau, \sigma) := \frac{\nu_c}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\tau^2}{2\sigma^2}\right), \tag{6.63}$$

the count covariance becomes

$$\tilde{c}_{ij} = 2\nu_c \left[\frac{h}{2} \operatorname{erf}\left(\frac{h}{\sqrt{2}\sigma}\right) - \sigma^2 \big[n(0, \sigma) - n(h, \sigma)\big]\right], \tag{6.64}$$

where $\operatorname{erf}(\cdot)$ denotes the error function

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \mathrm{d}x' \exp\big(-x'^2\big). \tag{6.65}$$

Thus, the count correlation coefficient is given by

$$r_{ij} = \alpha \left[\operatorname{erf}\left(\frac{h}{\sqrt{2}\sigma}\right) - \frac{2\sigma^2}{h} \big[n(0, \sigma) - n(h, \sigma)\big]\right]. \tag{6.66}$$

Its bin-size dependence is illustrated in Fig. 6.6B ($\sigma = 2$ ms and 16 ms, $\alpha = 0.5$).

6.5.3 Notation

α	common-input strength
capital letters	Fourier transforms
$c_{ij}(t, t')$	two-dimensional (raw) auto- ($i = j$) or cross- ($i \neq j$) correlation function of shot-noise processes i and j
$\tilde{c}_{ij}(t, t')$	two-dimensional (raw) auto- ($i = j$) or cross- ($i \neq j$) covariance function of shot-noise processes i and j
$\tilde{C}_{ij}(\omega, \omega')$	two-dimensional power- ($i = j$) or cross-spectrum ($i \neq j$) of shot-noise processes i and j

$c_{ij}(\tau)$	one-dimensional (raw) auto- ($i = j$) or cross- ($i \neq j$) correlation function of shot-noise processes i and j
$\tilde{c}_{ii}$	variance of shot-noise process i
$\tilde{c}_{ij}$	covariance of shot-noise processes i and j
$\tilde{c}^h_{ii}$	spike-count variance of process i for bin size h
$\tilde{c}^h_{ij}$	spike-count covariance of processes i and j for bin size h
$\tilde{c}_{ij}(\tau)$	one-dimensional auto- ($i = j$) or cross- ($i \neq j$) covariance function of shot-noise processes i and j
$\tilde{C}_{ij}(\omega)$	power- ($i = j$) or cross-spectrum ($i \neq j$) of shot-noise processes i and j
$\mathrm{E}[\cdot]$	expectation value
f_0	oscillation frequency of rate function for inhomogeneous Poisson process
$f_i(t)$	ith filter kernel
$F_i(\omega)$	Fourier transform of ith filter kernel
$\mathfrak{F}[x(t)](\omega)$	Fourier transform of $x(t)$
$\mathfrak{F}[x(t,t')](\omega,\omega')$	two-dimensional Fourier transform of $x(t,t')$
ϕ_k	phase of rate function for oscillatory Poisson process
$\phi_{ij}(\tau)$	auto- ($f_i = f_j$) or cross- ($f_i \neq f_j$) correlation of the filter kernels $f_i(t)$, $f_j(t)$
$\Phi_{ij}(\tau)$	power- ($f_i = f_j$) or cross- ($f_i \neq f_j$) spectrum of the filter kernels $f_i(t)$, $f_j(t)$
γ	order of Gamma process
h	bin size
$\kappa_{ij}(\omega)$	coherence of processes i and j
$\kappa'_{ij}(\omega)$	complex coherence of processes i and j
ν_i	time-averaged rate of process (neuron) i
$\nu_i(t)$	instantaneous rate of process (neuron) i
ω	angular frequency
ω_0	angular oscillation frequency of rate function for inhomogeneous Poisson process
$p_1(\tau)$	inter-spike interval distribution
$P_1(\omega)$	Fourier transform of inter-spike interval distribution
$p_k(\tau)$	kth-order inter-spike interval distribution
$p_{\mathrm{jit}}(\tau)$	jitter (delay) distribution
$P_{\mathrm{jit}}(\omega)$	Fourier transform of jitter (delay) distribution
$\psi_{ij}(t,t')$	two-dimensional auto- ($i = j$) or cross- ($i \neq j$) correlation function of spike trains i and j
$\tilde{\psi}_{ij}(t,t')$	two-dimensional auto- ($i = j$) or cross- ($i \neq j$) covariance function of spike trains i and j
$\tilde{\Psi}_{ij}(\omega,\omega')$	two-dimensional power- ($i = j$) or cross-spectrum ($i \neq j$) of spike trains i and j
$\psi_{ij}(\tau)$	one-dimensional (raw) auto- ($i = j$) or cross- ($i \neq j$) correlation function of spike trains i and j

$\tilde{\psi}_{ij}(\tau)$	one-dimensional auto- ($i = j$) or cross- ($i \neq j$) covariance function of spike trains i and j
$\tilde{\Psi}_{ij}(\omega)$	one-dimensional power- ($i = j$) or cross-spectrum ($i \neq j$) of spike trains i and j
r_{ij}	correlation coefficient of processes i and j
r_{ij}^h	spike-count correlation coefficient of processes i and j
t	time
T	simulation time
τ	time lag
$x_i(t)$	ith shot-noise signal
$x_i^h(t)$	ith spike-count signal with bin size h
$\xi_i(t)$	ith spike train
$\tilde{x}_i(t)$	fluctuations of ith shot-noise signal
$\tilde{x}_i^h(t)$	fluctuations of ith spike-count signal with bin size h
$\tilde{\xi}_i(t)$	fluctuations of ith spike train
$\tilde{\cdot}$ (tilde)	centralized measures (fluctuations around mean, e.g., spike-count fluctuations) or derived quantities (e.g., spike-count covariance)
z^*	complex conjugate of z
$(f * g)(\tau)$	convolution between $f(t)$ and $g(t)$

References

Aertsen A, Gerstein G, Habib M, Palm G (1989) Dynamics of neuronal firing correlation: modulation of "effective connectivity". J Neurophysiol 61(5):900–917

Bair W, Zohary E, Newsome W (2001) Correlated firing in Macaque visual area MT: time scales and relationship to behavior. J Neurosci 21(5):1676–1697

Brown EN, Kaas RE, Mitra PP (2004) Multiple neural spike train data analysis: state-of-the-art and future challenges. Nat Neurosci 7(5):456–461

Cox DR (1962) Renewal theory. Methuen, London

Daley DJ, Vere-Jones D (2005) An introduction to the theory of point processes, vol 1: Elementary theory and methods, 2nd edn. Springer, New York

Fano U (1947) Ionization yield of radiations. II. The fluctuations of the number of ions. Phys Rev 72(1):26–29

Feller W (1971) An introduction to probability theory and its applications, vol 2, 2nd edn. Wiley, New York

Halliday DM (2000) Weak, stochastic temporal correlation of large scale synaptic input is a major determinant of neuronal bandwidth. Neural Comput 12:693–707

Hollander M, Wolfe D (1999) Nonparametric statistical methods, 2nd edn. Wiley, New York

Jarvis MR, Mitra PP (2001) Sampling properties of the spectrum and coherency of sequences of action potentials. Neural Comput 13:717–749

Kohn A, Smith MA (2005) Stimulus dependence of neuronal correlations in primary visual cortex of the Macaque. J Neurosci 25(14):3661–3673

Lampl I, Reichova I, Ferster D (1999) Synchronous membrane potential fluctuations in neurons of the cat visual cortex. Neuron 22:361–374

Logothetis NK, Pauls J, Augath M, Trinath T, Oeltermann A (2001) Neurophysiological investigation of the basis of the fMRI signal. Nature 412:150–157

Moreno-Bote R, Parga N (2006) Auto- and crosscorrelograms for the spike response of leaky integrate-and-fire neurons with slow synapses. Phys Rev Lett 96:028101

Mukamel R, Gelbard H, Arieli A, Hasson U, Fried I, Malach R (2005) Coupling between neuronal firing, field potentials, and fMRI in human auditory cortex. Science 309(5736):951–954

Papoulis A, Pillai SU (2002) Probability, random variables, and stochastic processes, 4th edn. McGraw-Hill, Boston

Perkel DH, Gerstein GL, Moore GP (1967a) Neuronal spike trains and stochastic point processes. I. The single spike train. Biophys J 7(4):391–418

Perkel DH, Gerstein GL, Moore GP (1967b) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophys J 7(4):419–440

Pipa G, Grün S, van Vreeswijk C (2010). Impact of spike-train auto-structure on probability distribution of joint-spike events. Neural Comput (under revision)

Priestley M (1983) Spectral analysis and time series, vols I and II. Academic Press, San Diego

Rotter S, Riehle A, Rodriguez Molina V, Aertsen A, Nawrot MP (2005) Different time scales of spike train variability in motor cortex. Abstract viewer and itinerary planner, no 276.7. Society for Neuroscience, Washington

Sakurai Y, Takahashi S (2006) Dynamic synchrony of firing in the monkey prefrontal cortex during working-memory tasks. J Neurosci 6(40):10141–10153

Softky WR, Koch C (1993) The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs. J Neurosci 13(1):334–350

Tetzlaff T, Rotter S, Stark E, Abeles M, Aertsen A, Diesmann M (2008) Dependence of neuronal correlations on filter characteristics and marginal spike-train statistics. Neural Comput 20(9):2133–2184

Tuckwell HC (1988) Introduction to theoretical neurobiology, vol 1. Cambridge University Press, Cambridge. 0-521-35096-4

Vaadia E, Haalman I, Abeles M, Bergman H, Prut Y, Slovin H, Aertsen A (1995) Dynamics of neuronal interactions in monkey cortex in relation to behavioural events. Nature 373(6514):515–518

Zohary E, Shadlen MN, Newsome WT (1994) Correlated neuronal discharge rate and its implications for psychophysical performance. Nature 370:140–143

Chapter 7
Spike Metrics

Jonathan D. Victor and Keith P. Purpura

Abstract Important questions in neuroscience, such as how neural activity represents the sensory world, can be framed in terms of the extent to which spike trains differ from one another. Since spike trains can be considered to be sequences of stereotyped events, it is natural to focus on ways to quantify differences between event sequences, known as spike-train metrics. We begin by defining several families of these metrics, including metrics based on spike times, on interspike intervals, and on vector-space embedding. We show how these metrics can be applied to single-neuron and multineuronal data and then describe algorithms that calculate these metrics efficiently. Finally, we discuss analytical procedures based on these metrics, including methods for quantifying variability among spike trains, for constructing perceptual spaces, for calculating information-theoretic quantities, and for identifying candidate features of neural codes.

7.1 Introduction

7.1.1 Mathematics and Laboratory Data

Mathematical analysis of laboratory data plays a crucial role in systems neuroscience. Perhaps the most fundamental reason is that often, the questions that we ask of data are abstract. A prime example is the investigation of neural coding—delineation of the relationship between stimuli, actions, and/or behavioral states, and the activity of one or more neurons.

An invited chapter for "Analysis of Parallel Spike Trains" (S. Rotter and S. Grün, eds.)

J.D. Victor (✉)
Division of Systems Neurology and Neuroscience, Department of Neurology and Neuroscience, Weill Cornell Medical College, 1300 York Avenue, New York, NY 10065, USA
e-mail: jdvicto@med.cornell.edu
url: http://vivo.cornell.edu/individual/vivo/individual6150

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_7, © Springer Science+Business Media, LLC 2010

To apply mathematical analysis to laboratory data in a principled way, a necessary first step is to choose an appropriate mathematical framework. This would be simple if laboratory data corresponded precisely to mathematical entities, but this is rarely the case (Slepian 1976). Laboratory data do not permit one to apply methods that require taking limits as time goes to infinity or to zero, and one does not have access to statistical ensembles, just individual experiments. Thus, explicitly or implicitly, the neuroscientist must identify and abstract the essential features of the data that are relevant to the problem at hand or the goal of the analysis. We therefore begin with a discussion of these considerations as they relate to the electrical activity of neurons and to the problem of neural coding.

7.1.2 Representing Spike Trains as Samples of Point Processes

The electrical activity of individual neurons can be recorded extracellularly, intracellularly, and somewhat more indirectly by optical means. The electrical activity, as often measured with micro- or macroelectrodes, consists of a combination of small voltage fluctuations (on the order of 10 mV), upon which are superimposed larger, brief "action potentials": stereotyped voltage transients of approximately 100 mV that last a fraction of a millisecond. Since action potentials propagate without loss and result in the release of neurotransmitter, they are generally considered to represent the components of a neuron's activity that are "seen" by the rest of the nervous system. Consequently, the sequences of action potentials emitted by individual neurons (i.e., "spike trains") are a natural focus for the study of brain activity at the level of cells and circuits (Segundo and Perkel 1969; Abbott 2000; Sen et al. 1996). To abstract the brief, stereotyped nature of spike trains, we choose to represent them as instances of point processes, i.e., event sequences.[1]

7.1.3 Analyzing Point Processes: The Rationale for a Metric-Space Approach

The point-process representation (see Chap. 1) has substantial implications for the choice of signal-processing strategies. Had we chosen a vectorial representation[2]

[1] Waveforms of real action potentials are not completely stereotyped, and at least some of the waveform-to-waveform differences are systematic. By representing neural activity as a point process, we make the conscious decision to ignore these differences, as is typically done both in this book and in general. However, this simplification is not as restrictive as it might at first seem. By representing action potentials as instances of point processes, we are not assuming that every action potential has the same effect on post-synaptic neurons; we are merely assuming that the differences in the effects of each action potential can be understood from their temporal pattern. This makes good biological sense, since temporal pattern (in particular, the time since the preceding spike) is a crucial factor in governing both transmitter release and the subtle variations in action potential shape.

[2] More specifically, a Hilbert space—because the dot-product is defined.

(e.g., had we represented neural activity as a continuous voltage record), certain algebraic operations would immediately be defined—the usual vector-space operations of addition, multiplication by scalars, and inner ("dot") product. These operations are the basis of signal-processing methods such as filtering, averaging, spectral estimates, and signal detection. For point processes, these operations are not applicable, and these vector-based signal-processing methods cannot be directly applied.

While this might at first seem to be a severe disadvantage of the point process framework, further consideration shows that this is not the case. The reason is that vector-based procedures require the imposition of a high degree of mathematical structure, and this structure is often not appropriate or practical. One reason for this is that in a vector space, linearity plays a fundamental role, and this is arguably at odds with the nature of neural dynamics. A second reason is that the dot-product induces an intrinsic Euclidean geometry. By representing spike trains as samples of point processes, we are driven to analytical procedures that do not require us to define vector-space operations (e.g., how two spike trains can be added). The point process perspective can thus allow the researcher to investigate a wider gamut of behavior in the spike trains.

Two examples from sensory physiology emphasize this point. Human color vision is a three-parameter space determined by the three cone absorption spectra, and thus, one might hope that a three-dimensional vector space would provide an appropriate representation for color percepts. However, experimental measurement of perceptual distances within this three-dimensional space shows that perpendiculars need not be unique (Wuerger et al. 1995)—in contrast to the requirements of a Euclidean space. In olfaction, the situation is far more complex. Olfactory perceptual space may not even have a well-defined dimension, mixing of odorants need not lead to intermediate percepts, and gross violations of linearity are present (Hopfield 1995). Thus, the Euclidean geometry implied by vector spaces may be too confining to support a correspondence between neural activity and sensory perception.

To broaden our scope, we consider analytic procedures in which relationships between pairs of spike trains play a central role, but we do not require that we know how to "add" spike trains or to multiply them by scalars. Instead, our basic operation is a way to compute a distance (i.e., a measure of dissimilarity) between two spike trains, $d(A, B)$. In keeping with the philosophy of limiting our assumptions, we only require that the distances $d(A, B)$ are a "metric" (defined below). That is, spike trains are considered to be points in a metric space, rather than a vector space. While distances derived from vector spaces are necessarily metrics, the distances derived from metric spaces typically cannot be mimicked by a vector space. Vector space distances are unchanged by linear transformation; no corresponding property holds for metrics in general. Typical metric spaces, including the several of those we consider below, do not have Euclidean geometry (Aronov and Victor 2004). Thus, metric spaces (and the distances that define them) are much more general than vector spaces.

The reader might wonder about mathematical frameworks that are even more general. For example, the notion of a "topology" on the set of spike trains is even more general than that of a metric (see Singh et al. 2008 for a recent application of

this notion to spike-train analysis). A topology merely requires that we define relationships between spike trains via a system of "open sets", while a metric requires that we quantify distances (and these distances in turn define the open sets). Fundamentally, there is no a priori reason to exclude nonmetrizable topologies, though such spaces are typically quite strange and nonintuitive. Conversely, the notion of "distance" is widely applicable to our current understanding of perception and behavior, and can be readily quantified, at least in principle, for example, as the number of "just noticeable differences" between two stimuli. Moreover, a compilation of the distances between all pairs of points describes the intrinsic geometry of a space. Thus, the metric-space framework is capable of capturing the essence of what one hopes to account for (the intrinsic structure of the space represented by neural activity) and carries with it only minimal restrictions. Finally, we point out that under some circumstances, perceptual comparisons (Tversky and Gati 1982; Tversky 1977; Maloney and Yang 2003) and other components of cognition and learning may be nonmetric in that they violate either the postulate of symmetry or the triangle inequality. Generalizations of metrics that accommodate this behavior are straightforward; the reader is referred to (Victor et al. 2007) for further details.

We emphasize that we are not suggesting a single best way to define distances in neurophysiological data. Rather, we present the metric approach (especially as it applies to neural coding) as a general strategy to formalize biologically motivated hypotheses concerning the meaningful features of spike trains and to determine whether these hypotheses are supported by observed neural activity associated with sensory and motor events.

7.1.4 Plan for this Chapter

We begin with some formal preliminaries. We then define several families of metrics for spike trains, first considering metrics applicable to single-neuron responses and then metrics applicable to multineuronal activity. We then describe several algorithms by which these metrics can be calculated. We conclude by discussing a range of analytical procedures based on these metrics that can provide insight into spike trains and their relationship to behavior. These procedures include methods of quantifying variability among spike trains, describing their relationship to perceptual spaces, and information-theoretic techniques aimed at identifying candidate features of neural codes.

7.2 Spike Train Metrics

In this section, we define several families of metrics for spike trains. We emphasize "edit-length" metrics, which are specifically applicable to event sequences and yield distances that are fundamentally distinct from vector-space distances. Following this, we consider a second class of metrics that are consequences of embedding spike trains into vector spaces.

7.2.1 Notation and Preliminaries

We represent the activity of a single neuron as a sequence of stereotyped events. Correspondingly we represent multineuronal activity as a sequence of labeled events. More formally, we represent a spike train A observed over a period $[0, T]$ by a sequence of distinct real numbers $t_1, \ldots, t_{M(A)}$, the times of occurrence of the spikes. $M(A)$ is the total number of spikes (and may be 0), and the spike times are assumed to be listed in increasing order. When considering the activity of multiple neurons, we augment this framework by adding labels $l_1, l_2, \ldots, l_{M(A)}$. The labels l_j are drawn from a set $\{1, \ldots, L\}$ of abstract tags, indicating which neuron was responsible for each spike.

A metric $d(A, B)$ is a mapping from pairs of spike trains to the nonnegative real numbers. It satisfies three properties, namely (i) $d(A, B) > 0$, with equality only when $A = B$, (ii) symmetry: $d(A, B) = d(B, A)$, and (iii) the triangle inequality, $d(A, C) \leq d(A, B) + d(B, C)$. With these three conditions, the present use of the term "metric" is consistent with the topological definition of this term (Gaal 1964) and endows the set of event sequences (the spike trains) with the properties of a topological "metric space".[3]

7.2.2 Cost-Based (Edit Length) Metrics

7.2.2.1 General Definition

A simple, intuitive strategy enables the construction of metrics that formalize a range of biologically motivated notions of similarity. The common element in these metrics is that of a set of "elementary steps" between two spike trains, each of which has an associated nonnegative cost. We require that the cost $c(X, Y)$ of an elementary step from X to Y is symmetric and that for any two spike trains A and B, it is possible to find some sequence of elementary steps that begins at A and ends at B.

Any set of elementary steps satisfying these conditions leads to a metric between spike trains: the cheapest total cost to transform A to B via elementary steps. More formally, we define

$$d(A, B) = \min\left\{ \sum_{j=0}^{n-1} c(X_j, X_{j+1}) \right\}, \tag{7.1}$$

[3] Below we will also want to consider a very simple distance that is not, strictly speaking, a metric. This is the "spike count distance" $D^{\text{spike}}[0]$. For the spike-count distance, the distance between two spike trains is given by the difference in the number of spikes they contain. Thus, we can have $D^{\text{spike}}[0](A, B) = 0$ for distinct spike trains A and B if they contain the same number of spikes. While this distance is not strictly a metric, the formal structure of a metric space still applies, because we can think of $D^{\text{spike}}[0]$ as acting on the equivalence classes of "distinguishable" spike trains, rather than on the spike trains themselves.

where $\{X_0, X_1, \ldots, X_n\}$ is a sequence of spike trains with $X_0 = A$ and $X_n = B$. Metrics defined in this fashion are guaranteed to satisfy the triangle inequality, since the cheapest path from A to C cannot be more expensive than a path that is constrained to stop at B. These cost-based distances are analogous to "edit-length" distances for symbol sequences, including those used in analysis of EEG data (Wu and Gotman 1998), and, more prominently, in comparison of genetic sequences (Sellers 1974; Needleman and Wunsch 1970). In this analogy, each of the elementary steps can be considered to be a way of "editing" the spike train. The minimum total cost of the elementary transformations, the "edit-length", quantifies the dissimilarity of the spike trains. Thus, efficient dynamic programming algorithms for comparison of genetic sequences (Sellers 1974; Needleman and Wunsch 1970) can be adapted to calculate many kinds of spike metrics (Victor and Purpura 1997, 1996). Interestingly, although these algorithms have been in use for almost 40 years for genetic sequence comparison, their use in neuroscience is much more recent.

7.2.2.2 Spike Time Metrics

As a first example of the strategy described above, we create a family of metrics that are sensitive to the timing of individual spikes (Victor and Purpura 1997, 1996). The biological motivation for these metrics is that neurons have a firing threshold. As a consequence, neurons can often be regarded as coincidence detectors (Softky and Koch 1993; Egger et al. 1999; Kuba et al. 2005; Abeles 1982): one spike arriving on a presynaptic neuron will not cause a postsynaptic neuron to fire, but a sufficient number of spikes arriving within a sufficiently narrow time window will cause the postsynaptic neuron to fire. Thus, the "meaning" of a spike train depends on the timing of its spikes.

To capture this dependence in a metric, we invoke the above machinery with two kinds of elementary steps (Fig. 7.1). The first elementary step consists of inserting or deleting a spike and is assigned a cost of 1. This rule ensures that every spike train can be transformed to any other spike train by *some* path: the path that successively deletes all spikes from one train and then successively inserts all spikes into the second train. The second elementary step defines the sensitivity to spike timing. It consists of moving a single spike, and the associated cost is proportional to the amount of time that the spike is moved. That is, if two spike trains X and Y are identical except for a single spike that occurs at t_X in X and t_Y in Y, then

$$c(X, Y) = q|t_X - t_Y|. \tag{7.2}$$

Along with (7.1), this set of elementary steps defines a spike time metric, $D^{\text{spike}}[q]$. Note that these metrics constitute a parametric family, parameterized by the value of q that specifies the cost per unit time to move a spike. Since moving a spike by an amount $\Delta T = 1/q$ has the same cost as deleting it altogether, q can be viewed as determining the relative sensitivity of the metric to spike count and spike timing.

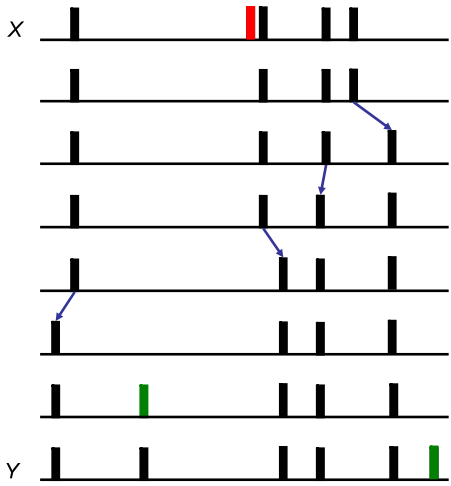

Fig. 7.1 A diagram of a sequence of elementary steps that transforms spike train X into spike train Y. Each elementary step is one of three types: deletion of a spike (deleted spike shown in *red*), insertion of a spike (inserted spike shown in *green*), or shifting a spike in time (*blue arrows*)

The character of $D^{\text{spike}}[q]$ depends strongly on q. When q is small, then the times of individual spikes have little influence on the calculated distance between spike trains. In the limit of $q = 0$, spikes can be shifted freely in time, and $D^{\text{spike}}[0](A, B) = |M(A) - M(B)|$. So $D^{\text{spike}}[0]$ corresponds to comparing spike trains in terms of their spike counts alone. As q increases, the metric $D^{\text{spike}}[q]$ becomes increasingly sensitive to spike timing: a change in the time of a spike by $1/q$ sec has the same cost as deleting the spike altogether. That is, for neurons that act like a coincidence detector with integration time (or temporal resolution) $1/q$, spike trains will have similar postsynaptic effects if they are similar in the sense quantified by $D^{\text{spike}}[q]$. Since the effective temporal resolution is typically not known in advance, it is useful to carry out analyses across a range of values of q, rather than specifying the resolution a priori.

7.2.2.3 Spike Interval Metrics

Within the same framework of (7.1), we next define a contrasting metric (Victor and Purpura 1997, 1996), one that is sensitive to the pattern of spike intervals, rather than individual times. The biological motivation is that the postsynaptic effects of a spike may depend strongly on the recent activity at that synapse (Sen et al. 1996; Dan and Poo 2004; Markram et al. 1997). For transformations dominated by this dependency, the meaning of a spike train does not depend on the spike times themselves (as captured by $D^{\text{spike}}[q]$) but rather on the sequence of intervals. To quantify this notion, we define $D^{\text{interval}}[q]$ by (7.1), along with a different set of elementary steps. For $D^{\text{interval}}[q]$, the first kind of elementary step is insertion or deletion of *an interspike interval*, at a cost of 1. The second elementary step consists of shortening or extending an existing interspike interval. The cost of this step is equal to $q\Delta T$, where ΔT is the amount of time by which the interval has been lengthened

or shortened.[4] Since changing the length of an interval changes the time of occurrence of all subsequent spikes at no additional cost, $D^{\text{interval}}[q]$ and $D^{\text{spike}}[q]$ have fundamentally different topological characteristics (Victor and Purpura 1997).

7.2.2.4 Multineuronal Cost-Based Metrics

We now consider ways of extending the above metrics to multineuronal data. While multineuronal extensions can be constructed for either of the above single-neuron metrics, the extension is more natural for $D^{\text{spike}}[q]$, so we focus on it.

Recall that our formal representation of multineuronal activity A is a sequence of $M(A)$ spike times t_j, each of which is associated with a label $l_j \in \{1, \ldots, L\}$ that indicates its neuron of origin. To extend $D^{\text{spike}}[q]$ to the multineuronal context (Aronov et al. 2003), we add another elementary step, consisting of changing the label associated with an event. The simplest way to do this is to assign the same cost, a parameter k, to any label change. These three steps, along with (7.1), define a two-parameter family of metrics, $D^{\text{spike}}[q, k]$, where for $k = 0$, the metric ignores the label associated with each event. The metrics with $k = 0$ correspond to a "summed population" of neural activity. Conversely, when $k = 2$, the impact of the label is maximal, since it costs as much to change the label as it does to delete a spike and then reinsert it with a new label. Metrics with $k = 2$ correspond to a "labeled lines" interpretation of neural activity.

7.2.2.5 Other Cost-Based Metrics

There are many directions in which the above examples can be generalized, and we present some of the more important ones here. In each case, additional degrees of freedom are added to the metrics described above. This flexibility arguably provides a closer approximation to biologic reality but often carries the penalty that a greater amount of data is required for a meaningful analysis. We also indicate which extensions are readily incorporated into the dynamic programming algorithms described in Sect. 7.2.2.6.

More Flexible Assignments of Costs to the Elementary Steps First, the cost of moving a spike by an amount of time Δt (or the amount by which an interval is changed) need not be proportional to Δt.

[4] Special consideration needs to be applied to the interval between the start of data collection and the first spike and to the interval between the last spike and the end of data collection. These are not interspike intervals, since they are bounded by the limits of data collection, rather than a second spike. It is therefore natural to take the view that each of these "intervals" could be considered an exact match to any interspike interval that is at least as long. This is equivalent to minimizing the distance to any spike train that matches the observed spike train within the data collection interval. A simpler (but arguably more arbitrary) strategy is simply to place an artificial spike at the beginning and end of data collection.

For example, $D^{\text{spike}}[q]$ can be generalized by replacing the cost defined in (7.2) by

$$c(X,Y) = Q\big(|t_X - t_Y|\big) \tag{7.3}$$

for any nondecreasing cost function $Q(|\Delta T|)$ satisfying $Q(0) = 0$ and

$$Q\big(|\Delta T| + \big|\Delta T'\big|\big) \le Q\big(|\Delta T|\big) + Q\big(\big|\Delta T'\big|\big). \tag{7.4}$$

This does not interfere with the dynamic programming algorithms, since (7.4) means that breaking down a single elementary step into two smaller ones can never decrease the total cost.

Another generalization is that the cost to move a spike could depend on its absolute time and on the distance moved. General dependences of this sort will interfere with the dynamic programming algorithm, since they allow for a minimal path in which a spike moves twice, once to enter a region of low-cost movement and once to move to its final position. But certain forms of this dependence are consistent with the dynamic programming algorithm. One such form is

$$c(X,Y) = q\big(\big|F(t_X) - F(t_Y)\big|\big), \tag{7.5}$$

where F is an increasing function. That is, $\tau = F(t)$ represents a distortion of time, and the cost to move a spike is uniform in the distorted time τ. Because of this distortion, the cost assignment of (7.5) allows for the metric to have a nonuniform dependence on spike times in the original time t. In particular, for small displacements, (7.5) can be approximated by

$$c(X,Y) \approx F'\left(\frac{t_X + t_Y}{2}\right) q\,\Delta T. \tag{7.6}$$

So for example, a decelerating function F would lead to a stronger dependence of the metric on the time of early spikes than on the time of late spikes.

Since the cost of moving a spike determined by (7.5) is related to a uniform cost through a time-distortion the corresponding metric (7.1) can be calculated from $D^{\text{spike}}[q](F(A), F(B))$, where $F(X)$ denotes a spike train derived from X by the time-distortion $\tau = F(t)$.

Other Kinds of Elementary Steps The kinds of elementary steps used to define a cost-based metric can be thought of as formalizing a hypothesis that certain aspects of spike train structure are meaningful. Above we have considered two contrasting metrics, focusing on absolute spike times and on spike intervals; these are but two of many possibilities. As another example, we consider the notion that spike times are important, but absolute time is uncertain. This arises in the analysis of spike trains associated with spontaneous motor activity (i.e., not synchronized to an external event or clock). Under these circumstances, spike trains that differ only by an overall translation in time have the same meaning. To capture this in a metric, one could augment $D^{\text{spike}}[q]$ by a step that allows an entire spike train to be translated en bloc at reduced cost per unit time, say, q', with $q' \ll q$.

For this metric, the dynamic programming algorithm for $D^{\text{spike}}[q]$ enables a calculation in polynomial time (see below). The reason is that in this metric, a minimal-cost path between two spike trains can always be found in which the en bloc translation moves at least one spike in train A into coincidence with a spike in train B. To see this, assume that no spikes are in coincidence and consider the effects of changing the size of the block movement by an infinitesimal (signed) amount dT. Say that this block movement is part of a path (7.1) in which there are $n_{A\to B}$ individual spikes that move forward in time between A and B, and $n_{B\to A}$ individual spikes that move backward. Then this infinitesimal en bloc translation incurs a net cost of $(q' - qn_{A\to B} + qn_{B\to A})\,dT$. If this number is nonzero, then either an infinitesimal translation forward or backward would reduce the total cost further. If it is zero, then an infinitesimal movement in either direction would leave the cost unchanged. In either case, further infinitesimal translations could be applied, without incurring additional cost, until at least one spike pair was in coincidence. Thus, to calculate this metric, it suffices to calculate $D^{\text{spike}}[q]$ for all $M(A)M(B)$ block translations that move any spike in train A into coincidence with any spike in train B.

Another kind of elementary step is motivated by the notion that motifs of spikes (Abeles and Prut 1996) (i.e., a set of three spikes, not necessarily contiguous, with specific interspike intervals) are meaningful. To capture this notion, one adds an elementary step that allows subsets of noncontiguous spikes to be moved as a block. One could also combine the rules of $D^{\text{spike}}[q]$ and $D^{\text{interval}}[q]$, perhaps associated with different costs. Unfortunately, it is unclear how to incorporate these generalizations into a polynomial-time algorithm, since in either case, there is the possibility that a minimal-cost sequence of transformations will require several movements of individual spike-train components.

Further Generalizations For the multineuronal metric $D^{\text{spike}}[q,k]$, the cost to change the label (neuron) associated with an event, k, is independent of the label itself. This restriction is inessential to the definition of the metric; the cost to change a label from l to l' can be an arbitrary symmetric function $k(l,l')$ of the labels. In its full generality, this extension incurs a substantial increase in the number of parameters, but this parameter explosion can be mitigated by a priori hypotheses on the form of $k(l,l')$. For each choice of values of $k(l,l')$, the time required for calculation of the metric does not increase, but the storage requirements of the "parallel" algorithm (see below) increase dramatically.

7.2.2.6 Algorithms

Identification of a minimal-cost path (7.1) might at first seem to be a daunting task. However, because the cost-based metrics $D^{\text{spike}}[q]$ and $D^{\text{interval}}[q]$ are similar to the edit-length distances used for comparison of genetic sequences, the efficient dynamic programming algorithms developed for sequence comparison (Sellers 1974; Needleman and Wunsch 1970) are readily adapted to calculate spike train metrics (Victor and Purpura 1997, 1996). For all of these algorithms, the number of computations required to calculate a distance between two responses is bounded by a

polynomial function of the number of spikes. This efficiency (in contrast to the very laborious search of all possible sequences of elementary steps) is important in that it makes the application of metric-space methods practical.

The algorithms described below have been implemented in public-domain software, available at http: http://neuroanalysis.org/toolkit/ or http://www.apst.spiketrain-analysis.org/ and described in (Goldberg et al. 2009).

The Basic Dynamic Programming Algorithm To describe this algorithm, we use a formulation (Victor et al. 2007) that, while perhaps more elaborate, provides an easier path to generalization. We focus on $D^{\text{spike}}[q]$, but the same algorithmic structure is applicable to $D^{\text{interval}}[q]$. A simpler formulation, but one that is less-readily generalized, can be found in (Victor and Purpura 1997, 1996).

We begin by deducing certain properties that any minimal-cost path (7.1) must have. First, we observe that we can always reorganize a minimal-cost path so that the first steps consist of deleting spikes that occur in train A but not in train B, the intermediate steps consist of moving some spikes that occur at times $A_{a_1}, A_{a_2}, \ldots, A_{a_R}$ in train A to "linked" spikes that occur at times $B_{b_1}, B_{b_2}, \ldots, B_{b_R}$ in train B, and the final steps consist of inserting spikes that occur in train B but not in train A (Fig. 7.1). This reorganization is always possible because any movement of a spike before it is deleted (or insertion of a spike followed by moving it) is inefficient—the spike should simply be deleted before it is moved, or inserted where it is needed. Moreover, the cost of moving a spike is independent of the times of moving any other spike, so performing the deletions first and the insertions last does not change the costs of the moves.

Therefore, a minimal-cost path can be summarized as an "alignment": a designated subset of R spikes in train A and a designated subset of R spikes in train B to which they are linked by R elementary steps that move a single spike. Given this alignment, the distance between the two spike trains can be expressed as

$$d(A,B) = \big(M(A) - R\big) + q \sum_{j=1}^{R} |A_{a_j} - B_{b_j}| + \big(M(B) - R\big), \tag{7.7}$$

where the first term is the cost of the deletions, the final term is the cost of the insertions, and the middle term is the cost of the moves, namely, the total link length multiplied by the cost, q.

Since (by hypothesis) the total cost of these steps is minimal, then the total link length $\sum_{j=1}^{R} |A_{a_j} - B_{b_j}|$ must also be minimal, among all possible links between R pairs of spikes. This in turn implies that the two sequences of spike times $A_{a_1}, A_{a_2}, \ldots, A_{a_R}$ and $B_{b_1}, B_{b_2}, \ldots, B_{b_R}$ must be monotonic. In other words, the links between spikes cannot cross, since if they did, then the total link length could be reduced by uncrossing them.

Let us now assume that we have found the minimal cost path between spike trains A and B. We now focus on the final spike in the two trains, at times $A_{M(A)}$ and $B_{M(B)}$. We use $X^{(k)}$ to denote a spike train consisting of the first k spikes in X. For example, $A^{(M(A)-1)}$ is the spike train A with the last spike deleted. If

the final spikes are connected by a link, then removing those spikes and their link must yield a minimal-cost path between the truncated spike trains. That is, if the final spikes are linked, then $d(A, B) = d(A^{(M(A)-1)}, B^{(M(B)-1)}) + q|A_{a_{M(A)}} - B_{b_{M(B)}}|$. Alternatively, if the final spikes are not connected to each other by a link, it must be that at least one of the spikes is unlinked, since otherwise, their links would cross. If the final spike in A is unlinked, then $d(A, B) = d(A^{(M(A)-1)}, B) + 1$; if the final spike in B is unlinked, then $d(A, B) = d(A, B^{(M(B)-1)}) + 1$. Since at least one of these possibilities must hold, we have the relationship

$$\begin{aligned} d(A, B) = \min\big\{ & d\big(A^{(M(A)-1)}, B^{(M(B)-1)}\big) + q|A_{a_{M(A)}} - B_{b_{M(B)}}|, \\ & d\big(A^{(M(A)-1)}, B\big) + 1, d\big(A, B^{(M(B)-1)}\big) + 1\big\}. \end{aligned} \tag{7.8}$$

That is, we have expressed the distance between two spike trains in terms of the distances between spike trains that have been shortened by one spike.

It is far more efficient to implement (7.8) as a forward iteration than as a recursion. To make this explicit, we note that (7.8) holds for all spike trains A and B, so it may be recast as

$$\begin{aligned} d\big(A^{(\alpha)}, B^{(\beta)}\big) = \min\big\{ & d\big(A^{(\alpha-1)}, B^{(\beta-1)}\big) + q|A_\alpha - B_\beta|, \\ & d\big(A^{(\alpha-1)}, B^{(\beta)}\big) + 1, d\big(A^{(\alpha)}, B^{(\beta-1)}\big) + 1\big\}. \end{aligned} \tag{7.9}$$

Equation (7.9) amounts to an iterative procedure for calculating $D^{\text{spike}}[q](A, B) = d(A^{(M(A))}, B^{(M(B))})$ by building up each spike train one spike at a time. The iteration is initialized (at $\alpha = 0$ or $\beta = 0$ when one spike train is empty) by noting that $d(A^{(\alpha)}, B^{(0)}) = \alpha$ (since all spikes must be deleted) and $d(A^{(0)}, B^{(\beta)}) = \beta$ (since all spikes must be inserted). At each stage of the algorithm, one spike train or the other is elongated by a single spike, and the resulting distance is the minimum of three simply calculated quantities. Thus, the computational burden is proportional to the product of the number of spikes in each spike train, i.e., $O(M^2)$, where M is the typical number of spikes in each train to be compared.

We also mention that fast algorithms for identifying minimal-cost alignments can also be based on weighted bipartite matching (Dubbs et al. 2009), and these are applicable not only to $D^{\text{spike}}[q]$ but also to variants that may be more suitable for Euclidean embedding and multidimensional scaling.

Extensions of the Dynamic Programming Algorithm $D^{\text{spike}}[q](A, B)$ can be calculated for multiple values of the cost parameter q in an efficient, parallel fashion (Victor et al. 2007). The key observation is that the dependence of $D^{\text{spike}}[q](A, B)$ on q is piecewise linear. The breakpoints occur when the cost of a link is equal to 2 (so that removing it is equal to the cost of deleting and inserting its paired spikes). The positions of the breakpoints can be calculated by a dynamic programming algorithm whose computational burden is $O(M^3)$.

The above algorithms can also be applied to the multineuronal spike-time metric $D^{\text{spike}}[q, k]$. The main hurdle is that in a minimal-cost alignment of multineuronal spike trains, links between paired spikes may cross if their labels are different (Fig. 7.2A). Thus, it would appear that the iterative process of (7.9) would have to

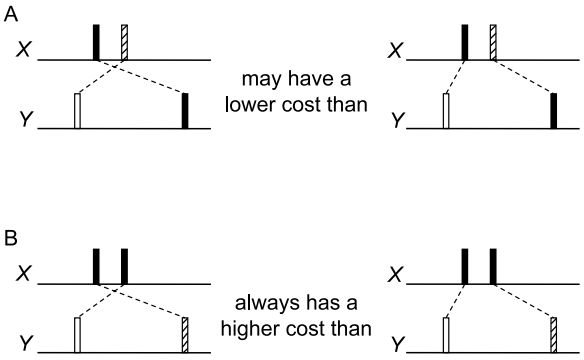

Fig. 7.2 The noncrossing rule for alignment of multineuronal responses. Spikes from different neurons (i.e., with different labels) are diagrammed as *filled*, *open*, and *hashed*. Panel **A** shows a configuration in which an alignment with crossed links may have a lower cost than an uncrossed alignment. Although the total link length is longer in the crossed alignment, one linked pair of spikes has the same label, allowing it to have a lower cost. In configuration **B**, both spikes in one of the spike trains (here, train X) have the same label. Because of this, the minimal-cost alignment must be uncrossed, since uncrossing the links does not influence whether the links connect spikes of the same label or of different labels

be implemented in nested loops for each of the L labels on each of the two spike trains. This would yield an algorithm with a computational burden of $O(M^{2L})$.

However, a closer look (Aronov 2003) yields an algorithm with a computational burden of only $O(M^{L+1})$. The reason for this reduction is the observation (Aronov 2003) that for certain configurations of labeled spikes, the noncrossing rule still holds (Fig. 7.2B). In particular, noncrossing can be guaranteed if both spikes in one train have the same label, since in this case, uncrossing the links does not affect whether any labels need to be changed and thus, cannot incur an additional cost k. To exploit this observation, the two spike trains are treated asymmetrically. That is, the iterative process of (7.9) is applied to one spike train independently of label and to the other spike train on a label-by-label basis. Details of the algorithm are provided in Aronov (2003).

7.2.3 Spike-Train Metrics Based on Vector-Space Embeddings

Above, we began with the notion of spike trains as event sequences and defined distances based on the "cost" of transformations between these sequences. Here, we describe another class of metrics that are based on embedding the spike trains into a vector space. Typically (van Rossum 2001; Richmond and Optican 1987), the embedding is linear, so that the resulting metric respects linearity. But this is not a prerequisite for this approach, and, more recently, Houghton (2009) has developed metrics based on nonlinear embeddings.

There are two main reasons for considering vector-space embeddings. First, the embedding process (i.e., the transformation of an event sequence $t_1, \ldots, t_{M(A)}$ into

a function of time $A(t)$) can be considered as an abstraction of what happens at a synapse: a sequence of action potentials results in a pattern of transmitter release, and this, in turn, has an effect $A(t)$ on the postsynaptic conductance. Second, in many cases, these metrics are simpler to calculate than the cost-based metrics.

7.2.3.1 Single-Neuron Metrics Based on Vector Space Embeddings

We use a formulation that encompasses the metrics considered by van Rossum (2001) and Houghton and Sen (2008). The first step is to represent a spike train A as a function of continuous time, $A(t)$. To do this, the event sequence $t_1, \ldots, t_{M(A)}$ is taken to be a sum of delta-functions

$$\delta_A(t) = \sum_{j=1}^{M(A)} \delta(t - t_j), \tag{7.10}$$

and the resulting sum $\delta_A(t)$ is convolved with a kernel function $K(t)$. This is equivalent to replacing each spike by the kernel waveshape and adding the waveshapes when they overlap. More formally,

$$A(t) = (\delta_A * K)(t) = \int_{-\infty}^{\infty} \delta_A(\tau) K(t - \tau)\, d\tau = \sum_{j=1}^{M(A)} K(t - t_j). \tag{7.11}$$

With the above embedding, any vector-space distance can be used to define a distance between the two spike trains, $d(A, B)$; in particular, the L^p-norm yields the distance

$$d(A, B) = \left(\int_{-\infty}^{\infty} \left| A(t) - B(t) \right|^p dt \right)^{1/p}. \tag{7.12}$$

For all metrics defined in this fashion, the distance between two spike trains that differ by inserting or deleting one spike is given by the L^p-norm of the kernel K, i.e., $(\int_{-\infty}^{\infty} |K(t)|^p\, dt)^{1/p}$.

van Rossum (2001) focuses on the L^2-distance (the Euclidean distance) and the exponential kernel

$$K^{VR}(t_c; t) = \begin{cases} \frac{1}{\sqrt{t_c}} e^{-t/t_c}, & t \geq 0, \\ 0, & t < 0. \end{cases} \tag{7.13}$$

We denote the metric that results from combining (7.11), (7.12) and (7.13) by $D^{VR}[t_c](A, B)$, where t_c is a parameter that controls the sensitivity of the metric to temporal detail. The kernel is normalized so that for two spike trains that differ by the addition or deletion of a single spike, $(D^{VR}[t_c](A, B))^2 = 1/2$, regardless of t_c.

$D^{VR}[t_c](A, B)$ is designed to be similar to $D^{\text{spike}}[q](A, B)$, with $1/t_c$ playing a role analogous to that of q in the cost-based metrics. That is, small values of $1/t_c$ compare spike trains based on the number of spikes, while large values of $1/t_c$ compare spike trains based on precise times of occurrence. To see this, we consider

several simple cases. For two spike trains that have only a single spike for which the times differ by an amount ΔT, the distance is given by $(D^{VR}[t_c](A, B))^2 = 1 - e^{-|\Delta T|/t_c}$, a quantity that is 0 at $\Delta T = 0$ and, for small ΔT, increases proportionally to $|\Delta T|/t_c$. In the limit as $t_c \to \infty$, $2(D^{VR}[t_c](A, B))^2$ is the square of difference in the number of spikes in the two trains (similar to the count-dominated behavior of $D^{\text{spike}}[q](A, B)$ as $q \to 0$), while in the limit as $t_c \to 0$, $2(D^{VR}[t_c](A, B))^2$ is the number of spikes in the two spike trains that occur at distinct times (matching the timing-dominated behavior of $D^{\text{spike}}[q](A, B)$ as $q \to \infty$).

Houghton and Sen (2008) explicitly consider L^1-norms and a kernel that makes the correspondence to $D^{\text{spike}}[q](A, B)$ even closer. For their kernel choice

$$K^{HS}(q; t) = \begin{cases} q/2, & 0 \le t < 2/q, \\ 0, & \text{otherwise}, \end{cases} \tag{7.14}$$

and $p = 1$ in (7.12), the correspondence of the distance $D^{HS}[q](A, B)$ to $D^{\text{spike}}[q](A, B)$ is exact when one spike train has no spikes, when both spike trains have one spike, when all spikes within each train are widely separated, when $q \to 0$, or when $q \to \infty$. However, like $D^{VR}[t_c](A, B)$, the correspondence of $D^{HS}[q](A, B)$ to $D^{\text{spike}}[q](A, B)$ is typically not exact when there are many spikes in each train, and the spikes occur with separations less than $2/q$ (or $O(t_c)$). The main qualitative difference is that $D^{VR}[t_c](A, B)$ and $D^{HS}[q](A, B)$ are Euclidean or can be converted to a Euclidean distance by a power-law transformation, while $D^{\text{spike}}[q](A, B)$ cannot (Aronov and Victor 2004).

In addition to the exponential (van Rossum 2001) and boxcar (Houghton and Sen 2008) kernel, other kernel shapes, such as a Gaussian, have been used in this context (Schreiber et al. 2004). Metrics on spike trains can also be derived by binning the spike times (Lim and Capranica 1994). Here, the temporal function $A(t)$ used in (7.12) is the mean firing rate in each bin. These metrics can be viewed as approximate versions of the metrics considered above, in which the convolution integral (7.11) is replaced by a discrete sum, and the kernel (for a bin width $2/q$) is given by (7.14).

Finally, we point out that the transformation from the sequence of delta-functions (7.10) to the temporal function $A(t)$ (7.11) need not be linear. Indeed, since the latter can be considered to represent the postsynaptic effect of an impulse train, it is reasonable to consider nonlinear transformations that caricature biophysical mechanisms, such as an incomplete return to resting potential when spikes are in rapid sequence. Houghton (Houghton and Sen 2008) has recently done this, implementing a simple model of short-term synaptic adaptation (Sen et al. 1996).

7.2.3.2 Multineuronal Metrics Based on Vector-Space Embeddings

Houghton and Sen (2008) have extended the above strategy to multineuronal metrics. The construction begins by extending, to multiple neurons, the representation (10) of a single neuron's spike train A as a sequence of delta-functions $\delta_A(t)$. To do this, they augment the delta-function corresponding to each spike by a unit vector $\vec{c}_l$;

the direction of this unit vector represents the label l (neuron of origin) of that spike. However, rather than simply assigning a separate orthogonal unit vector $\vec{e}_l$ as the direction for each of the L labels l, they allow the directions to be nonorthogonal, i.e., $\vec{c}_l = \sum c_{l,r}\vec{e}_r$. As we will see below, orthogonal labels correspond to labeled lines (the $k = 2$ extreme for $D^{\text{spike}}[q, k]$), collinear labels correspond to a summed population code (the $k = 0$ extreme for $D^{\text{spike}}[q, k]$), and intermediate choices correspond to intermediate behaviors.

With these preliminaries, a multineuronal spike train (with L neurons) with events at times t_j and associated with labels l_j is represented by an L-dimensional array of sums of scaled delta-functions,

$$\vec{\delta}_A(t) = \sum_{j=1}^{M(A)} \vec{c}_{l_j}\delta(t - t_j). \tag{7.15}$$

In coordinates, $\vec{\delta}_A(t) = (\delta_{A,1}(t), \ldots, \delta_{A,L}(t))$, where each scaled delta-function $\delta_{A,r}(t)$ is given by

$$\delta_{A,r}(t) = \sum_{j=1}^{M(A)} c_{l_j,r}\delta(t - t_j). \tag{7.16}$$

As in the single-neuron metrics, temporal factors are taken into account by convolving the delta-function array (7.15) by a kernel, yielding $\vec{A}(t) = (\vec{\delta}_A * K)(t)$, with the convolution carried out separately for each coordinate (7.11). (In principle, different kernels can be assigned to each coordinate, or different kernels can be assigned to each neuron prior to mixing them in (7.15).) Finally, the distance between two multineuronal spike trains A and B is the L^p-norm between their associated temporal functions $\vec{A}(t)$ and $\vec{B}(t)$, namely,

$$d(A, B) = \left(\int_{-\infty}^{\infty} \sum_{l=1}^{L} \left| A_l(t) - B_l(t) \right|^p dt \right)^{1/p}, \tag{7.17}$$

where $A_l(t)$ and $B_l(t)$ are, respectively, the lth component of $\vec{A}(t)$ and $\vec{B}(t)$. We note that (7.17) is equivalent to

$$d^p(A, B) = \sum_{l=1}^{L} d^p(A_l, B_l). \tag{7.18}$$

As an example of this construction, Houghton and Sen (2008) consider the two-neuron case and assign the unit vector $\vec{c}_1 = \binom{1}{0}$ to the first neuron, and $\vec{c}_2 = \binom{\cos\theta}{\sin\theta}$ to the second neuron. $\theta = 0$ corresponds to summing the spikes independent of neuron of origin, because spikes of all labels are represented by the same direction ($\vec{c}_1 = \vec{c}_2$). On the other hand, $\theta = \pi/2$ (for which $\vec{c}_1$ and $\vec{c}_2$ are orthogonal) corresponds to a labeled line code. In this case, each $A_l(t)$ and $B_l(t)$ consists of a sequence of delta-functions corresponding to the spikes of an individual neuron, and (7.18) shows that their contributions to the overall squared distance, (7.17), merely add.

7.2.3.3 Computational Considerations

Computation of metrics based on vector-space embeddings is straightforward. One strategy is to approximate the integrals in (7.12) or (7.17) or by a discrete sum; in this situation, the computational burden is $O(ML^2D)$, where M is the typical number of spikes from each neuron, L is the number of neurons, and D is the number of points used for the discretization (Houghton and Sen 2008). For the typical norms of $p = 1$ or $p = 2$, these integrals can be reduced to sums over all spikes ($p = 1$) or spike pairs ($p = 2$), resulting in a further saving if the density of spikes is low. Note that the computational burden of these metrics grows much more slowly as a function of the number of neurons (L) than for the cost-based metrics ($O(M^{L+1})$, see above).

7.2.4 Applications

7.2.4.1 Overview

In this section, we discuss several ways in which spike metrics can be used to ask biological questions about neuronal spike trains. (Here, we no longer distinguish between single-unit spike trains and multineuronal "labeled" spike trains; the material in this section applies to both.) We will mention applications to specific neural systems, but we organize our discussion in terms of the nature of the analytical goal. To do this, it is helpful to take a geometrical view, in which each spike train is represented by a point in an abstract space. The most direct application of spike metrics is simply to use the distances to quantify the variability within a set of spike trains. In the geometric view, the statistics of these distances provides a description of the cloud of neural responses.

However, quantifying variability does not probe the manner in which the spike trains relate to perception or behavior. One way to take this next step is to analyze the pairwise distances by standard multivariate techniques, such as multidimensional scaling. Via multidimensional scaling, the individual responses are then embedded into a neural "response space". In the neural response space, the distances between the points representing individual responses approximate the metric distances between the spike trains (as determined by a spike metric), and each kind of stimulus or behavior corresponds to a labeled cloud of points in the response space. One can then examine the positions of the clouds of spike trains that correspond to each stimulus or behavior and ask whether similarities in these domains correspond to similar kinds of neural activity. In other words, one can ask whether the notion of similarity provided by the spike metric provides a neural *representation* of the perceptual or behavioral phenomena.

From the point of view of neural coding, it is crucial to measure the quality of the representation of the perceptual or behavioral domain by the spike trains. Geometrically, this corresponds to asking whether the clouds that correspond to

each stimulus or behavior are distinct, vs. overlapping. The former corresponds to a high-fidelity representation, while the latter corresponds to a noisy one. Because of their nonparametric nature, the tools of information theory are natural to use in this context. If a spike metric leads to a high-fidelity representation, then the temporal features that it captures are candidates for neural codes. Conversely, when it is possible to show that behavioral performance exceeds that of a candidate code (Jacobs et al. 2009), the neural code can be ruled out.

7.2.4.2 Assessment of Variability

Measures of variability for time series depend in general on the time scale of interest; for example, a standard way of describing the variability of a continuous-time series is to compute its power spectrum, which quantifies variability as a function of frequency. This holds for point processes as well. In the present context, this observation means that in assessing the variability of spike trains, it makes sense to use spike metrics that are sensitive to a range of timescales (e.g., $D^{\text{spike}}[q]$ for a range of values of q).

Some examples of the use of spike metric for this purpose include the study of Kreiman et al. (2000), who examined a class of afferents in the weakly electric fish *Eigenmannia* whose spike trains are loosely phase-locked to the periodic discharge of its electrosensory organ. Kreiman et al. (2000) chose to quantify the variability by D^{spike}, since (because of the phase-locking) measures based on spike count or its variance proved ineffective in capturing trial-to-trial variability.

Grewe et al. (2003) used D^{spike}, along with a Euclidean metric, to examine variability in a motion-sensitive visual neuron (H1) of the blowfly. By examining variability in the presence of varying amounts of added sensory noise, they determined that performance of the H1 neuron was limited by its internal noise, rather than photon noise.

In response to full-field random flicker, retinal and lateral geniculate neurons often fire in discrete "firing events" consisting of several spikes, at times that are reproducible across trials. Reinagel and Reid (2002) used D^{spike} to quantify this reproducibility and, moreover, to show that these firing patterns were conserved not only across trials, but also across animals.

Finally, several authors have used D^{spike} to evaluate spike train variability of synthetic data for the purpose of evaluating models and describing their qualitative behavior (Keat et al. 2001; Tiesinga 2004; Banerjee et al. 2008).

7.2.4.3 Construction of Response Spaces

A variety of multivariate techniques can be used to visualize the geometry of the neural "response spaces" that correspond to a spike-train metric. The basic strategy for doing this is multidimensional scaling (MDS) (Kruskal and Wish 1978). In the framework of MDS, given a set of spike trains A and a particular spike metric D,

one seeks an embedding φ from the spike trains A to n-dimensional vectors $\varphi(A) = (\varphi_1(A), \ldots, \varphi_n(A))$ for which the spike metrics $D(A, B)$ are well approximated by the standard Euclidean distances between $\varphi(A)$ and $\varphi(B)$:

$$D(A, B)^2 \approx \left|\varphi(A) - \varphi(B)\right|^2 = \sum_{j=1}^{n} \big(\varphi_j(A) - \varphi_j(B)\big)^2. \tag{7.19}$$

Standard Multidimensional Scaling To determine an embedding φ that satisfies (7.19), we carry out the classical MDS recipe (Kruskal and Wish 1978). The crucial step is to diagonalize the symmetric matrix M_{AB}:

$$M_{AB} = \frac{1}{2}\Bigg(-D(A, B)^2 + \frac{1}{N}\sum_S D(A, S)^2 + \frac{1}{N}\sum_S D(B, S)^2 - \frac{1}{N^2}\sum_{S,S'} D\big(S, S'\big)^2\Bigg), \tag{7.20}$$

where N is the number of spike trains in the dataset, and the summations range over all spike trains S or pairs of spike trains S and S'. Rows and columns of M are indexed by each of the spike trains in the data set, so the entries in each of its eigenvectors $\phi^{[j]}$ are indexed by the spike trains as well. Thus, we may write

$$M_{AB} = \sum_{j=1}^{N} \lambda_j \phi_A^{[j]} \phi_B^{[j]}, \tag{7.21}$$

where $\phi_A^{[j]}$ denotes the value of the jth eigenvector at spike train A. Provided that the eigenvalues λ_j are all nonnegative, this allows us to write the desired embedding φ as

$$\varphi(A) = \big(\varphi_1(A), \ldots, \varphi_n(A)\big) = \big(\sqrt{\lambda_1}\phi_A^{[1]}, \ldots, \sqrt{\lambda_n}\phi_A^{[n]}\big) \tag{7.22}$$

(see Kruskal and Wish 1978).

The analysis also yields embeddings in lower-dimensional spaces that approximate the metric $D(A, B)$. To find an approximate embedding, we simply number the eigenvalues in descending order and only use the first $K < N$ of them for coordinates in (7.22). These approximate embeddings (e.g., with $K = 2$ or $K = 3$) can be used to visualize the relative similarities of a set of spike trains, as determined by $D(A, B)$.

Examples The above procedures have been used to characterize the geometry of response spaces in several sensory systems. In the auditory system, Victor and Purpura (1997) applied this approach to the data of Middlebrooks et al. (1994), who recorded responses of single neurons in cat ectosylvian gyrus to sounds that varied in spatial location. The original data showed that these neurons' responses could distinguish the origin of the sound source in a panoramic (360°) fashion and that spike timing at a resolution of ca. 4 ms was critical for this (Middlebrooks et al. 1994; Furukawa and Middlebrooks 2002). The reanalysis (Victor and Purpura 1997),

which applied MDS to the distances yielded by D^{spike} and D^{interval}, added to this picture by showing that the geometry of these responses (i.e., their relative distances) recovered the circular geometry of the stimulus space.

In the visual system, Aronov et al. (2003) used $D^{\text{spike}}[q, k]$ to characterize the representation of spatial phase across pairs of neurons in primary visual cortex. The geometry of the stimulus set (the circle of spatial phase) could be recovered by applying MDS to the response similarities, as quantified by $D^{\text{spike}}[q, k]$. Moreover, response spaces became more nearly circular for nonzero values of k, indicating that within a local cluster, the neuron of origin of a spike, as well as its timing, contributes to the representation of spatial phase.

Aronov et al. (2003) also introduced a technique that can be used to interpret the axes obtained in the MDS procedure. Essentially, for each dimension m in (7.22), they regressed the coordinate value m of each response, $\phi_A^{[m]}$, against a binned post-stimulus histogram $A(t)$. By showing that $\phi_A^{[m]}$ could be approximated by a linear weighting of the response time course, i.e.,

$$\phi_A^{[m]} \approx \int A(t) L_m(t)\, dt, \tag{7.23}$$

they identified "temporal profiles" $L_m(t)$ that could be associated with each dimension of the embedding.

Recently, Di Lorenzo et al. (2009) applied this procedure to single-neuron recordings in the nucleus tractus solitarius of the rat, during stimulation by each of the four primary tastants (salt, sweet, sour, bitter) and their binary mixtures. Neurons that were broadly tuned in terms of spike count were nevertheless able to distinguish among these tastants via temporal coding. The neural response space, as constructed by the above procedure, recapitulated the tetrahedral geometry anticipated from gustatory psychophysics (Erickson 1984).

Implications of Non-Euclidean Nature of Spike Metrics The above approach (standard MDS) seeks to embed spike trains into a Euclidean vector space in a distance-preserving manner. Thus, the embedding can only be exact if the spike metric itself is Euclidean, which is not the case (Aronov and Victor 2004). The above procedure fails to be exact for non-Euclidean metrics because some of the eigenvalues of M are negative. Consequently, the bilinear form (7.21) is not positive definite (and hence, not an inner product), and the coordinates (7.22) are not real. Nevertheless, since the first several eigenvalues of M are typically positive, the above procedure finds an Euclidean approximation of the spike metric that suffices for visualization of the spike trains. Other approaches to deal with this problem are described below.

Relationship to the "Kernel Trick" and van Rossum-Type Metrics The "kernel trick" is applicable to a situation in which one wishes to classify a set of objects (here, the spike trains) which either do not have a linear structure, or in which the important features are nonlinearly related to the objects. To do this, one introduces an embedding ψ, which maps the objects into a vector space. Then, the original objects can be classified via linear classifiers in the feature space.

This is a generalization of the approach of van Rossum (2001), Richmond and Optican (1987), in which spike trains are embedded into an infinite-dimensional vector space (functions of time) by convolving them with a smoothing kernel (e.g., (7.11)). Suppose that we have a set of spike trains A_k that correspond to temporal functions $A_k(t)$ via this transformation. It is natural to ask what would happen if we apply the above MDS procedure to their L^2-distances, (7.12). Since this is manifestly a Euclidean distance, an exact embedding must be possible. However, we cannot recover the original vector space of functions of time, because the latter is infinite-dimensional, but the MDS procedure necessarily yields a finite-dimensional embedding. Instead, we would find that the successive eigenvectors of M correspond to the successive principal components of the set of temporal functions $A_k(t)$ (minus their overall mean). That is, although the MDS procedure does not recover the smoothing kernel that defines the distance, it finds the vector space that contains all of the temporal functions that result from applying this kernel.

As a converse, we mention that these relationships imply that the non-Euclidean spike metrics cannot be calculated via a "kernel trick", since if this were possible, then the metrics could be recovered *exactly* by an MDS procedure (see below).

Nonlinear Scaling Above we have focused on finding an embedding φ that approximates a spike metric by a Euclidean distance (7.19). However, since spike metrics typically do not correspond to any Euclidean distance (Aronov and Victor 2004), there is no guarantee that embeddings that satisfy (7.19) can be found. This motivates several other strategies for response space construction, which we now briefly mention. One such strategy is to seek embeddings into a curved space. That is, the Euclidean distance on the right-hand side of (7.19) is replaced by the geodesic distance between $\varphi(A)$ and $\varphi(B)$ within a data-defined curved manifold. This can be accomplished by the isomap (Tenenbaum et al. 2000) and geometric diffusion (Coifman et al. 2005) methods of dimensionality reduction. A second approach is to replace $D(A, B)$ in (7.19) by $f(D(A, B))$ for some monotonic function f. The resulting embedding preserves the rank order of the distances, and, if f is concave, $f(D(A, B))$ remains a metric. $f(D(A, B)) = D(A, B)^\beta$ for $0 < \beta < 1$ is a natural choice for this transformation, since it is scale-invariant. However, while specific choices of β (e.g., $\beta = 1/2$) may be useful to "euclideanize" particular datasets, it is possible to show (Aronov and Victor 2004) that there is no single choice of $\beta > 0$ that universally suffices to transform $D^{\text{spike}}[q, k]$ into a Euclidean distance.

7.2.4.4 Applications to Information-Theoretic Analysis

Finally, we consider applications of spike metrics to information-theoretic analysis of neural data. Information theory (IT) (Shannon and Weaver 1949) (see Cover and Thomas 1991 for a general review) forms a natural framework for the analysis of neural coding (Rieke et al. 1997) (Chap. 13). However, the application of IT to neural systems can be difficult: estimates of mutual information from laboratory data can be biased, imprecise, or both. The estimation problem, whose origin is that the

space of possible responses is undersampled, is compounded for analyses of multineuronal activity, because the dimensionality of the response domain is proportional to the number of neurons.

To motivate some strategies to mitigate this difficulty, we consider a naïve attempt to estimate mutual information from laboratory data. The mutual information between a set of stimuli S and a set of responses R is defined as

$$I(S,R) = -\sum_{s\in S} p(s)\log p(s) + \sum_{r\in R} p(r)\sum_{s\in S} p(s|r)\log p(s|r), \qquad (7.24)$$

where the first term is the entropy of the unconditioned distribution of stimuli, and the second term subtracts the average entropy of the distribution of stimuli, conditioned on observing a particular response $r \in R$. Typically, the distribution of stimuli is known, so the estimation of information rests on the conditional probabilities $p(s|r) = p(s,r)/p(r)$. Below we will make use of a reorganization of (7.24) into a symmetric form,

$$I(S,R) = -\sum_{s\in S} p(s)\log p(s) - \sum_{r\in R} p(r)\log p(r) + \sum_{s\in S, r\in R} p(s,r)\log p(s,r), \qquad (7.25)$$

which states that mutual information is the difference between the entropies of the stimulus and response distributions (the first two terms) and the entropy of their joint distribution (the third term).

Implementing (7.24) or (7.25) directly requires estimates of the joint stimulus-response probabilities $p(s,r)$. The obvious way to obtain these estimates is to count up the number of joint occurrences of each stimulus s and each response r, and divide by the number of events. To do this, the investigator must know how to partition the response domain R into different kinds of responses, $r_1, r_2, \ldots$. The Data Processing Inequality states that if probability estimates from different responses r_1 and r_2 are inadvertently pooled, then the resulting estimate of mutual information will be downwardly biased. Thus, it might appear that the most conservative approach would be to estimate $p(s,r)$ by separately tracking all distinguishable responses. This would then avoid the downward bias due to pooling responses.

The difficulty with this approach is that when the response domain is partitioned very finely, then the number of events that contributes to each estimate of $p(s,r)$ is small, often either 0 or 1. Since entropy and information are nonlinear functions of the underlying probabilities, an overly narrow binning of the response space incurs an *upward* bias (Treves and Panzeri 1995; Miller 1955; Carlton 1969; Victor 2000) in the estimate of mutual information. So the investigator has a dilemma: to avoid a downward bias due to the Data Processing Inequality, the stimulus domain must be sampled as finely as possible, but this leads to an upward bias because of the nonlinearity of the logarithm.

There are two general strategies that can be used to mitigate this dilemma (see Victor 2006 for further details). One strategy is to use a fine partitioning of the response domain but to use advanced estimation techniques (Paninski 2003;

Nemenman et al. 2004) to reduce the upward estimation bias inherent in naïve estimators. This strategy makes minimal assumptions about the nature of the neural code but still has large data requirements.

The second strategy, which is where spike metrics are relevant, is based on the observation that pooling distinct responses r_1 and r_2 only reduce the estimate of mutual information when the responses are associated with distinct a posteriori probabilities, $p(s|r_1)$ and $p(s|r_2)$. Conversely, if two responses r_1 and r_2 have the same "meaning" (i.e., lead to identical a posteriori estimates of what the stimulus was), then probability estimates can be pooled without incurring an upward bias.

This places the focus on determining which neural responses have the same meaning. Each spike metric, in essence, is a formal hypothesis about exactly this: two neural responses are hypothesized to have the same meaning if their distance is zero, and increasing distances indicates progressively different meanings.

Motivated by this observation, a spike metric can be used to provide an estimate of mutual information that is relatively unaffected by the undersampling problem but strongly sensitive to whether, in fact, the hypothetical distance is close to the correct one. A strategy for doing this for a discrete stimulus set is detailed in Victor and Purpura (1997); we summarize it here.

The first step in the analysis is to calculate all of the pairwise distances between the responses. Then, response clusters are created, with one cluster for each stimulus s_α. The clusters are formed based on the experimenter's knowledge of which stimulus elicited each response. In particular, a response r is placed into a cluster β if the average distance between r and all the responses elicited by the stimulus s_β is smaller than the average distance between r and the responses elicited by any other stimulus. (The averaging process is carried out following a negative-power law transformation, to emphasize near-matches. The exponent, typically -2, is the parameter z of Victor and Purpura 1997.) The result of applying this clustering to all responses is a table $N(\alpha, \beta)$ which counts the number of times that a response to the stimulus s_α is assigned to cluster β. From this table mutual information is estimated as

$$\begin{aligned} I(S, R) \approx & -\sum_{\alpha} p(\alpha, \bullet) \log p(\alpha, \bullet) - \sum_{\beta} p(\bullet, \beta) \log p(\bullet, \beta) \\ & + \sum_{\alpha, \beta} p(\alpha, \beta) \log p(\alpha, \beta), \end{aligned} \tag{7.26}$$

where $p(\alpha, \beta) = \frac{N(\alpha, \beta)}{\sum_{\alpha, \beta} N(\alpha, \beta)}$, $p(\alpha, \bullet) = \sum_\beta p(\alpha, \beta)$, and $p(\bullet, \beta) = \sum_\alpha p(\alpha, \beta)$.

The reason that (7.26) represents a useful approach to the undersampling strategy is that $p(s, r)$ has been replaced by $p(\alpha, \beta)$. The former requires keeping track, separately, of the number of occurrences of each response, while the latter only requires keeping track of the number of occurrences of responses in the same cluster. That is, we have dealt with the undersampling problem by introducing a clustering procedure: we have lumped responses together (into the same cluster β) if they are

similar to responses that are known to be elicited by the stimulus s_β.[5] The Data Processing Inequality guarantees that this will result in an underestimate of information (assuming that the clusters are sufficiently well sampled), but the key point is that the extent of this underestimate is determined by whether or not the metric-based clustering properly classifies spike trains that have the same meaning.

In sum, estimates of information based on (7.26) reflect the extent to which the chosen spike metric accurately captures meaningful differences between spike trains. That is, the dependence of the information estimates of (7.26) on the choice of spike metric (e.g., for $D^{\text{spike}}[q,k]$, the temporal precision parameter q and the spike label parameter k) characterizes the informative features of the neural response. In particular, if $q_{\max}$ is the value of q for which estimates of information via (7.26) based on $D^{\text{spike}}[q]$ (or $D^{\text{spike}}[q,k]$) achieves its highest value, then $1/q_{\max}$ can be viewed as the "informative temporal precision" of a spike, namely, the amount of time by which moving a spike has the same impact on the meaning of the spike train as deleting the spike altogether.

Examples Victor and Purpura (1996) used this approach extensively to characterize how individual neurons in primary (V1) and secondary (V2 and V3) visual cortices carried information about multiple stimulus attributes, such as contrast, orientation, and texture. They found that greater information was recovered from clustering based on $D^{\text{spike}}[q]$ than from that based on $D^{\text{interval}}[q]$. Moreover, using $D^{\text{spike}}[q]$, they found that the informative precision of a neuron's response depended on the stimulus attribute—with the highest precision for contrast (10 to 30 ms) and lowest precision for texture ($\sim$100 ms). Thus, individual spike trains can be considered to carry information about several stimulus attributes in a temporally multiplexed fashion.

Estimates of informative temporal precision obtained by the above approach are necessarily averages across the entire response. This is underscored by the study of Reich et al. (2000), who used D^{spike} to show that most of the information about contrast could be extracted from the latency of the first spike and that, if only the first spike is considered, the informative temporal precision can be as low as 2 to 5 ms.

Samonds and Bonds (2004) and Samonds et al. (2003), examined signaling of orientation in cat primary visual cortex with D^{spike} and D^{interval}, including multineuronal extensions of these measures. Their analysis showed that large orientation differences could be signaled by firing rate but that orientation differences of 10° or less were signaled by the temporal fine structure (2 to 10 ms) of spike times and spike intervals.

Additional information-theoretic applications of spike metrics in the visual system include the work of Mechler et al. (1998), who showed that temporal structure

[5] The "hard clustering" used in this procedure might also lead to an underestimate of information, in comparison to a procedure that gives soft, or probabilistic, assignments to each response. Recently, I. Nelken (2009) has proposed a procedure that circumvents this difficulty, by applying the "binless embedding" method (Victor 2002) to the distances calculated by spike metrics.

played a much larger role in the coding of edges (square-wave gratings) than of smooth variations (sine gratings), and the study of Chichilnisky and Rieke (2005), who found that for near-threshold responses in retinal ganglion cells, the informative temporal precision was approximately 100 ms.

There have also been a number of applications of this approach in other sensory systems (e.g., Machens et al. 2001 in the grasshopper auditory system, Di Lorenzo and Victor 2003, 2007 in the gustatory system). Particularly noteworthy are the combined behavioral and neurophysiological experiments of Laurent and colleagues (MacLeod et al. 1998) in the olfactory system of the locust. They used D^{spike} to show that increasing timing jitter of spike trains in projection neurons leads to loss of behavioral discrimination of similar odors, though coarse odor discrimination remains intact (Stopfer et al. 1997). These elegant experiments demonstrate the functional relevance of precise spike timing for fine sensory discriminations.

Several of the above studies (e.g., Victor and Purpura 1996; Di Lorenzo and Victor 2003, 2007) made use of a surrogate data technique, "exchange resampling". The authors reanalyzed surrogate data sets in which individual spikes were swapped across pairs of responses to the same stimulus. These surrogate data sets, by construction, have the same number of spikes in each trial as the original data and have the same time-varying firing rates as the original data, but spike correlations within trials are destroyed. Information estimates from these surrogate data sets were somewhat less than the estimates obtained from the original data, indicating that the pattern of spikes in individual trials, and not just the time course of firing probability, was informative.

It is important to emphasize that these methods quantify the amount of information that is available in the neural response and the spike train features that carry this information. In order to claim that the information is actually used, the analyses must eventually be coupled to behavior (MacLeod et al. 1998; Jacobs et al. 2009).

7.3 Conclusion

Spike metrics provide a formal structure for analyzing neural activity as event sequences and provide a means to assess variability, to visualize patterns of response similarity, and to estimate information-theoretic quantities. While many simple and useful spike metrics can be calculated by efficient dynamic programming algorithms, extensions of the approach to additional metrics present a range of algorithmic challenges.

Acknowledgements This work is supported in part by National Eye Institute Grant 1RO1 EY9314 to J. Victor and National Institute of Mental Health Grant 1RO1 MH68012 to Daniel Gardner. We would like to thank Sebastien Louis for his thorough reading of a draft and his thoughtful and helpful comments.

References

Abbott LF (2000) Integrating with action potentials. Neuron 26:3–4
Abeles M (1982) Role of the cortical neuron: integrator or coincidence detector?. Isr J Med Sci 18:83–92
Abeles M, Prut Y (1996) Spatio-temporal firing patterns in the frontal cortex of behaving monkeys. J Physiol Paris 90:249–250
Aronov D (2003) Fast algorithm for the metric-space analysis of simultaneous responses of multiple single neurons. J Neurosci Methods 124:175–179
Aronov D, Victor JD (2004) Non-Euclidean properties of spike train metric spaces. Phys Rev E Stat Nonlin Soft Matter Phys 69:061905
Aronov D, Reich DS, Mechler F, Victor JD (2003) Neural coding of spatial phase in V1 of the macaque monkey. J Neurophysiol 89:3304–3327
Banerjee A, Series P, Pouget A (2008) Dynamical constraints on using precise spike timing to compute in recurrent cortical networks. Neural Comput 20:974–993
Carlton AG (1969) On the bias of information estimates. Psychol Bull 71:108–109
Chichilnisky EJ, Rieke F (2005) Detection sensitivity and temporal resolution of visual signals near absolute threshold in the salamander retina. J Neurosci 25:318–330
Coifman RR, Lafon S, Lee AB, Maggioni M, Nadler B, Warner F, Zucker SW (2005) Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. Proc Natl Acad Sci USA 102:7426–7431
Cover TM, Thomas JA (1991) Elements of information theory, Schilling DL (ed). Wiley, New York
Dan Y, Poo MM (2004) Spike timing-dependent plasticity of neural circuits. Neuron 44:23–30
Di Lorenzo PM, Victor JD (2003) Taste response variability and temporal coding in the nucleus of the solitary tract of the rat. J Neurophysiol 90:1418–1431
Di Lorenzo PM, Victor JD (2007) Neural coding mechanisms for flow rate in taste-responsive cells in the nucleus of the solitary tract of the rat. J Neurophysiol 97:1857–1861
Di Lorenzo PM, Chen J-Y, Victor JD (2009) Quality time: representation of a multidimensional sensory domain through temporal coding. J Neurosci 29(29):9227–9238
Dubbs AJ, Seiler BA, Magnasco MO (2009) A fast Lp spike alignment metric. arXiv:0907.3137v2
Egger V, Feldmeyer D, Sakmann B (1999) Coincidence detection and changes of synaptic efficacy in spiny stellate neurons in rat barrel cortex. Nat Neurosci 2:1098–1105
Erickson RP (1984) Ohrwall, Henning and von Skramlik; the foundations of the four primary positions in taste. Neurosci Biobehav Rev 8:105–127
Furukawa S, Middlebrooks JC (2002) Cortical representation of auditory space: information-bearing features of spike patterns. J Neurophysiol 87:1749–1762
Gaal SA (1964) Point set topology. Academic Press, New York
Goldberg DH, Victor JD, Gardner EP, Gardner D (2009) Spike train analysis toolkit: enabling wider application of information-theoretic techniques to neurophysiology. Neuroinformatics 7(3):165–178
Grewe J, Kretzberg J, Warzecha AK, Egelhaaf M (2003) Impact of photon noise on the reliability of a motion-sensitive neuron in the fly's visual system. J Neurosci 23:10776–10783
Hopfield JJ (1995) Pattern recognition computation using action potential timing for stimulus representation. Nature 376:33–36
Houghton C. (2009) Studying spike trains using a van Rossum metric with a synapse-like filter. J Computat Neurosci 26:149–155
Houghton C, Sen K (2008) A new multineuron spike train metric. Neural Comput 20(6) 1495–1511
Jacobs AL, Fridman G, Douglas RM, Alam NM, Latham PE, Prusky GT, Nirenberg S (2009) Ruling out and ruling in neural codes. Proc Natl Acad Sci USA 106:5936–5941
Keat J, Reinagel P, Reid RC, Meister M (2001) Predicting every spike: a model for the responses of visual neurons. Neuron 30:803–817

Kreiman G, Krahe R, Metzner W, Koch C, Gabbiani F (2000) Robustness and variability of neuronal coding by amplitude-sensitive afferents in the weakly electric fish eigenmannia. J Neurophysiol 84:189–204
Kruskal JB, Wish M (1978) Multidimensional scaling. Sage, Beverly Hills
Kuba H, Yamada R, Fukui I, Ohmori H (2005) Tonotopic specialization of auditory coincidence detection in nucleus laminaris of the chick. J Neurosci 25:1924–1934
Lim D, Capranica RR (1994) Measurement of temporal regularity of spike train responses in auditory nerve fibers of the green treefrog. J Neurosci Methods 52:203–213
Machens C, Prinz P, Stemmler M, Ronacher B, Herz A (2001) Discrimination of behaviorally relevant signals by auditory receptor neurons. Neurocomputing 38:263–268
MacLeod K, Backer A, Laurent G (1998) Who reads temporal information contained across synchronized and oscillatory spike trains?. Nature 395:693–698
Maloney LT, Yang JN (2003) Maximum likelihood difference scaling. J Vision 3:5. doi:10.1167/3.8.5
Markram H, Lubke J, Frotscher M, Sakmann B (1997) Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. Science 275:213–215
Mechler F, Victor JD, Purpura KP, Shapley R (1998) Robust temporal coding of contrast by V1 neurons for transient but not for steady-state stimuli. J Neurosci 18:6583–6598
Middlebrooks JC, Clock AE, Xu L, Green DM (1994) A panoramic code for sound location by cortical neurons. Science 264:842–844
Miller GA (1955) Note on the bias on information estimates. Information Theory in Psychology: Problems and Methods II-B:95–100
Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 48:443–453
Nelken I (2009) Personal communication
Nemenman I, Bialek W, de Ruyter van Steveninck R (2004) Entropy and information in neural spike trains: progress on the sampling problem. Phys Rev E Stat Nonlin Soft Matter Phys 69:056111
Paninski L (2003) Estimation of entropy and mutual information. Neural Comput 15:1191
Reich D, Mechler F, Victor J (2000) Temporal coding of contrast in primary visual cortex: when, what, and why?. J Neurophysiol 85:1039–1050
Reinagel P, Reid RC (2002) Precise firing events are conserved across neurons. J Neurosci 22:6837–6841
Richmond BJ, Optican LM (1987) Temporal encoding of two-dimensional patterns by single units in primate inferior temporal cortex. II. Quantification of response waveform. J Neurophysiol 57:147–161
Rieke F, Warland D, de Ruyter van Steveninck R, Bialek W (1997) Spikes: exploring the neural code. MIT Press, Cambridge
Samonds JM, Bonds AB (2004) From another angle: differences in cortical coding between fine and coarse discrimination of orientation. J Neurophysiol 91:1193–1202
Samonds JM, Allison JD, Brown HA, Bonds AB (2003) Cooperation between area 17 neuron pairs enhances fine discrimination of orientation. J Neurosci 23:2416–2425
Schreiber S, Fellous JM, Tiesinga P, Sejnowski TJ (2004) Influence of ionic conductances on spike timing reliability of cortical neurons for suprathreshold rhythmic inputs. J Neurophysiol 91:194–205
Segundo JP, Perkel DH (1969) The nerve cell as an analyzer of spike trains. In: Brazier MAB (ed) The interneuron. University of California Press, Berkeley, pp 349–390
Sellers P (1974) On the theory and computation of evolutionary distances. SIAM J Appl Math 26:787–793
Sen K, Jorge-Rivera JC, Marder E, Abbott LF (1996) Decoding synapses. J Neurosci 16:6307–6318
Shannon CE, Weaver W (1949) The mathematical theory of communication. University of Illinois Press, Urbana
Singh G, Memoli F, Ishkhanov T, Sapiro G, Carlsson G, Ringach DL (2008) Topological analysis of population activity in visual cortex. J Vision 8:1–18

Slepian D (1976) On bandwidth. Proc IEEE 64:292–300
Softky WR, Koch C (1993) The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs. J Neurosci 13:334–350
Stopfer M, Bhagavan S, Smith BH, Laurent G (1997) Impaired odour discrimination on desynchronization of odour-encoding neural assemblies. Nature 390:70–74
Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290:2319–2323
Tiesinga PHE (2004) Chaos-induced modulation of reliability boosts output firing rate in downstream cortical areas. Phys Rev E Stat Nonlin Soft Matter Phys 69:031912
Treves A, Panzeri S (1995) The upward bias in measures of information derived from limited data samples. Neural Comput 7:399–407
Tversky A (1977) Features of similarity. Psychol Rev 84:327–352
Tversky A, Gati I (1982) Similarity, separability, and the triangle inequality. Psychol Rev 89:123–154
van Rossum MC (2001) A novel spike distance. Neural Comput 13:751–763
Victor JD (2000) Asymptotic bias in information estimates and the exponential (Bell) polynomials. Neural Comput 12:2797–2804
Victor JD (2002) Binless strategies for estimation of information from neural data. Phys Rev E 66:51903
Victor JD (2006) Approaches to information-theoretic analysis of neural activity. Biological Theory 1:302–316
Victor JD, Purpura KP (1996) Nature and precision of temporal coding in visual cortex: a metric-space analysis. J Neurophysiol 76:1310–1326
Victor JD, Purpura KP (1997) Metric-space analysis of spike trains: theory, algorithms and application. Network 8:127–164
Victor JD, Goldberg DH, Gardner D (2007) Dynamic programming algorithms for comparing multineuronal spike trains via cost-based metrics and alignments. J Neurosci Methods 161:351–360
Wu L, Gotman J (1998) Segmentation and classification of EEG during epileptic seizures. Electroencephalogr Clin Neurophysiol 106:344–356
Wuerger SM, Maloney LT, Krauskopf J (1995) Proximity judgments in color space: tests of a Euclidean color geometry. Vision Res 35:827–835

Chapter 8
Gravitational Clustering

George Gerstein

Abstract When multiple spike trains are simultaneously recorded, it is desirable to screen the data for signs of neuronal association. The gravity transformation does this by representing the N observed spike trains as charges on N particles moving in an N-space. An appropriate force law results in particle trajectories and aggregations that can be interpreted in terms of dynamic neuronal interactions and of neuronal assemblies.

8.1 Introduction

Since the seminal work of Hebb (1949), it has been commonly accepted that neurons do not act alone but form assemblies for their various tasks. Indeed as soon as it became possible to simultaneously record the activity of two neurons, at the same time it was noted that some neurons exhibited nonrandom, frequently near-synchronous time relationships to each other and to external events in the laboratory like stimulus or movement. Various analysis methods related to correlation of spike activity allowed parsing of such observations into "effective connectivity", a simplified cartoon of neuron-like elements that would produce similar temporal activity relations (Gerstein and Perkel 1972; Aertsen et al. 1989). The attendant vocabulary included (a) direct synaptic connection, (b) shared synaptic input, (c) stimulus-related modulation of either, and (d) stimulus-related shared rate modulations. These circuit models have a progressively increasing time scale starting in the milliseconds for direct synaptic effects and going up to seconds for the shared rate effects. However, two or three neurons do not make an assembly.

G. Gerstein (✉)
Department of Neuroscience, University of Pennsylvania, 215 Stemmler Hall, 3405 Hamilton Walk, Philadelphia, PA 19104-6074, USA
e-mail: george@mulab.physiol.upenn.edu
url: http://mulab.physiol.upenn.edu

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_8, © Springer Science+Business Media, LLC 2010

In due course recording technology has advanced. It is now possible to record some 100 neurons at a time in a given structure either by optical or electrode means. There remain problems in the spike shape sorting which is needed to produce clean spike trains, each coming from only a particular neuron. Sufficient effort and verification, tedious though it be, can overcome these problems. However, because of a combinatorial explosion, it is highly impractical to use pairwise analysis on such data to search for signatures of assemblies and for investigation of their properties. This chapter about gravitational clustering is one approach to an efficient search for evidence of interactions among many simultaneously recorded neurons. Although the representation is still fundamentally pairwise, it allows evaluation of all observed neurons at the same time. Furthermore, like the time-resolved correlation methods for neuronal pairs, the gravity representation allows the observation of dynamic behavior in assembly structure and membership. In its basic form the gravity representation (Gerstein et al. 1985; Gerstein and Aertsen 1985) is most sensitive to near-synchronous time relations among neurons, a restriction similar to that of "Unitary Event" analysis (Grün et al. 2002; Pauluis and Baker 2000). See Chap. 10. Various modifications of the basic gravity analysis allow lifting this restriction and enhancing performance. The gravitational clustering approach has been successfully applied to a range of experimental data (Aertsen et al. 1987, 1991; Gochin et al. 1990; Lindsey et al. 1989, 1992a, 1992b, 1997; Lindsey 2001; Maldonado and Gerstein 1996; Gerstein et al. 1998; Arata et al. 2000; Martin 2001; Morris et al. 2003).

We first describe the basic gravity representation and various visualization tools available to examine its results. We then will examine variations on the theme for increased sensitivity and selectivity. However, the exposition in this article is necessarily condensed, and it is advisable to consult the original papers for details and examples.

8.2 The Basic Gravity Representation

Temporal analysis of multiple spike train data is mapped into a mechanical system where spike timing effects produce spatial clustering; this type of clustering analysis was originally proposed for detecting botanical similarity (Wright 1977). Each neuron is represented as a particle in a high-dimensional space. Action potentials produce transient "charges" on the particle which corresponds to the neuron fired. The particles move according to forces caused by attraction or repulsion of the charges. In consequence there will be aggregation of articles corresponding to neurons which fire near synchronously. Such aggregation then is the signature of synchronous neuronal assemblies if they are present. The speed of aggregation or disaggregation reflects the dynamics of the assembly structure during the time course of the recording. This approach should be compared to multiple pair correlation. See Chap. 5.

N neurons are assumed to be simultaneously recorded. Following the original work on gravitational clustering (Gerstein et al. 1985; Gerstein and Aertsen 1985),

we represent these neurons as N particles in N-dimensional space. The initial coordinate along the jth axis of the ith particle is given by

$$\mathbf{x}_i^j = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases} \tag{8.1}$$

The charge on particle i at time t is given by

$$q_i(t) = \sum_k K(t - T_k) - \lambda_i, \tag{8.2}$$

where T_k are the times of the spikes fired by this cell, and $K(t)$ is a kernel function which is convolved with the spike train. The kernel function is assumed to have unit area and is frequently taken as a simple decreasing exponential with a time constant τ. The firing rate of the ith neuron is given by λ_i; this is subtracted to ensure that the mean charge on a given particle is zero. In the basic version of the analysis, only data with stationary firing rates are considered. However, it is possible to extend the method to nonstationary rates, in which case a time varying rate $\lambda_i(t)$ is subtracted in (8.2). The rate is normally estimated over a short window of data (Gerstein and Aertsen 1985). Figure 8.1 illustrates the transformation of a spike train into a charge function (8.2) using forward or backward exponential kernels. Obviously kernels of different shape and time scale can be used; in particular it is worth considering gamma like kernels that mimic synaptic potentials and have an effective dead time. Issues of kernel time scale are addressed in Chap. 6.

In the usual gravitational clustering analysis, particle positions are adjusted according to the pairwise attractions between particles. Thus, using a discrete time step of δt, the position update rule is

$$\mathbf{x}_i(t + \partial t) = \mathbf{x}_i(t) + \kappa \partial t \sum_{\forall j \neq i} Q_{i,j} \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|}, \tag{8.3}$$

where

$$Q_{i,j} = q_i q_j. \tag{8.4}$$

Note that this is a force and update rule that has velocity rather than acceleration proportional to attractive force. In real physical systems this would only occur over a narrow range of conditions for particles moving in a viscous medium.

The constant κ controls both the observed variance (noise) in the particle trajectories and the rate of aggregation of particles which correspond to synchronized neurons. The time step ∂t has usually been fixed at 1 ms in most applications. The denominator term within the summation prevents a dependence of the force on the current separation of particles. The computational instability as the two particles approach very closely is usually eliminated by setting the interparticle force to zero when the interparticle distance is less than some critical threshold, typically 10% of the original distance. An alternative version of (8.3), where the denominator term is omitted, has also been described (Dayhoff 1994) and eliminates the instability at close particle approach distance. That version has a computational advantage since its run time scales as N^2 rather than N^3 of the original formulation. However, the

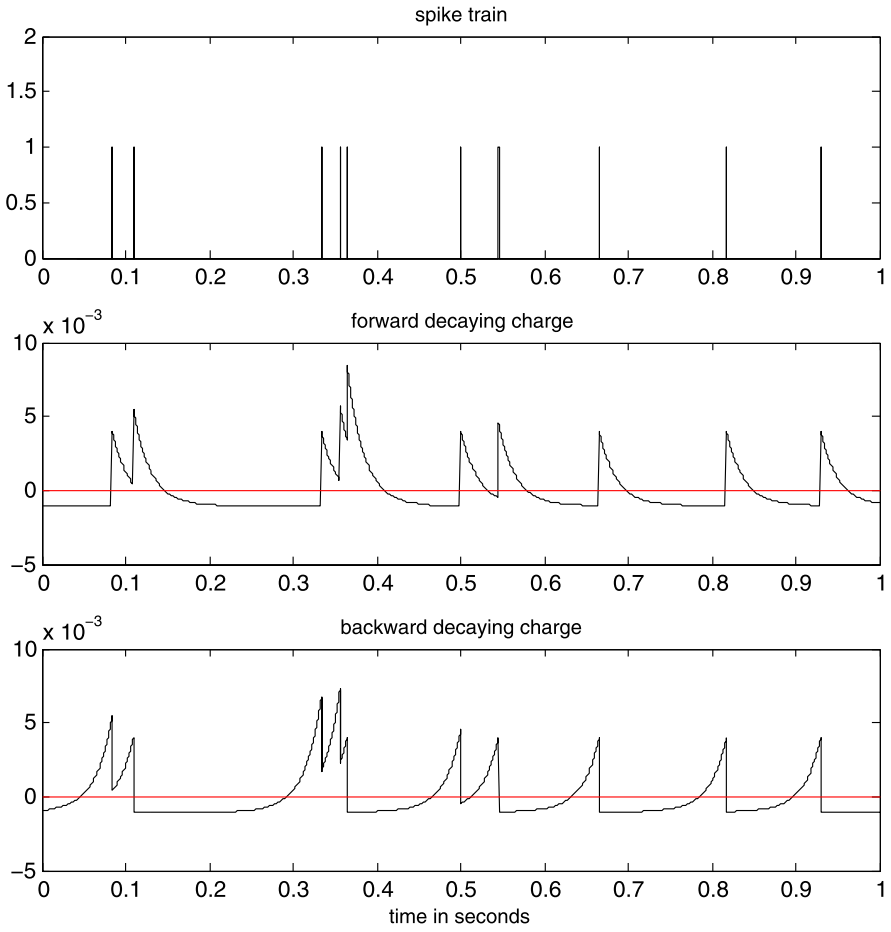

Fig. 8.1 Cartoon of spike train, forward and backward charges using an exponential kernel in convolution with the spike train

resulting particle trajectories are noisier and may be harder to interpret; comparative testing needs to be done.

Discrimination between particle distances corresponding to independent or synchronous neuron pairs depends on the variance of the charge product (8.4). Using a standard statistic for discrimination (Green and Swets 1966)

$$d' = \frac{\overline{Q_{i,j}^{\mathrm{SYNC}}} - \overline{Q_{i,j}^{\mathrm{INDEP}}}}{\mathrm{stdev}(Q_{i,j}^{\mathrm{INDEP}})} \tag{8.5}$$

it turns out (Baker and Gerstein 2000) that discrimination increases with the degree of synchrony and in addition that interval statistics of the spike train are important. For Poisson spike trains, discrimination does not depend on firing rate. For spike

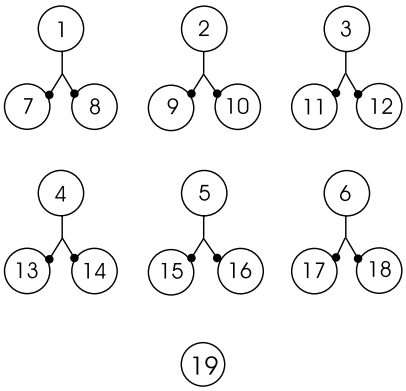

Fig. 8.2 Cartoon of the neuronal groups used in a simulator to produce the data used in subsequent figures of this article. There are six presynaptic neurons, 12 postsynaptic neurons, and one independent neuron. Interval distributions of spike trains are gamma order 4, with rate 10/second

trains with gamma interval distributions, discrimination increases both with firing rate and order of the gamma distribution.

8.3 Visualization of the Gravitational Analysis

With the above equations, there will be aggregation of the particles representing approximately synchronously firing neurons, while particles representing independent neurons will remain roughly near their starting points in the N-space. Such aggregations cannot be directly visualized because they are N-dimensional and their detection is generally a problem in cluster analysis. However, we have found several simple and useful graphical methods that reduce the dimensionality for such visualization.

In order to illustrate the several visualization tools, we generate a data set which has six small groups of neurons, each consisting of a driver and two post synaptic followers and one independent neuron. Individual homogeneous spike trains have a rate of 10 spikes/second and a gamma interval distribution of order four. Synaptic strength between driver and followers has been adjusted to give cross-correlograms, similar to those observed in real recordings, and is the same in all six subassemblies. The effective connectivity used in the simulator is summarized in Fig. 8.2.

The first such method graphs the time course of the distance between particles for each pair of particles. The distance between any two particles in the N-space is of course a vector quantity, but here we will consider only its magnitude. For N neurons, there will be $N(N-1)/2$ of such intrapair distance trajectories. The trajectories for pairs involving independent neurons will remain (noisily) at the initial distance value; trajectories for pairs involving synchronous neurons will slope downward at rates related to the strength of the synchrony.

The pair distance trajectories for this data set are shown in Fig. 8.3. The majority of trajectories remain noisily near the original value and correspond to neuron pairs that are not interacting and therefore without excess near synchronous spike timings. Trajectories for interacting neuron pairs decrease systematically in time,

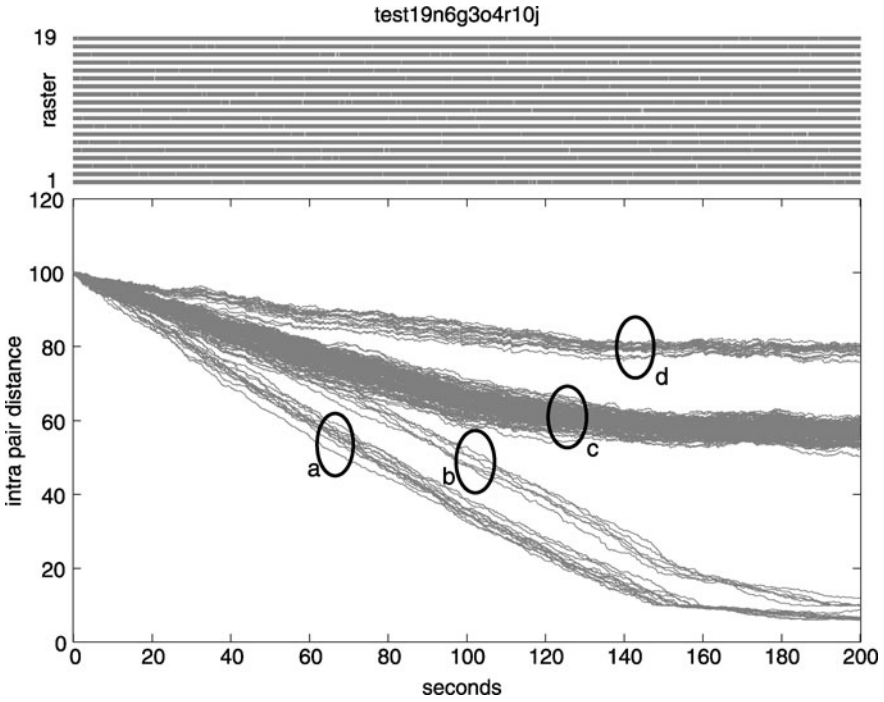

Fig. 8.3 Evolution of intrapair particle distance during the gravity computation. The same kernel is used for acceptor and effector charges, so that both particles in each pair are moved (see text). The labeled trajectory bundles are: (**a**) prepost (12), (**b**) shared input (6), (**d**) independent-other (18), and (**c**) all other pairings (135)

sloping down to the minimum approach distance. Thus the most rapidly aggregating particles with pair distance trajectories involved in bundle **a** result from the 12 direct prepost synaptic interactions of the data set. The slightly slower aggregation of the 6 trajectories in bundle **b** result from pairs with shared input. The least aggregation is seen in the 18 trajectories of bundle **d** where one neuron of each pair is the independent 19. Finally the large trajectory bundle labeled **c** shows some relatively small movement and involves the remaining pairs that are not directly or indirectly related through synaptic activity. These small but systematic movements in bundle **c** are caused by the effect of the strong particle aggregations in the underlying N-space on the geometric projection. Since all six of the three neuron groups were simulated with the same synaptic strength parameters, the aggregating trajectories within bundles **a** and **b** are similar, although not identical, and barely allow the separation of direct synaptic interaction from shared input. Note that if the synaptic parameters in the subgroups had been varied, we would expect to see descending trajectories with more varied slopes and times, shallower slopes corresponding to the weaker synaptic strengths resulting in overlap of bundles **a** and **b**.

It is sometimes useful to display the particle pair velocity by smoothing and taking the time derivative of the trajectories of the pair distance graph. This is shown in Fig. 8.4 with the same four bundle identifications as in Fig. 8.3.

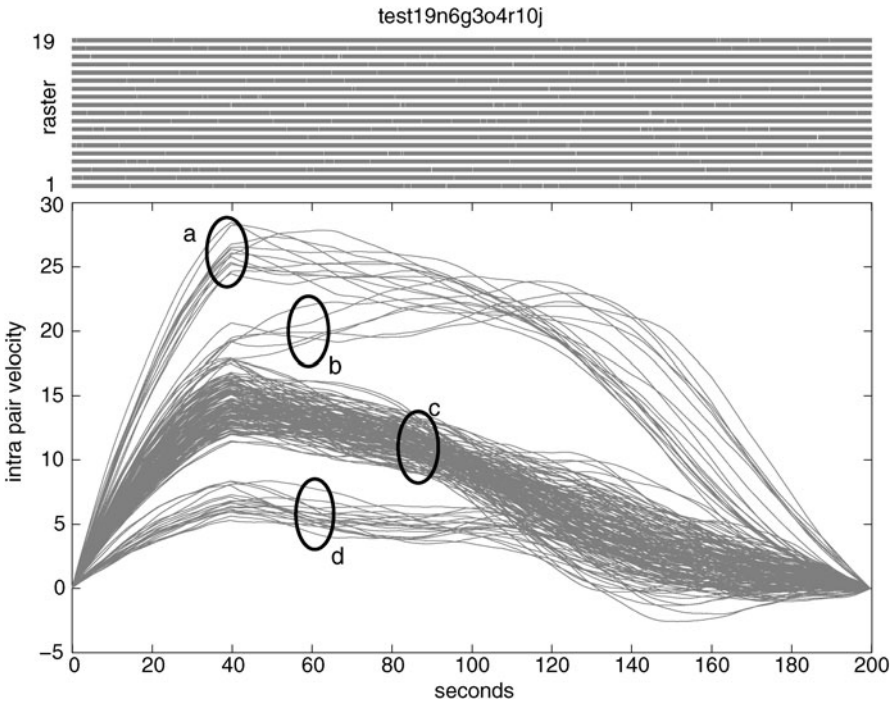

Fig. 8.4 Evolution of intrapair particle velocity during the gravity computation. The picture is somewhat distorted by the low-pass filters used in the velocity calculation, but the several trajectory bundles correspond to those in Fig. 8.3

Projection of the N-dimensional system to a two-dimensional plane loses information, can produce spurious apparent particle aggregation, and must be carefully checked against the pair distance data. However such projection allows direct visualization of the development of the particle movements and their interpretation in terms of neuronal interactions. Such projections can be made in many ways (Sammon 1969; Stuart et al. 2002); we have found a simple geometric projection convenient for small numbers of neurons <20. Here the projection plane is determined by the centers of gravity of various particle combinations and hence is dynamic rather than a fixed plane. An example for the same data used for Figs. 8.3 and 8.4 is shown in Figs. 8.5 and 8.6. The projection of the starting particle positions is shown in Fig. 8.5. In Fig. 8.6 gray projected trajectories for each particle are shown, with the final position indicated by the particle number (1–19 for these data). There are six clusters of three particles, and one noisily stationary particle 19; the clustering is in agreement with the connection diagram (Fig. 8.2) used in the simulator that produced these spike trains. In a more complicated situation with overlapping assemblies (Strangman 1997; Gerstein 1998) a simple strategy to clarify the overlaps is to delete from the data all members of a defined cluster and then recompute, cycling one at a time through all defined clusters.

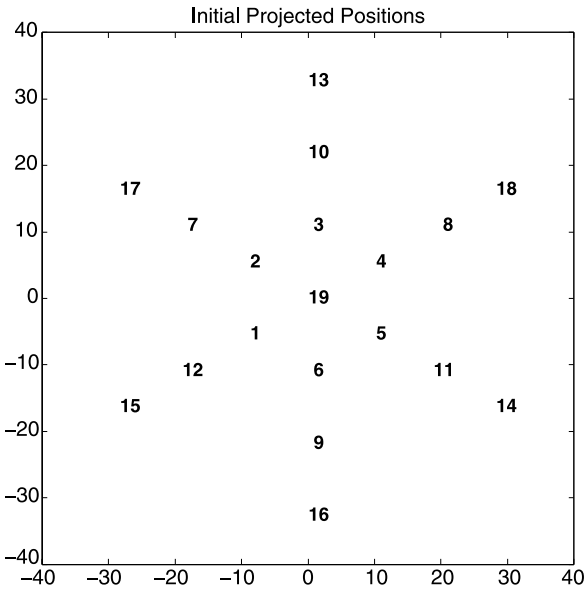

Fig. 8.5 Initial particle positions in a projection to a two-dimensional plane. The plane is determined by a function of the current particle coordinates in the original N-space and hence varies during the progress of the computation

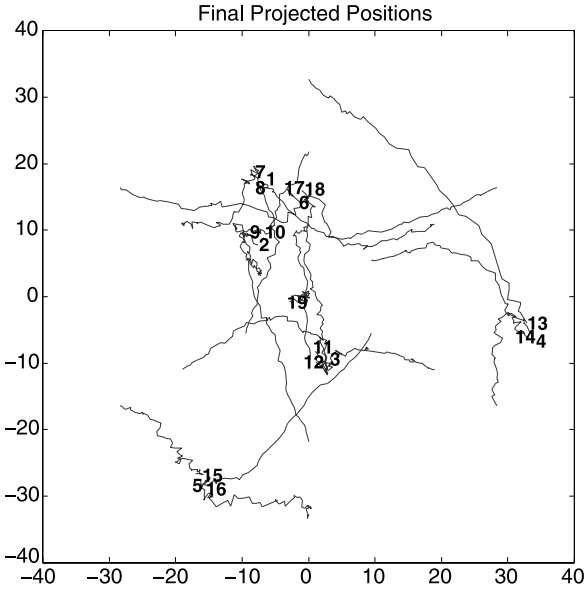

Fig. 8.6 Final particle positions and their trajectories in a projection to the two-dimensional plane

Note also that the gravity calculation itself is dynamic. Aggregations can proceed at varying rates, may pause and even disaggregate. This is seen by changes of slope in trajectories of pair distance (Fig. 8.3) or in the process of forming the clusters in the projection display. Such behavior is best seen if the projection display is viewed as a movie.

8.4 Significance Testing for Distance Trajectories

For noninteracting neurons, the bundle of corresponding pair distance trajectories will remain noisily near the starting value, with gradually widening envelope. In contrast, trajectories corresponding to interacting neuron pairs will steadily slope downward, and it is important to have a significance test to properly distinguish the two situations. Unfortunately, no analytic significance test has yet been devised. We must resort to Monte Carlo methods with surrogates. Surrogates of the original data are best produced by shuffling or shifting in any way that destroys temporal relations on the time scale of physiological neuronal interactions but does not alter the general rate structure or any other characteristics of each spike train. See Chap. 17 for more general surrogate issues. If there is no rate modulation in the data, an adequate surrogate is to shift each spike train by some random amount larger than say 100 ms. For data with a known stimulus structure, the shifts should be on known stimulation boundaries, i.e., by an integral number of stimulus intervals. To avoid losing data, shifts are made circular. By generating and running the gravity analysis on 100 such surrogates we get a one percent envelope of the trajectories for (artificially) noninteracting neuron pairs. Examples are shown in Lindsey et al. (1992a, 1992b) and Lindsey and Gerstein (2006).

Note that if shifting on known stimulus boundaries, it is necessary to avoid repeating a particular relative shift between two particular spike trains in different surrogates. With numbers like 20 neurons and 100 stimuli, this is not difficult but requires some care in making programs that produce surrogates.

8.5 Variations on the Basic Gravity Computation

8.5.1 Forward and Backward Charges

In the basic gravity computation all particle charges use the same kernel as shown, for example, in the middle panel of Fig. 8.1. In these conditions two particles representing neurons with some near-synchronous activity will move toward each other irrespective of the order of firing since for appropriate delays, the two charges will overlap. The gravity computation can be made sensitive to firing sequence by letting each particle carry two charges, one for use when it is the particle being moved (the i index in (8.3), called the "acceptor charge"), and one for use when it is one of the other particles that are causing the movement (the j index in (8.3), called the "effector charge"). If the charge on the i particle is like the middle panel in Fig. 8.1 (a forward exponential) and the charges on the j particles are like the bottom panel of Fig. 8.1 (a backwards exponential), the charges will overlap only if the neuron corresponding to the i particle fires before the ones represented by the j particles, i.e., if it is presynaptic to them. Thus, in the N-space, the i particle will move toward the relevant j particles, while the j particles stay noisily in place. If the charge definitions are reversed, the i particle will move only when its neuron fires after the ones represented by the j particles, i.e., if it is postsynaptic to them. Additional details and examples are in Gerstein and Aertsen (1985).

8.5.2 Nonlinear Functions of Charge Product

The force driving the particles in the N-space is proportional to the charge product $Q_{ij} = q_i q_j$ as in (8.3) and (8.4). It is sometimes convenient to make the force a nonlinear function of the charge product. In this case we modify (8.3) so that

$$\mathbf{x}_i(t + \partial t) = \mathbf{x}_i(t) + \kappa \partial t \sum_{\forall j \neq i} f(Q_{i,j}) \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|}. \tag{8.6}$$

Here $f(u)$ is a nonlinear function with a slope which increases its magnitude with increasing distance from $u = 0$. This will have the effect of suppressing small, noisy fluctuation relative to the large excursions in $Q_{i,j}$ that are unlikely to have been caused by chance. The simplest form of suitable nonlinearity is

$$f(u) = A|u|^m \operatorname{sign}(u), \tag{8.7}$$

where the power m can be varied to change the functional form; use of this form permits even and nonintegral m whilst ensuring that the sign of $f(u)$ always equals the sign of u. The constant A permits scaling of the nonlinear function.

It is very advisable to apply a moving window smoothing to the charge product $Q_{i,j}$ when using a nonlinearity as in (8.6) and (8.7). This avoids degradation of performance by excess standard deviation of the charge product. A detailed analysis of performance under various levels of synchrony (the d' of (8.5)) and for various powers m is given in Baker and Gerstein (2000). Essentially for small d', i.e., low levels of synchrony, the nonlinear formulation gives no improvement over the original version of (8.3). For d' values greater than 1.5, there is considerable improvement of performance.

For practical application of the algorithm, we have found that it is advisable to initially scale $Q_{i,j}$ by its standard deviation:

$$Q'_{i,j}(t) = \frac{Q_{i,j}}{\operatorname{stdev}(Q_{i,j})}. \tag{8.8}$$

Such a scaling makes the variability of all charge products the same, regardless of cell firing rate or interspike interval statistics. By normalizing for this first, the choices of constant κ in (8.3) can be standardized, and the same value used across different datasets. This reduces the need to "tune" performance of the algorithm by testing multiple values of κ, although it may still be necessary to explore various kernels that can be used to convert the spike trains into charge. The standard deviation of the charge product can be calculated over the entire length of the data if the firing rate of the neurons is stationary, or by using a moving window if not (compare with the calculation of mean charge using a moving window in nonstationary data, Gerstein and Aertsen 1985). Normalization by standard deviation as a means to correct for nonstationarity has also been used by Aertsen et al. (1987, 1989).

8.5.3 Temporal Clustering of Pair Interactions

In the basic gravity computation synchrony in the firing of two neurons creates an attractive force between the two corresponding particles and causes their aggregation. If however these two neurons are members of a larger synchronous assembly, we may expect that additional particle pairs will also coalesce into the same cluster in the N-space.

Suppose that three neurons, i, j, and k, are members of the same assembly. The paired synchronous firings (i, j), (i, k), and (j, k) will each occur at above chance rates, and this will lead the three corresponding particles to move toward each other. However, since all three neurons are part of the same assembly, we might expect an excess temporal clustering of the several pair coincidences within a moving window of length w; such counts of multiple pair coincidences are highly unlikely to occur by chance. The closely spaced occurrence of multiple pair synchronies will be a much more robust indicator of assembly membership than any one of the individual pairwise synchronies alone. Note that this is not a measure of triple coincidences since we are looking for closely spaced but not synchronous pair occurrences. See Chap. 12 for calculation of high-order coincidences.

These ideas suggest a further enhancement to the particle movement rule of (8.6):

$$\mathbf{x}_i(t+\partial t) = \mathbf{x}_i(t) + \kappa\,\partial t \sum_{\forall j \neq i} f\left(Q'_{i,j}\right) g\left(\sum_{\forall k \neq i,\, j} Q'_{i,k} Q'_{j,k}\right) \frac{\mathbf{x}_j - \mathbf{x}_i}{|\mathbf{x}_j - \mathbf{x}_i|}. \quad (8.9)$$

The time window w for the two-pair product term is implicit in the kernel used to calculate the several charge functions; it may be useful to use a kernel with a longer time constant for the pair product term in g than used for the single argument of the f function. The scaling function $g(u)$ should be always positive with a minimum of zero and a small integer maximum; also we require that $g(0) = 1$ with a slope near zero in a region around $u = 0$. Many such functions are possible; an example is given in Baker and Gerstein (2000).

If no other neuron fires near-synchronously with cells i and j during the window period w, the summation over k in (8.9) will be close to zero; g will be nearly one, and the basic gravity algorithm will be unaffected. However, if another cell k does participate in the assembly in the period w, the summation will on average be above zero, since both $Q'_{i,k}$ and $Q'_{j,k}$ will have nonzero, positive mean. The force causing aggregation between i and j will therefore be enhanced. By symmetry, a similar effect will occur for the force between particle pairs (i, k) and (j, k), and all three particles will therefore aggregate more rapidly than had the term in $g(u)$ not been included. Examples and performance measures are given in Baker and Gerstein (2000). It turns out that both the nonlinear and the multiple pair versions enhance aggregation if the interval structure of the spike trains is given by a gamma function of low order. However, there is no enhancement and even degradation of performance over the basic linear gravity algorithm if the spike trains are Poisson. It is apparently easier to detect synchronies among neurons that produce gamma interval distributions (i.e., with dead time) as compared to Poisson interval distributions.

8.6 Tuned Gravity

The usual application of the gravity analysis is as a preliminary screening tool to identify neuronal groupings defined by near synchronous firing. Once identified, the spike trains from neurons in such groups may be studied by various additional detailed spike train analysis methods such as joint peri-stimulus histograms (JPSTH) (Aertsen et al. 1989) or cross-correlation. The detection of near-synchrony in the gravity computation depends critically on the kernel chosen for the initial conversion of spike train into a charge function. In particular, the kernel must cover the expected range of near-synchrony. This poses a problem if there are long but fairly exact delays among the different neuron firings, i.e., "delayed synchrony". The usual limited duration kernels would simply be blind to such delayed interactions since there would never be charge overlaps. If a kernel long enough to encompass the delays is used, the overall computation will become excessively noisy and will also lose time resolution. One way to deal with such a situation without losing sensitivity to ordinary near-synchrony is to do a preanalysis with cross-correlation.

All pairs in the data set are analyzed with cross-correlation. When an offset peak corresponding to a delayed interaction is noted, the kernels used for that particular pair of spike trains are offset by the same amount but only when they both appear respectively as i and j in (8.3). Kernels used for all the other i and j combinations remain in their normal un-shifted form. The gravity computation and consequent particle aggregations will now be sensitive both to the usual near synchronies but also to the longer delays between certain neurons as detected through the preliminary cross-correlation step.

More detail and examples are given in Lindsey and Gerstein (2006).

8.7 Repeating Synchrony Patterns and Time Markers

The aggregation process in the gravity computation reflects near synchronies among the recorded neurons in a dynamic way. This is evident in short periods of high slope in the pair distance display and is even more evident in the corresponding pair velocity display. Often multiple pairs show transient rapid aggregation (high velocity) during the same short period, representing the transient activation of a particular neuronal assembly. Such epochs can be identified by setting an amplitude criterion in the pair velocity display or alternatively a slope criterion in the pair distance display. A small amount of smoothing may reduce noise in such a procedure. The several neurons participating in each of the selected epochs define multiple synchrony patterns. It is then easy to compare these patterns and search for repeats that might be expected in any real repetitive stimulation or behavioral paradigm. Particular repeating synchrony patterns may be associated with some particular phase of the paradigm. Details and illustrations of these ideas are in Lindsey et al. (1997).

There is another interesting approach to the relation between the particle aggregation process and time markers of events like stimuli in the laboratory. Conceptually, we want to average each pair distance trajectory relative to the time markers

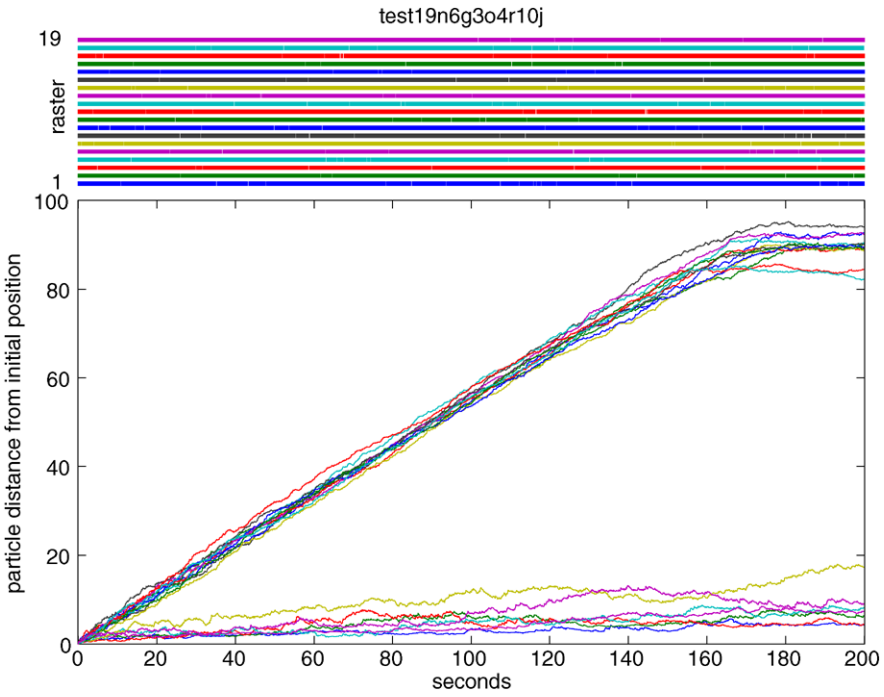

Fig. 8.7 Distance each particle has moved from its original position during the computation. In this case the acceptor charges were taken as forward and effector charges as backward (see Fig. 8.1), so that only postsynaptic particles move, while the presynaptic and independent particles are noisily stationary

and over some short time window like 1 or 2 seconds. This would detect any significant modulation of the synchrony process by the marked laboratory events. For each trial entering into this average, we set the pair distance value at the marker to zero, thus avoiding the long slope over the course of the long duration of the data set. Computationally, this process is better carried out at the level of the charge product. Thus we calculate the event time-locked average of the $Q_{i,j}$ or, better, its variance-normalized version $Q'_{i,j}$ of (8.4) and (8.8). These are then used in the basic (8.3) or enhanced versions ((8.6) and (8.9)) of the computation. Details and examples are given in Baker and Gerstein (2000).

8.8 Other Visualizations

Two final useful visualization tools for the "move postsynaptic" situation discussed above in the "forward and backward charge" section are available in both two-dimensional and three-dimensional displays. We start with the same data as used for Figs. 8.3–8.6 but with parameters set so that only the postsynaptic particle moves during the gravity computation. In the two-dimensional display in Fig. 8.7, we plot

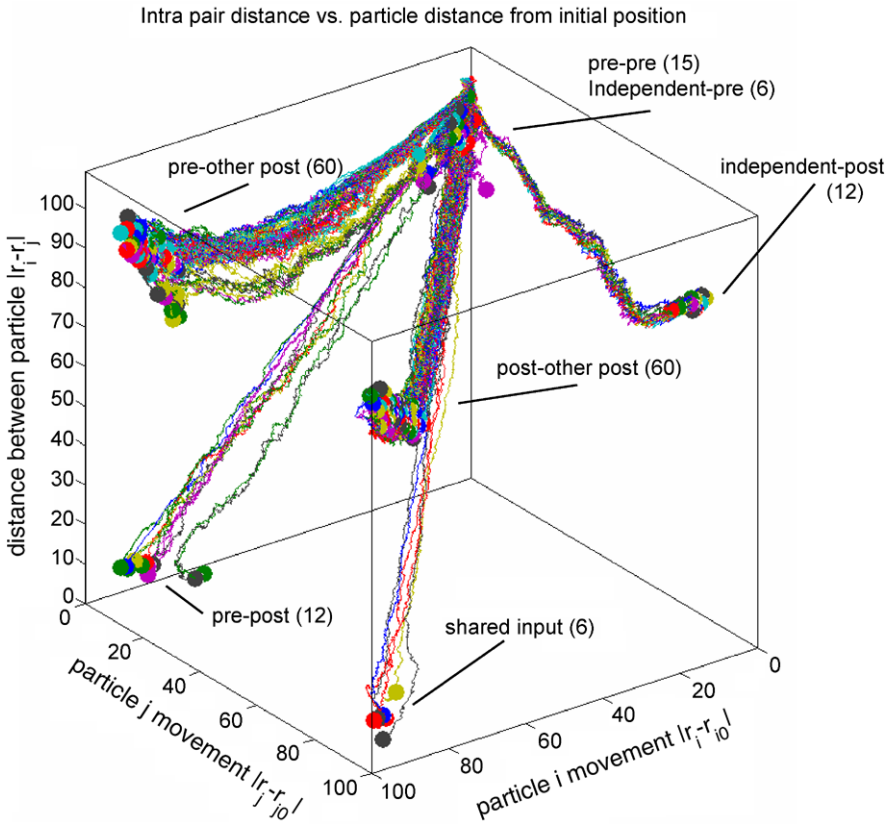

Fig. 8.8 A three-dimensional display of the same data as used for Fig. 8.7. The **x** coordinate is distance the *i* particle has moved from its initial position, the **y** coordinate is the distance the *j* particle has moved from its initial position, and the **z** coordinate is the distance between the *i* and *j* particles. The identities of the several distinct bundles of trajectories are labeled on the figure

the distance of each particle from its initial position as the gravity computation proceeds. The lower bundle of trajectories shows the seven particles that hardly move from their initial position (the six drivers 1–6 and the independent 19). The upper bundle of trajectories contains the movements of all the postsynaptic particles (7–18).

For the three-dimensional display shown in Fig. 8.8, we expand on the idea of particle distance from initial position. Here we plot trajectories for each particle pair using three coordinate values: the distance of particle *i* from its original position along **x**; the distance of particle *j* from its original position along **y**; and the distance between particles *i* and *j* along **z**. Distinct bundles of trajectories are formed in the 3D plot and are labeled according to the timing (synaptic) relations of the corresponding neuron pairs. More detail is given in Lindsey and Gerstein (2006).

8.9 Conclusion

The gravity transformation is a useful tool for initial screening of a multi-single-neuron recording. By processing all available spike trains at once it can rapidly identify neuron groupings based on near-synchrony of firing. The process is dynamic, so that temporal modulations and relative strengths of the neuronal associations are easily made visible.

Interesting neuronal groupings can then be examined more thoroughly by various other appropriate analytic tools.

This article has presented a brief overview of the basic gravity transformation with appropriate visualization tools and significance testing, and has described a number of enhancements that may be useful for particular data sets. Most of these enhancements deal with improved sensitivity, but it should be noted that with some of them performance will depend on the types of interval distributions in the spike trains.

A somewhat primitive suite of programs to perform the gravity transformation together with a "readme" and some test data sets is available for downloading at http://mulab.physiol.upenn.edu or http://www.apst.spiketrain-analysis.org/ and may be freely used and modified as desired. Although these programs can handle any number of neuronal spike trains, the projection of the N-space to a two-dimensional plane is limited to 10 neurons. The relevant projection algorithm may be examined in the Fortran program "prjtmaml5.for" in the "fsources" directory of the download and easily extended for larger numbers of neurons as a standalone program in new programming languages (preferably directly in Matlab).

References

Aertsen AMHJ, Bonhoeffer T, Kruger J (1987). In: Caianiello ER (ed) Physics of cognitive processes. World Sciences Publishers, Singapore, pp 1–34

Aertsen AMHJ, Gerstein GL, Habib MK, Palm G (1989) Dynamics of neuronal firing correlation – modulation of effective connectivity. J Neurophysiol 61:900–917

Aertsen A, Vaadia E, Abeles M, Ahissar E, Bergman H, Karmon B, Lavner Y, Margalit E, Nelken I, Rotter S (1991) Neural interactions in the frontal-cortex of a behaving monkey – signs of dependence on stimulus context and behavioral state. J Hirnforsch 32:735–743

Arata A, Hernandez YM, Lindsey BG, Morris KF, Shannon R (2000) Transient configurations of baroresponsive respiratory-related brainstem neuronal assemblies in the cat. J Physiol 525(2):509–530

Baker SN, Gerstein GL (2000) Improvements to the sensitivity of gravitational clustering for multiple neuron recordings. Neural Comput 12:2597–2620

Dayhoff JE (1994) Synchrony detection in neural assemblies. Biol Cybernet 71:263–270

Gerstein GL (1998) Correlation based analysis methods for neural ensemble data. In: Nicolelis M, Simon S, Corless J (eds) Methods for simultaneous neuronal ensemble recordings. CRC Press, Boca Raton, pp 157–178

Gerstein GL, Aertsen AMHJ (1985) Representation of cooperative firing activity among simultaneously recorded neurons. J Neurophysiol 54:1513–1528

Gerstein GL, Perkel DH (1972) Mutual temporal relationships among neuronal spike trains. Biophysical J 12:453–473

Gerstein GL, Perkel DH, Dayhoff JE (1985) Cooperative firing activity in simultaneously recorded populations of neurons: detection and measurement. J Neurosci 5:881–889
Gerstein GL, Bloom MJ, Maldonado PE (1998) Organization and perturbation of neuronal assemblies. In: Brugge J, Poon P (eds) Central auditory processing and neural modeling. Plenum, New York
Gochin PM, Gerstein GL, Kaltenbach JA (1990) Dynamic temporal properties of effective connections in rat dorsal cochlear nucleus. Brain Res 510:195–202
Green D, Swets J (1966) Signal detection theory and psychophysics. Wiley, New York
Grün S, Diesmann M, Aertsen A (2002) Unitary events in multiple single-neuron spiking activity: detection and significance. Neural Comput 14:43–80
Hebb DO (1949) The organization of behavior. Wiley, New York
Lindsey BG (2001) How is the respiratory central pattern generator configured and reconfigured?. Adv Exp Med Biol 499:179–184
Lindsey BG, Gerstein GL (2006) Two enhancements of the gravity algorithm for multiple spike train analysis. J Neurosci Methods 150:116–127
Lindsey BG, Shannon R, Gerstein GL (1989) Gravitational representation of simultaneously recorded brain stem respiratory neuron spike trains. Brain Res 483:373–378
Lindsey BG, Hernandez YM, Morris K, Shannon R, Gerstein GL (1992b) Dynamic reconfiguration of brain stem neural assemblies: respiratory phase-dependent synchrony vs. modulation of firing rates. J Neurophysiol 67:923–930
Lindsey BG, Hernandez YM, Morris K, Shannon R, Gerstein GL (1992a) Respiratory related neural assemblies in the brain stem midline. J Neurophysiol 67:905–922
Lindsey BG, Morris KF, Shannon R, Gerstein GL (1997) Repeated patterns of distributed synchrony in neuronal assemblies. J Neurophysiol 78:1714–1719
Maldonado PE, Gerstein GL (1996) Neuronal assembly dynamics in the rat auditory cortex during reorganization induced by intracortical microstimulation. Exp Brain Res 112:431–441
Martin PD (2001) Locomotion towards a goal alters the synchronous firing of neurons recorded simultaneously in the subiculum and nucleus accumbens of rats. Behav Brain Res 124:19–28
Morris KF, Baekey DM, Nuding SC, Segers LS, Shannon R, Lindsey BG (2003) Neural network plasticity in respiratory control. J Appl Physiol 94:1242–1252
Pauluis Q, Baker SN (2000) An accurate measure of the instantaneous discharge probability, with application to unitary joint-event analysis. Neural Comput 12:647–669
Sammon JW Jr (1969) A non-linear mapping for data structure analysis. IEEE Trans Comput C 18:401–409
Strangman G (1997) Detecting synchronous cell assemblies with limited data and overlapping assemblies. Neural Comput 9:51–76
Stuart L, Walter M, Borisyuk R (2002) Visualisation of synchronous firing in multi-dimensional spike trains. BioSystems 67:265–279
Wright WE (1977) Gravitational clustering. Pat Recogn 9:151–166

Part III
Multiple-Neuron Spike Patterns

Chapter 9
Spatio-Temporal Patterns

Moshe Abeles

Abstract Precise time relations among spikes was reported by many authors and criticized by many. This chapter describes how such patterns may be detected and how to assess their statistical significance. The stress is on practical aspects of the methodology. The chapter illustrates examples of analysis of real data that were recorded in the cortex of behaving monkeys.

9.1 Introduction

There are numerous reports in the literature describing the existence of precise spatio-temporal patterns in recordings from the cerebral cortex. The earliest ones were probably: Klemm and Sherry (1981); Abeles (1982); Dayhoff and Gerstein (1983); Landolt et al. (1985); and Legendy and Salckman (1985). Many others were published since then. My own observations of precise patterns involving 3 neurons with long delays in between led to the hypothesis that cortical activity may be organized in groups of neurons firing synchronously. Each such group excites the next group by multiple converging diverging connections. A chain of such groups with essentially feed-forward was called a synfire chain. (See Chaps. 6 and 7 of Abeles 1982, for the first explicit description of the findings and the synfire chain model.) The hypothesis that cortical activity is generated by synfire chains predicts the existence of such patterns. Yet methods to reliably detect such patterns are still lacking. To better understand the progress that has been made and the obstacles which need to be overcome, three issues must be addressed as regards any study of spatio-temporal patterns. The first two are methodological: how were the patterns detected, and which statistics were used to rule out chance occurrence? The third is

M. Abeles (✉)
The Leslie and Susan Gonda (Goldschmied) Multidisciplinary Brain Research Center, Bar-Ilan University, Ramat-Gan 52900, Israel
e-mail: abelesm@mail.biu.ac.il
url: http://www.biu.ac.il/interdis/gondabrain/researchers/ABELES%20MOSHE.html

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_9, © Springer Science+Business Media, LLC 2010

both empirical and theoretical and involves showing that observed spatio-temporal patterns of activity in the cortex are a necessary ingredient of brain activity and not simply an epiphenomenon.

This chapter deals only with the first two issues. The first section describes the various types of precise spatio-temporal patterns and how they can be detected. The second and third parts explore the statistical methods for evaluating the probability that the detected patterns are due to chance.

9.2 Types of Precise Spatio-Temporal Patterns

Firing patterns may also be produced by internal ionic mechanisms of single neurons. The simplest examples are the periodic firing of a pace maker or the internal structure of a burst of spikes. Firing patterns may be generated by a network of neurons. This chapter only deals with patterns that may be attributed to the network activity, and therefore, it relates only to patterns whose spikes are elicited by different neurons. Note that this is a very conservative approach as a pattern such as "*neuron A fired, then neuron B fired and then neuron A fired again*" may well be generated by a network.

The simplest cross-neuronal pattern one can think of is the precise pairwise synchrony. Such patterns were explicitly described in the visual cortex of cats by Toyama et al. (1981). However, synchronous spiking drew much attention only when it was suggested that they are associated with the binding of different visual elements into an object (Eckhorn et al. 1988; Gray and Singer 1989). Zero-lag correlations may be clearly seen when computing the cross-correlations between the firing times of two neurons (see Chaps. 5, 6). Often, they are so strong that no statistics is needed to convince the researcher that they exist. Figure 9.1 illustrates an extreme but very rare case. In our lab we encountered 2–3 occurrences out of thousands of cross-correlations.

The requirement for a very tight correlation is often relaxed and correlations within a few ms are also classified as zero-lag correlations. S. Grün termed these correlations "unitary events" and analyzed them extensively (Grün et al. 1999, Chap. 10).

More broadly, a pairwise pattern can be defined as a tight synchronization with a delay. This type of pattern, as shown in Fig. 9.2, is more frequent although not often reported since there is no a priori reason to look for precise synchrony at a lag of 70 ms (and not at any other lag). Judging whether this type of peak is statistically significant is problematic. Figure 9.2 shows one way to attempt to assess the statistical significance of such a peak which is dealt with in more detail the next section. Note that the correlogram in Fig. 9.2 is by no means a very extreme case. Many cases in which the delayed peaks are much more extreme have been found.

A general definition for a precise firing pattern is an N-tuplet pattern composed of N spikes $\{1, 2, \ldots, N\}$ with delays of $\{t_2 \pm \delta_2, t_3 \pm \delta_3, \ldots, t_N \pm \delta_N\}$ from the first spike. The time tolerance around the delay ($\pm\delta$) may be different for each spike (e.g., Tetko and Villa 2001), but typically it is kept constant and is set at ± 0.5 or

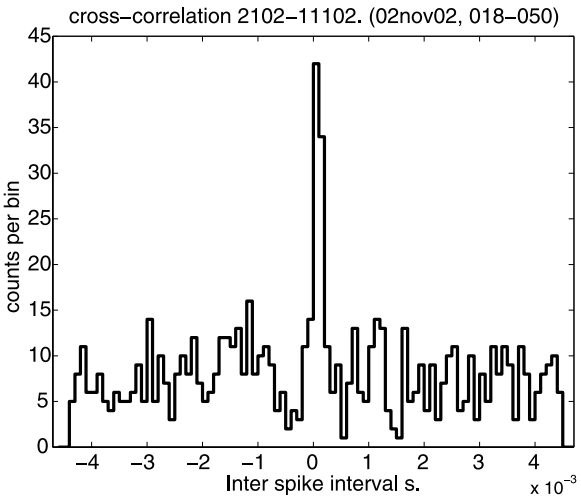

Fig. 9.1 *Zero-lag correlation*. Cross-correlation between two well-isolated single units recorded through two separate microelectrodes in the premotor cortex of a behaving monkey. The bin size is 0.1 ms (and this was the resolution with which the spike time was recorded). The center peak is less than 0.5-ms wide. The distance between the electrode tips was ∼5.6 mm, so the peak cannot be due to both electrodes recording the same neuron. Careful inspection of the electrode records in wide band (1–6,000 Hz) shows clearly that the coincident spikes were not artifacts. No relation between the synchronous spikes and behavior was found

± 1.5 ms. The maximal allowed delay t_N is set a priori. Here, this type of sequence is termed a *precise firing sequence* (*PFS*) *of order N and tolerance* δ.

When the activity of K ($K \geq N$) neurons over a duration of T seconds is measured, there are $\binom{K}{N}$ types of PFS of order N composed of N different neurons, and for each type, there are $(t_N/2\delta)^{N-1}$ different time delays. If the neurons fire independently at fixed rates of $\{\lambda_i\}$, $i = 1, 2, \ldots, N$, the expected number of PFSs of a given composition and time delays is

$$x_N = T \cdot \lambda_1 \cdot \prod_{i=2}^{N} \lambda_i \cdot 2\delta.$$

Unless this expected number is small, there are likely to be a huge number of patterns occurring by chance, and it is difficult to differentiate the real ones from the random ones. If the firing rates are high or the tolerance is high, so that the product $2\lambda \cdot \delta$ is not much smaller than 1, the researcher should start looking for patterns that repeat at least R times so that $prob\{R \text{ or more} \mid x_N\}$ is small. This probability can normally be safely estimated by assuming that the counts for a particular pattern are distributed in a Poissonian fashion. If so distributed, then

$$prob\{R \text{ or more} \mid x\} = \sum_{i=R}^{\infty} e^{-x} \cdot x^i / i! = 1 - \sum_{i=0}^{R-1} e^{-x} \cdot x^i / i!.$$

See the next section for a validity of the formula above.

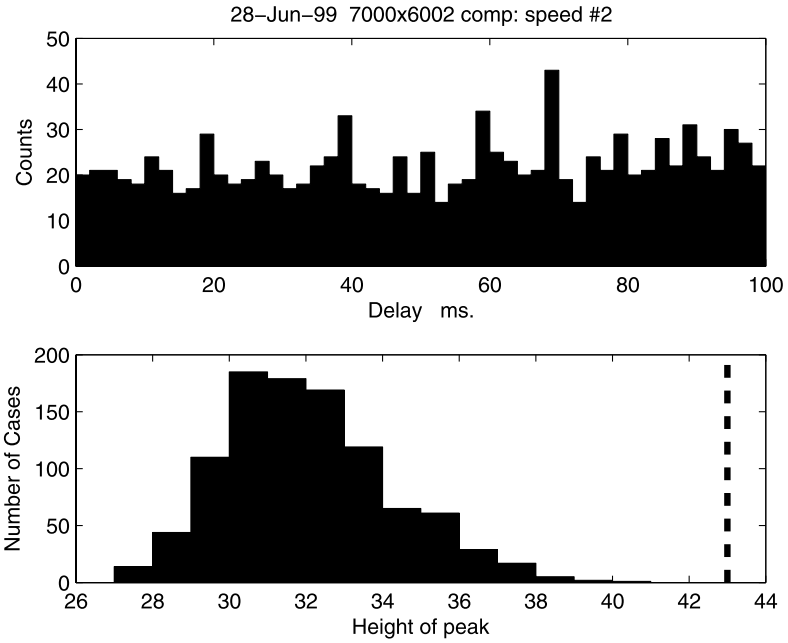

Fig. 9.2 *Example of precise lagged synchrony*. *Top* – cross-correlation between two premotor neurons of a monkey while freely scribbling. The highest peak is at a delay of 70 ms where 43 counts were found. This does not appear to be particularly significant. *Bottom* – the spike trains were teetered within ±3.5 ms, cross-correlated with the exact same parameters (bin size and maximal delay) as for the original trains. In each correlation the highest count was located. This was iterated 1,000 times, and a histogram of the highest count in each correlogram noted. Clearly the probability of obtaining 43 counts by chance with a spiking accuracy of less than 7 ms is below 0.001

As an example with realistic values, assume that we are dealing with 3 ($N = 3$) units whose activity was measured for 200 s, the neurons fired at 5 Hz, and we want to detect PFSs with tolerance of ±1.5 ms. The expected count is $200 \cdot 5 \cdot (5 \cdot 0.003)^2 = 0.225$, which is small, and a few repetitions of the very same pattern could easily be detected (e.g., the probability of finding 5 or more repetitions when the expected is 0.225 is below one in 300,000). However, if the firing rates were 10 Hz, and the tolerance ±5 ms, the expected count would be 20, and there would be hardly any point in looking for triplets that returned a few times more than expected ($prob\{25 \text{ or more} \mid 20\} > 0.15$). If we were looking for quintuplets, the expected count would be $200 \cdot 10 \cdot (10 \cdot 0.01)^4 = 0.2$.

The main stumbling block is to find an algorithm which (after deciding on the desired: tolerance, max duration of a PFS, its order and the minimal number of repetitions desired) can find all PFSs that meet these limits. An early algorithm of this type was the "sliding tape" algorithm suggested by G.L. Gerstein (Abeles and Gerstein 1988). In this algorithm the data are depicted as though they were lying along a long paper tape. The firing of each neuron is marked by holes in one line

along the tape (this would look very much like the old paper tape used to feed data to a computer). Now imagine that two copies of the same tape are slid past each other. Whenever there is a repeated constellation of holes, they would be laid one on top of the other (after appropriate shift). Thus this algorithm detects all patterns that repeat twice or more. In step one, a list is made of the PFSs that meet the criteria for total duration and minimal complexity, and step two identifies those that repeated above the minimal desired number (see Abeles and Gerstein 1988, for more details). However, as this algorithm starts with all patterns that repeated twice, the initial conditions should be such that the expected rate of such N-tuplets is small.

When the expected number is small, then there is no limit to the complexity of the searched pattern (N). Any N overlapping holes within t_n seconds in the two tapes will be discovered. However, so far, when N was large, we found only very few repetitions of patterns, and it may be impossible to reliably associate between the patterns and behavior (or stimuli). A personal advice is not to go beyond quintuplets, unless there is a large number of repeating quintuplets. (If there is one excessively repeated N-tuplet, then there might be N-choose-5 excessively repeating subpatterns of order 5.)

A limited scope may be searching for triplets only (Abeles and Gat 2001), i.e., triplets of the form $\langle u_1, u_2, u_3; t_2 \pm \delta, t_3 \pm \delta \rangle$ where u_i is the id of the neuron, and t_i is the delay of spike of neuron i from the firing of the first neuron. If the maximal delay is $\boldsymbol{\tau}$, we define a table with $\boldsymbol{\tau}/2\delta$ by $\boldsymbol{\tau}/2\delta$ square bins. Whenever there are 3 spikes from u_1, u_2, u_3 within $\boldsymbol{\tau}$, we add one to the appropriate bin. An example of such a histogram is illustrated in Fig. 9.3a.

If the two delays from the first spike are $t_2 \pm \delta$, $t_3 \pm \delta$, then the delay between a spike from the second neuron to a spike of the third neuron can be in the range of $t_3 - t_2 \pm 2\delta$. This may be avoided by splitting the square bins into two triangles, making sure that no delay range is above $\pm\delta$. This was done by Abeles (1983) but was not exploited much in the literature.

Finally, a sequence of synchronized groups of neurons may be detected by the appearance of oblique "worms" when the similarity between the activity of many spikes recorded in parallel at time t_1 and the activity at time t_2 is plotted on a t_1 vs. t_2 plan (Schrader et al. 2008).

9.2.1 Statistical Significance

It was justifiably pointed out that if you record enough data, you may occasionally find a PFS that repeats many times such as in Figs. 9.2 or 9.3 (Oram et al. 1999). Thus, the crucial point is to assess the probability that such a pattern occurred by chance.

Typically, we record from several neurons in parallel and have no prior knowledge of the time delays involved in an excessively repeating pattern. Thus, we attempt to find patterns by testing a large number of possibilities. If the recording was

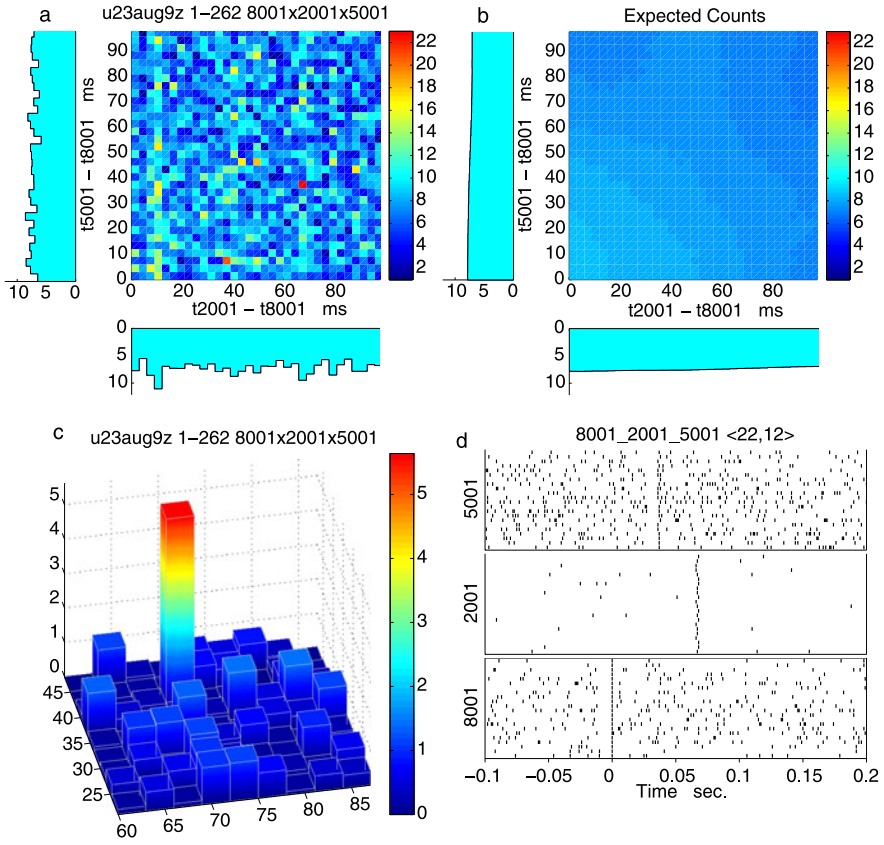

Fig. 9.3 3-*fold patterns*. Recordings were from the premotor area of a monkey. Data were parsed so that only pieces around a certain drawing arc-shape were left. The spike trains were first filtered so as to leave only the first spike in a burst. The data here relate to three well-isolated spike trains recorded by three different microelectrodes. Bin size was $2\delta = 3$ ms. **a**. A flat view of the counting matrix from above. Two patterns with delays of $(40, 10)$ and $(70, 40)$ ms from the trigger spike stand out (in *red*). **b**. A flat view from above of the expected counts, as obtained by convolving the matrix in (**a**) with a Gaussian kernel having standard deviations of five bins with the central value set to 0 (similar smoothing could be achieved by repeatedly teetering the spike trains within ± 6 bins and computing the average histogram). **c**. A 3D view of a small region around the bin at $(40, 70)$. The Z-axis describes the surprise (minus log probability) of finding so many or more in a bin given the expected number. At the peak it reaches 5.5 meaning one in 300,000. **d**. Dot display of all the spikes around the occurrence of the pattern $\langle 8001, 5001, 2001; 40 \pm 1.5, 70 \pm 1.5\rangle$. All spikes in the period are shown (no "burst filtering" applied before plotting). Time 0 designates the time of the first spike of the pattern. As all of them are aligned, they form a straight vertical line. Indirect evidence that these were not merely chance events may be derived from the following observation. When chance spikes are picked, the density of spikes before and after them should be the same (the autocorrelation is symmetric). Here, a short period just before the spikes of 8001 and 5001 in the pattern is quiescent, but the immediate period after them is not. So they cannot be randomly picked spikes. A similar asymmetry was reported in Shmiel et al. (2006)

made from K neurons and we are looking for triplets with delays up to τ and resolution of $\pm\delta$, we are searching for excessively repeating triplets in one out of N possibilities:

$$N = \binom{K}{3} \cdot \left(\frac{\tau}{2\delta}\right)^2.$$

Take, for example, 10 neurons, max delay of 0.3 s, and bin of ± 1.5 ms. This would amount to 1,200,000 possible different PFSs. Obviously there would be a vast number of (around 12,000) triplets with a significance of 0.01 appearing by chance. This is the issue of multiple testing. When so doing, the fact that we try again and again to find a significant event should be taken into account. The most conservative solution is to use a Bonferroni correction, by which we divide the desired significance (say 0.01) by the number of possibilities (1,200,000 in the example above), and look for an event with a probability below that number (1 in 120,000,000 for the example above). If we find one such event, then the activity is not likely to be random. But what if in the above example we find 200 different PFSs with chance probability below 1 in 10,000? We would expect to see 120 such events by chance; the probability of seeing 200 or more by chance when 120 are expected is miniscule. (1-*poisscdf*$(199, 120) = 0.000000000016$ for Poisson counting).

This issue is examined in two parts. First, the issue of assigning significance to one particular pattern is discussed, followed in part two by the issue of multiple trials.

9.2.2 Significance of a Particular PFS

Suppose that you expect to see a particular triplet $PFS = \langle u_1, u_2, u_3; t_2, t_3\rangle$ and want to know whether it indeed excessively repeated in the data. How does such a situation arise? You may expect true synchrony ($t_2 = t_3 = 0$) and look for such events. The chapter on "unitary events" 10 deals with this issue. The pattern may be an expectation put forward by a model. Or, you can divide the data into a training and a test dataset. The PFS that was most prominent in the training set can be selected, and then the significance of its occurrence in the test set can be tested.

Let us assume that you looked at the data and found that a particular (predefined) pattern appeared N times. Is it significant? One way to answer this question is to generate many (say 1,000) surrogatespike trains with similar statistics to the actual spike trains (see Chap. 17 on Surrogate spike train generation) and for each surrogate, calculate how many times this particular pattern appeared. These measurements are then used to build a probability density function for obtaining $0, 1, \ldots$ repetitions, and then the probability of seeing N or more repetitions is calculated. If the result is below the defined threshold α, it is considered significant.

This approach can run up against two types of problems, inappropriate surrogate and erroneous classification of nonprecise time patterns as precise.

Inappropriate surrogate: If the surrogate is made up of high-order gamma distributions, there may be a great number of chance patterns because high-order gamma distributions generate quasiperiodic spike trains. In a periodic spike train any set of spikes will appear repeatedly. Or, if the spike trains are actually quasiperiodic but the surrogate is a Poissonian spike train, the surrogate will reveal much fewer patterns than unrelated but periodic spike trains. See Baker and Lemon (2000) for such cases.

Nonprecise time patterns: Suppose that there is a wide hill in the 3-fold correlation. Its peak may well be significant when compared to 3-fold correlations of uncorrelated surrogates. However, we are interested in precise patterns, and this peak is not precise in the sense that there are multiple similar patterns that also repeat many times.

A partial solution to both problems is as follows: we determine whether the suspected PFS occurred excessively by considering its neighboring patterns. In other words, we count how many PFSs there are within bins representing little shorter or longer delays than the target bin, and use the mean number of repetitions per bin as the expected count (call it x). We then need to compute

$$P = \{N \text{ or more} \mid x\}.$$

This probability, obviously, depends on the pdf of the counts. This pdf may be Poissonian even if the spike trains themselves are not (Abeles and Gerstein 1988; Abeles and Gat 2001). In short, the argument is as follows: for every possible delay $\langle t_2, t_3, \ldots \rangle$ and every possible instance of a recording $t \in T$, we test whether the pattern $PFS = \langle u_1, u_2, u_3, \ldots; t_2, t_3, \ldots \rangle$ occurred. (For example, if $T = 100$ s, and the recording resolution is 1 ms, we try 100,000 times.) The probability of finding such a pattern on a given trial is extremely small. Counting how many times we succeeded is expected to be distributed in a Poissonian fashion. Note that this is true even for nonstationary firing processes, because "if the individual processes are well behaved, their superposition is asymptotically a Poisson process" (Cox and Isham 1980).

Usually, a point process is judged to be Poissonian by the finding that its Fano factor (variance-to-mean ratio) for the inter-event-intervals is 1. However, the variance of a point process is often inappropriately measured. The correct way is to compute the power spectrum of the point process and extrapolate the low-frequency component to zero frequency (Brillinger 1975). In our case we take a region in the delays of the PFSs (for instance, a small square of 10 by 10 bins in the 3-fold count matrix). Now we generate a counting point process and insert an event whenever 1 is added to one of the bins in this region. This counting point process should be Poissonian. This has been shown to be the case in many simulated nonstationary point processes and many recorded spike trains when burst-filtering was applied (Fig. 3 in Abeles and Gat 2001 clearly shows the superiority of computing Fano factors by FFT and the value of burst filtering, but without burst filtering), this FFT method may be misleading.

A sufficient burst-filtering is to define a burst as two or more spikes with intervals between two successive spikes that are smaller than 3 times the bin size (6δ in the

symbols used above). For each such burst, only the first spike is considered when searching for PFSs. The rationale is that bursts represent strong deviations from Poissonian point processes, and with a limited amount of data, they also affect the statistics of the counting process. If we had a huge amount of data so that the above counting process could be generated by looking at one bin alone (and not a square region of bins), then as long as the bin was smaller than the spiking refractory period, this burst-filtering would not be required.

It could be argued that bursts signify moments at which very strong excitatory inputs reached the recorded neuron and therefore should be assigned a higher weight than an isolated spike. It could also be claimed that in many pyramidal-to-pyramidal connections the synapse is of the "depressing type", so that for a high-frequency burst, the effect of the second, third, etc. spikes are much reduced (see Thomson and Lamy 2009 for a review). Therefore, by extension only the first spike is relevant. Regardless, it is clear that by considering only the first spike in a burst, the number of repeating PFSs is reduced, so that the estimate of PFSs is conservative.

If the spike times are teetered within a window W and then a search for patterns is made, the counts in the count-matrix are likely to be smeared as though they had been convolved with a window WS that looks like $WS = W * (-W)$, where $*$ symbolizes convolution (Abeles and Gat 2001). Such a convolution demands much less computations than generating multiple versions of teetered spike trains. However, when this idea was tested on the pairwise correlations of a very large number of simulated Poissonian spike trains, it became obvious that the estimates were over-conservative. Instead, we gain maximal power by convolving with WS' given by:

$$WS' = \begin{cases} WS(t), & t \neq 0, \\ f \cdot WS(0), & t = 0, \end{cases}$$

where $f = 0.4$ for a rectangular WS, and 0.6 for a Gaussian WS (Stark and Abeles 2009). WS' gives less weight to the counts in the very bin for which we are estimating the expectation by the convolution.

A practical point is that instead of convolution in time, it is faster to multiply the FFT of the data with the FFT of WS' and then transform back to the time domain. To avoid edge effects, one may expand the counting vector or matrix by repeating it symmetrically at the edges and then trimming after transforming back to the time domain.

9.2.3 Multiple Comparisons

When there is no prior knowledge on the detailed structure of the PFS, it is tempting to scan through all the possible PFSs and pick the most prominent ones. As this means looking for significance out of a huge number of possibilities, it is not clear how to judge what is significant. Two approaches to this situation have been

suggested: examining the PDF of the probabilities over all the trials and devising a statistic that is expected to reach extreme values if there is an excess of PFSs.

9.2.3.1 PDF of Probabilities

For every possible PFS, we estimate the probability of its being a chance occurrence, as we would do if it were the one we were a priori looking for (by one of the methods in the previous sections). In the example above, we would obtain 1,200,000 probabilities. These probabilities can be treated as random variables. If every occurrence was due to chance, these probabilities should be equally distributed between 0 and 1. If they are not, then the samples we had are not governed by chance alone (Fisher 1932). However in our case, there is a further complication. The counts upon which we base our calculations are discrete random variables (having values of 0, 1, 2, ...). To remedy this, the probability of $\{R$ or more given $x\}$ is replaced by a random value between $\{R$ or more given $x\}$ and $\{R+1$ or more given $x\}$ (Pearson 1950). (The author is grateful to D.R. Brillinger for these references.) Figure 9.4 illustrates the PDF of such probabilities.

Real data distributions such as in Fig. 9.4 were either flat or showed a small excess of very low probabilities. Such excess indicates the presence of an excess of PFSs above chance.

Fig. 9.4 *PDF of probabilities in multiple comparisons*. The *black strip* shows the top of a histogram (with $\sim 5{,}000$ bins) of the probabilities in each element of $\sim 2{,}000{,}000$ different counts of possible PFSs. The *gray line* at the middle is the expected count if counts were random (and the way of computing their probabilities was accurate). It is often hard to see in such histograms what happens at very low probabilities. Therefore, in **d**, **e**, and **f** some summary statistics are given in the *four lines* at the *bottom* of the figure. These describe, from *top* to *bottom*: What are the probability of seeing the total number of elements with probabilities below 0.001 given the expected; same for elements with probabilities below 0.01; same for the smallest bin in the histogram; and the probability of finding correlations with significant PFSs by chance. These probabilities are computed by assuming that the number of cases in the histogram are distributed in Poissonian manner around the expected. In the random spike trains (**d**) and in some real data (**e**), these statistics are insignificant, but in some real data (**f**), the first three statistics were highly significant. **a**. Probability of obtaining n or more given x. 2,001,000 x's were randomly distributed between 5 and 15, and 2,001,000 n's were obtained by sampling Poisson distribution with expectation x ($poissrndf(n, x)$ in Matlab). The probabilities of getting n or more by chance (1-$poisscdf(n-1, x)$ in Matlab) are not equally distributed between 0 and 1. **b**. When probabilities are taken as a random value between $\{n$ or more given $x\}$ and $\{n+1$ or more given $x\}$, the distribution is flat. **c**. 201 count matrices were computed for 201 simulations of three correlated spike trains. Each matrix had 100×100 bins. The expected ($\times$) was estimated by convolving the matrices with a 2D Gaussian kernel having standard deviations of five bins. There are too few very small probabilities. **d**. Same as (**c**), but the center of the smoothing kernel was set to 0. Now the distribution of probabilities is exactly flat. **e**. Data from the premotor cortex of a scribbling monkey. Data around a certain arc shape were selected. The distribution is flat, like for random numbers. **f**. Data from the same recording date, but selected around a different arc. There is a very slight, but highly significant ($p < 0.002$), excess of probabilities below 1‰

9.2.3.2 A Summarizing Statistic

This approach was suggested by Geman and Bienenstock (Hatsopoulos et al. 2003). Their basic premise is as follows: We make the null hypothesis (H0) that in the spike train there is nothing which is precise up to $\pm w$. If so, then teetering all spikes within $\pm w$ should not affect any statistic that we derive from the data. Therefore, choose a statistic that you believe to represent precise firing, compute it for the data,

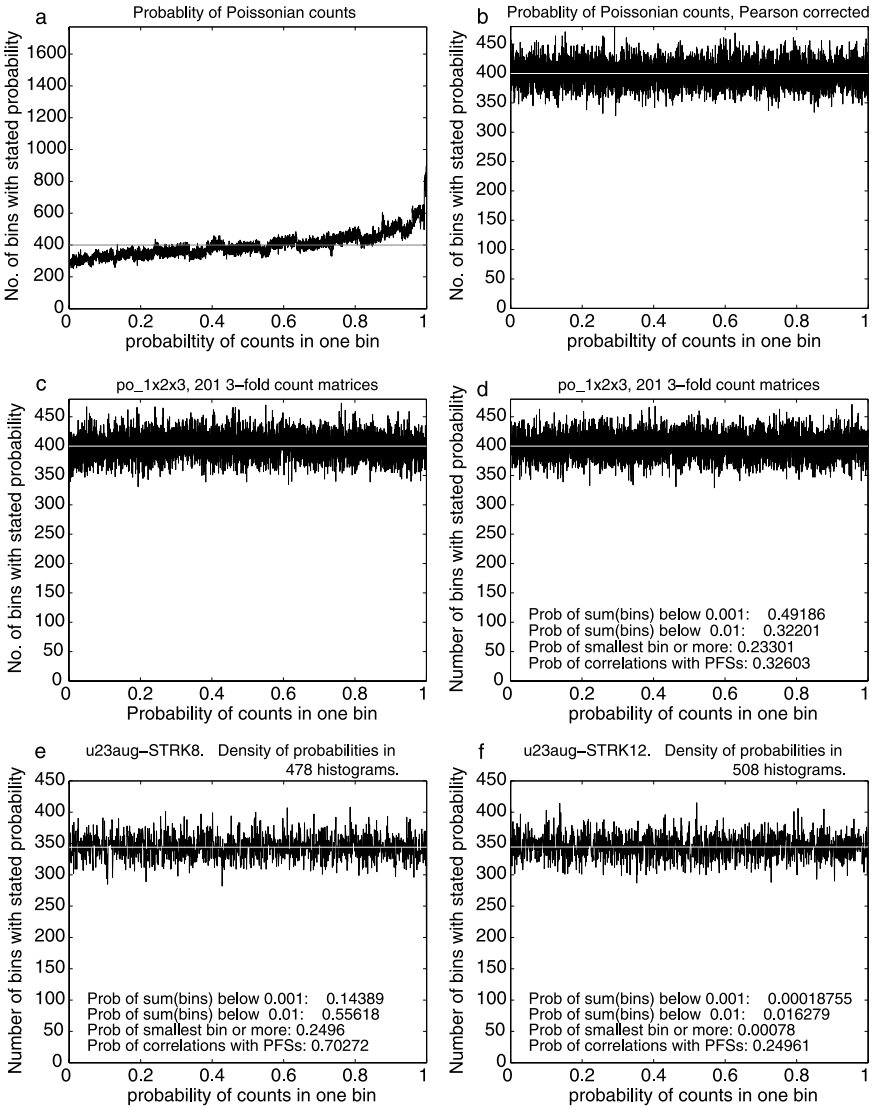

Fig. 9.4 Continued

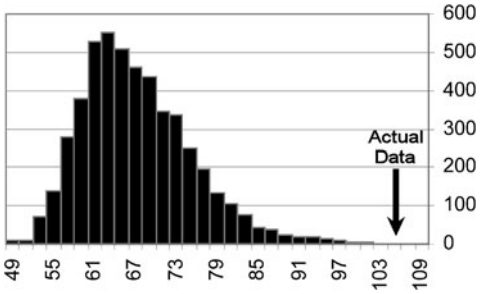

Fig. 9.5 *Teetered and real statistics (adapted from* Shmiel et al. 2006*)*. Data were eight spike trains recorded in the premotor cortex of a monkey that was freely scribbling. Data around a particular arc were collected and all pairwise correlations computed. Only pairs recorded through different electrodes were used. S was computed for the data. Then, the spikes were teetered within ± 5 ms, and S was recomputed. This was repeated 5,000 times. Clearly, the probability of getting S of the actual data is well below 0.0002

and then teeter the spike train many times and after each teeter recompute the same statistic. Find in how many cases the statistic derived from the teetered data exceeds the statistic from the actual data. This yields you an estimate of the probability of finding so many PFSs by chance.

A simple illustration of that is given in Fig. 9.2(bottom). There the statistic was the value at the highest bin of the correlogram.

Figure 9.5 illustrates such a procedure for a more complex statistic. There, the chosen statistic (S) was $S = -\sum_{i \in \{10 \text{ lowest } P\}} \log_2(P_i\{\text{leastlikely bin}\})$.

This approach has the advantage that by using a teetering window (W) of different width one may also estimate the timing accuracy of the data, as is illustrated in Fig. 9.6.

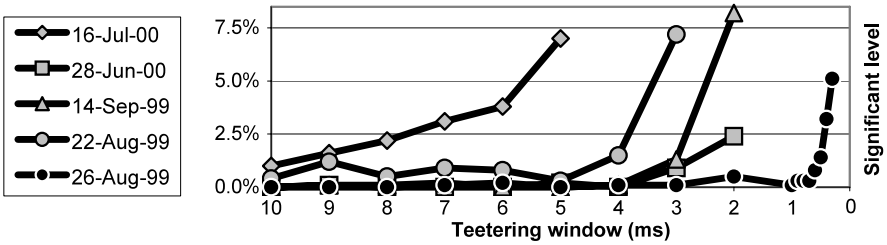

Fig. 9.6 *Time accuracy for five experimental sessions (adapted from* Shmiel et al. 2006*)*. If we select 2.5% as the desired significance level, the accuracy on these five experimental days was between 0.5 and 8 ms. The significance level is for the S-values as defined in the text. Data from two monkeys moving at will in a curved manner on the horizontal plan. This analysis proves that there are precise timing relations between pairs of neurons. It shows that these are found in relation to particular drawing shapes and not other, but are not precisely time locked to any known external event (see Shmiel et al. 2006, for more details). The issue of how such precise timing is generated is beyond the scope of this method chapter. It suffices it to state that the synfire-chain model predicts the existence of such PFSs

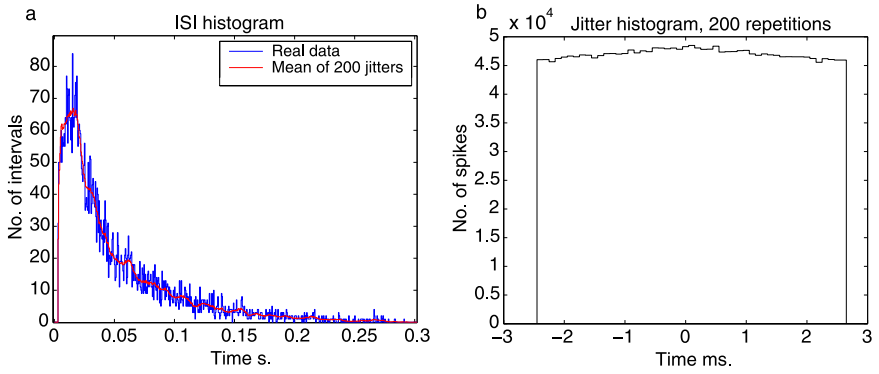

Fig. 9.7 *The statistics of jittering*. Jittering statistics for spikes in the premotor cortex of a monkey while scribbling. **a**. ISI histograms. *Blue* for the real data, *red* – average for 20 jitters. **b**. Histogram of jittering intervals. The spike trains were pruned so that no two spikes came closer than 9 ms. Jittering was done within ± 2.5 ms. However, when two spikes were closer than 14 ms, the jittering did not allow them to come closer than 9. In this way the total number of spikes in the jittered trains was identical to the real data, and the ISI histograms were identical. However, the jittering distribution (**b**) is not exactly square. Note: If the recovery from refractoriness was a step function, the ISI of the teetered data would show an S-shaped recovery. However the "noisiness" of the real ISI (*blue* in **a**) may mask such effects. It may take large amount of data to show significant small changes. This issue requires further studies and quantitative statements about the amount of data needed for detecting a given (small) change in the histogram.

This approach indicates the level of confidence with which we can reject the null hypothesis. However it does not tell us which PFSs are significant. Nevertheless, it is reasonable first to test whether there are any precise PFSs and define the optimal time resolution to use in a search. Only then can we proceed to look for the exact composition of these patterns.

This approach has several shortcomings. The first is that any teetering will convolve firing rates with the teetering window. Thus, any statistics that are affected by firing rates will be affected by teetering. However, if the effect of teetering is small and the amount of data is limited, this may not be detectable. The following example deals with such a case. Suppose that we have uncorrelated spike trains measured for T seconds and that at a certain bin we found 100 counts. After teetering we found only 98 counts. The chances of detecting 100 or more when 98 is expected is 0.433, which would not be considered significant. If we measure the data for $10T$ s and find 1000 counts, whereas the teetered data accounted for 980, the probability of obtaining 1000 when 980 is expected is 0.271, which is still not significant. With this example, one would need to find a situation with 13,000 counts in a bin (with 12,740 after teetering) to reject the null hypothesis with 1% confidence. No experimental situation approaches anywhere near such high counts per bin. However, if there are sharp frequency transitions and a large teetering window, such problems may arise. Figure 9.7 illustrates the effect of teetering on recovery from (artificially imposed) refractoriness.

Note that the limitation on no sharp rate transitions does not invalidate the Geman and Bienenstock approach. Very sharp firing rate transitions do arise when there is some precise mechanism of controlling spike timing, and therefore it is valid to claim that teetering indeed proves the presence of such timing. However, often researchers are not interested in precise spike timing controlled by an external stimulus (such as a click in the auditory system), intracellular mechanisms (such as recovery from refractoriness, or pacemaker firing), or experimental limitations (such as detection dead time imposed by one spike on a second spike from the same electrode). To avoid such pitfalls, the following precautions should be taken:

1. Pass all spike trains through a burst-filter such that no two spikes come closer than D ms. When teetering, if two spikes come closer than $2D$, alter the teetering window to prevent the teetered spikes from being closer than D. This may slightly affect the teetering window as illustrated in Fig. 9.7b.
2. Look for firing patterns using only units that come from different electrodes. If this is too stringent, do not look for patterns with intervals shorter than the maximal dead time of the spike detection algorithm.
3. Avoid including units with close to periodic firing patterns.
4. Be cautious when finding PFSs with 0 lags. This may represent electric coupling between two neurons or a very strong but brief common drive.

All the above restrict the chances of detecting PFSs, but caution should be the watchword until the scientific community is convinced that PFSs exist and that they are clearly related to behavior.

9.2.4 Further Work

This chapter does not relate to the issue of relevance of PFSs to behavior. In our view, the existence and dependence on behavior of PFSs has already been proven (Villa et al. 1999; Shmiel et al. 2006; Maldonado et al. 2008). There is a need for more work on these issues from other labs.

The methods described above may be computationally too time consuming when a very large number of units are recorded simultaneously. A recent method developed by G.L. Gerstein (Schrader et al. 2008) is very useful for such cases.

The methods described above do not allow for time-warping of the patterns. Tetko and Villa (2001) provided a partial solution to this issue, but this work has not been further developed or extensively used.

PFSs are expected to appear if activity is propagated as a wave in a synfire chain. The multiple converging/diverging connections in such a chain suggest that it may occupy a limited volume. Detecting these requires recording from many neurons in a small (say $0.2 \times 0.2 \times 1$ mm) volume in a behaving animal. To date, such recordings are not available.

References

Abeles M (1982) Local cortical circuits: an electrophysiological study. Springer, Berlin
Abeles M (1983) The quantification and graphic display of correlation among three spike trains. IEEE Trans BME 30:235–239
Abeles M, Gerstein J (1988) Detecting spatiotemporal firing patterns among simultaneously recorded single neurons. J Neurophysiol 60:909–924
Abeles M, Gat I (2001) Detecting precise firing sequences in experimental data. J Neurosci Methods 107:141–154
Baker S, Lemon RN (2000) Precise spatiotemporal repeating patterns in monkey primary and supplementary motor areas occur at chance level. J Neurophysiol 84:1770–1780
Brillinger DR (1975) Statistical inference for stationary point processes. In: Puri ML (ed) Stochastic processes and related topics. Proceedings of the summer research institute on statistical inference for stochastic processes. New York, Academic Press, pp 55–99
Cox DR, Isham V (1980) Point processes. Chapman and Hall, London
Dayhoff J, Gerstein GL (1983) Favored patterns in spike trains. II. Application. J Neurophysiol 49:1349–1363
Eckhorn R, Bauer R, Jordan W, Brosch M, Kruse W, Munk M, Reitboeck HJ (1988) Coherent oscillations: a mechanism of feature linking in the visual cortex? Multiple electrode and correlation analyses in the cat. Biol Cybern 60:121–130
Fisher RA (1932) Statistical methods for research workers, 4th edn. Oliver and Boyd, London
Gray CM, Singer W (1989) Stimulus specific neuronal oscillations in orientation columns of cat visual cortex. Proc Natl Acad Sci USA 86:1698–1989
Grün S, Diesmann M, Grammont F, Riehle A, Aertsen A (1999) Detecting unitary events without discretization of time. J Neurosci Methods 94:67–79
Hatsopoulos N, Geman S, Amarasingham A, Bienenstock E (2003) At what time scale does the nervous system operate?. Neurocomp 52–54:25–29
Klemm WR, Sherry CJ (1981) Serial ordering in spike trains: what's it "trying to tell us"? Int J Neurosci 14:15–23
Landolt J, Reinis S, Weiss D (1985) Identification of local neuronal circuits in the visual cortex of the cat. Soc Neurosci Abstr 11:1010
Legendy C, Salckman M (1985) Bursts and recurrences of bursts in the spike trains of spontaneously active striate cortex neurons. J Neurophysiol 53:926–939
Maldonado P, Babul C, Singer W, Rodriguez E, Berger D, Grün S (2008) Synchronization of neuronal responses in primary visual cortex of monkeys viewing natural images. J Neurophysiol 100:1523–1532
Oram MW, Wiener MC, Lestienne R, Richmond BJ (1999) Stochastic nature of precisely timed spike patterns in visual system neuronal responses. J Neurophysiol 82:3021–3033
Pearson ES (1950) On questions raised by the combination of tests based on discontinuous distributions. Biometrika 37:383–398
Schrader S, Grün S, Diesmann M, Gerstein G (2008) Detecting synfire chain activity using massively parallel spike train recording. J Neurophysiol 100:2165–2176
Shmiel T, Drori R, Shmiel O, Ben-Shaul Y, Nadasdy Z, Shemesh M, Teicher M, Abeles M (2006) Temporally precise cortical firing patterns are associated with distinct action segments. J Neurophysiol 96:2645–2652
Stark E, Abeles M (2009) Unbiased estimation of precise temporal correlations between spike trains. J Neurosci Methods 179(1):90–100
Tetko IV, Villa AEP (2001) A patterns grouping algorithm for analysis of spatiotemporal patterns in neuronal spike trains. Detection of repeated patterns. J Neurosci Methods 105:1–14
Thomson AM, Lamy C (2009) Functional maps of neocortical local circuitry. Front Neurosci 1:19–42
Toyama K, Kimura M, Tanaka K (1981) Cross-correlation analysis of interneuronal connectivity in cat visual cortex. J Neurophysiol 46:191–201
Villa AEP, Tetko IV, Hyland B, Najem A (1999) Spatiotemporal activity patterns of rat cortical neurons predict responses in a conditioned task. Proc Natl Acad Sci USA 96:1106–1111

Chapter 10
Unitary Event Analysis

Sonja Grün, Markus Diesmann, and Ad Aertsen

Abstract It has been proposed that cortical neurons organize dynamically into functional groups ("cell assemblies") by the temporal structure of their joint spiking activity. The Unitary Events analysis method detects conspicuous patterns of coincident spike activity among simultaneously recorded single neurons. The statistical significance of a pattern is evaluated by comparing the number of occurrences to the number expected on the basis of the firing rates of the neurons. Key elements of the method are the proper formulation of the null hypothesis and the derivation of the corresponding count distribution of coincidences used in the significance test. Performing the analysis in a sliding window manner results in a time-resolved measure of significant spike synchrony. In this chapter we review the basic components of UE analysis and explore its dependencies on parameters like the allowed temporal imprecision and features of the data like firing rate and coincidence rate. Violations of the assumptions of stationarity of the firing rate within the analysis window and Poisson statistics can be tolerated to a reasonable degree without inducing false positives. We conclude that the UE method is robust already in its basic form. Still, it is preferable to use coincidence distributions for the significance test that are well adapted to particular features of the data. The chapter presents practical advice and solutions based on surrogates.

10.1 Introduction

The principles of neuronal information processing are still not well understood and continue to be debated (Shadlen and Movshon 1999). In the classical view,

S. Grün (✉)
Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan
e-mail: gruen@brain.riken.jp
url: http://www.cnpsn.brain.riken.jp

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_10, © Springer Science+Business Media, LLC 2010

firing rates play a central role in neural coding (Barlow 1972). This idea indeed led to fundamental insights into the neuronal mechanisms of brain function. In parallel, however, a different concept was developed, in which the temporal organization of spike discharges within functional groups of neurons, so-called neuronal assemblies (Hebb 1949; Gerstein et al. 1989), contributes to neural coding (Von der Malsburg 1981; Abeles 1991; Singer 1999; Harris 2005). It was argued that the biophysics of synaptic integration favors coincident presynaptic events over asynchronous ones (Abeles 1982; Softky and Koch 1993; Goedeke and Diesmann 2008). Accordingly, synchronized spikes are considered a property of neuronal signals that can be detected and propagated by other neurons (Diesmann et al. 1999). In addition, these spike correlations should be dynamic, reflecting varying affiliations of the neurons, depending on stimulus and behavioral context. Thereby, synchrony of firing would be directly available to the brain as a potential neural code.

Experimental studies provide support for both perspectives, and both coding schemes may well coexist. However, the discussion about the relevant coding scheme is often implicitly a discussion about the analysis methods and their ability to decide between the two. Therefore, there is a need for analysis tools that allow us to reliably detect correlated spiking activity that is not explained by the firing rates of the neurons alone. The Unitary Events (UE) analysis is such a tool. It was designed to detect coordinated spiking activity that occurs significantly more often than predicted by the firing rates of the neurons. The method allows one to analyze correlations not only between pairs of neurons but also between multiple neurons, by considering the various spike patterns across the neurons. In addition, the method allows one to extract the dynamics of correlation between the neurons by performing the analysis in a time-resolved manner. This enables us to relate the occurrence of spike synchrony to behavior.

This chapter reviews first the basic components of the UE method: a significance test based on the null hypothesis of independent firing and the machinery for a time-resolved analysis. Under certain conditions, the analysis can be based on analytical expressions. We use these expressions to discuss the characteristics of the method and demonstrate that the approach copes well with typical features of experimental data (nonstationarities in time and across trials, deviation from Poisson). Subsequently, we discuss solutions based on surrogates that incorporate more complex features of experimental data into the null hypothesis. After relating the UE approach to other correlation analysis methods, we conclude with a practical guideline for data analysis.

10.2 Basic Elements of the UE approach

The UE analysis method (Grün et al. 2002a) is designed to detect coincident spike patterns between two or more simultaneously recorded spike trains and to assess the significance of the observation. The specific questions addressed by this analysis are: (1) do the simultaneously recorded neurons show correlations of their spiking

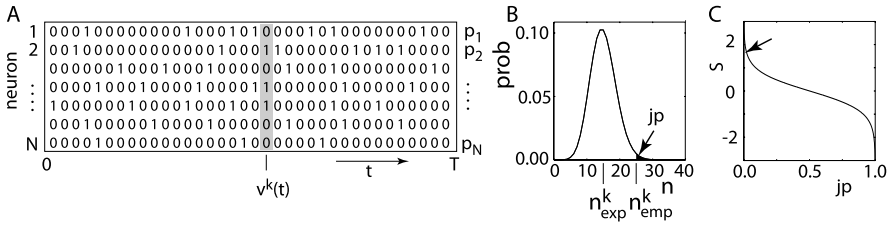

Fig. 10.1 **A**. Binary data representation of N parallel spike trains within a data stretch of T bins of width h. **B**. Distribution of coincidences given the expected number n_{exp}. Significance (jp-value, *black area*) of the number of empirically found coincidences n_{emp}. **C**. Transformation of the jp-value into the surprise measure S. The *arrow* indicates the surprise value corresponding to the identified jp-value in **B**. (Modified from Grün 2009.)

activity, (2) is any such correlation specific to subgroups of the neurons, and (3) do these correlations change dynamically in dependence of stimuli or behavior?

10.2.1 Detection of Joint-Spike Events

The spiking activity of all simultaneously recorded neurons is represented, after appropriate time discretization (e.g., $h = 1$ ms), as parallel sequences of zeros and ones; "1" indicating the existence of at least one spike (clipping), and "0" the absence of spikes (Fig. 10.1A). Under the assumption of stationary firing, the marginal firing probability p_i of neuron i is estimated by evaluating its spike frequency, i.e., the number of ones c_i within the total number of bins T/h in the observed time interval of duration T, and thus, $p_i = c_i/(T/h)$. A similar statistic can be obtained for a particular coincidence pattern composed of zeros and ones. There are at most 2^N different coincidence patterns in data of N simultaneously observed neurons. Due to the finite recording time and the dominance of zeros, however, the actual number of different coincidence patterns found is typically much lower. A unique index k is assigned to each existing pattern based on some arbitrary sorting. For each pattern k, we determine the number of occurrences termed the empirical count n^k_{emp}.

10.2.2 Null Hypothesis

We are interested in detecting whether a coincidence pattern occurs significantly more often than expected on the basis of the firing rates of the neurons involved. To this end, we compare the empirical number of occurrences of pattern k to the expected number by calculating the joint probability of occurrence of the particular 0–1 configuration assuming statistical independence:

$$P^k_{\text{exp}} = \prod_{i=1}^{N} \varphi\left(\overrightarrow{v}(i)\right) \quad \text{with } \varphi\left(\overrightarrow{v}(i)\right) = \begin{cases} p_i & \text{if } \overrightarrow{v}(i) = 1, \\ 1 - p_i & \text{if } \overrightarrow{v}(i) = 0, \end{cases} \tag{10.1}$$

where p_i is the occupation probability of a bin of neuron i.

The expected number of occurrences of pattern k is then simply given by

$$n_{\text{exp}}^k = P_{\text{exp}}^k \cdot \frac{T}{h}.$$

For better readability, we omit from now on the pattern index k but keep in mind that each expression also holds for any pattern k with P_{exp} defined by (10.1).

If multiple trials are available and cross-trial stationarity cannot be guaranteed, the total number of expected coincidences (10.1) is calculated as the sum of the expected counts of the individual trials

$$n_{\text{exp}} = \sum_{j=1}^{M} \frac{T}{h} P_{\text{exp},j} = \sum_{j=1}^{M} \frac{T}{h} \cdot \prod_{i=1}^{N} \varphi_j\left(\overrightarrow{v}(i)\right)$$
$$\text{with } \varphi_j\left(\overrightarrow{v}(i)\right) = \begin{cases} \frac{c_{i,j}}{T/h} & \text{if } \overrightarrow{v}(i) = 1, \\ 1 - \frac{c_{i,j}}{T/h} & \text{if } \overrightarrow{v}(i) = 0, \end{cases} \tag{10.2}$$

where $c_{i,j}$ is the count for neuron i in trial j. In case we can assume stationarity across trials within T, the firing probabilities of the neurons can be derived as averages across the M trials $p_i = \frac{1}{M} \sum_{j=1}^{M} \frac{c_{i,j}}{T/h}$. The empirical coincidence count is the sum of the coincidences found in the individual trials, $n_{\text{emp}} = \sum_{j=1}^{M} n_{\text{emp},j}$.

10.2.3 Significance of Joint-Spike Events

Next, we evaluate whether the empirical number of coincidences significantly deviates from the expected number. To this end we test if the number of empirical coincidences is consistent with the coincidence distribution resulting from independent processes. The probability P_{exp} to observe pattern k in a particular bin is typically low because already the contributing spikes have a low probability of occurrence. As P_{exp} holds for all bins of the analysis interval, the observed number of patterns is governed by a binomial distribution. For moderate n_{exp}, it is well approximated by a Poisson distribution $\mathscr{P}(n, n_{\text{exp}})$ with n_{exp} as the sole parameter. Note that this distribution ignores any dependencies between the bins. For example, there is a finite probability for n occurrences of a pattern requiring a spike of a particular neuron, even if n exceeds the total number of bins in the interval or the number of spikes the neuron has generated. Similarly, any interval statistics of the spike generation process deviating from Poisson leads to correlation between the bins. More complex analytical distributions can take some of these boundary conditions into account (see Grün et al. 2002a, 2003). Here we use the Poisson distribution because the closed-form expression enables us to illustrate the construction of the significance test and its main characteristics. In Sect. 10.4 we show that the Poisson distribution is also of practical use because it is quite robust against typical violations of the simple assumption that spike trains are generated by a Poisson process with constant rate. In later sections we argue that the large diversity of the experimental data

suggests the generation of count distributions employing the idea of surrogate data rather than the use of explicitly parameterized analytical expressions.

Based on the count distribution $\mathscr{P}$, we define the significance of the empirical number of coincidences n_{emp} as the p-value (here called joint-p-value, jp), i.e., the probability of observing at least n_{emp} coincidences (Fig. 10.1B):

$$\begin{aligned} \text{jp}(n_{\text{emp}}|n_{\text{exp}}) &= \sum_{n=n_{\text{emp}}}^{\infty} \mathscr{P}(n_{\text{emp}}, n_{\text{exp}}) \\ &= \sum_{n=n_{\text{emp}}}^{\infty} \frac{(n_{\text{exp}})^n}{n!} \cdot \exp(-n_{\text{exp}}). \end{aligned} \tag{10.3}$$

If jp is smaller than a predefined significance level α, we infer excess synchrony. If jp is larger than $1-\alpha$, we infer significantly missing coincidences. If excess is detected for the respective coincidence pattern, we call the instantiations of the spike patterns Unitary Events (UE).

Because highly significant events are indicated by very small jp values, we logarithmically transform the jp value into the surprise measure (Palm 1981) for better visualization (Fig. 10.1C):

$$S(\text{jp}) = \log \frac{1-\text{jp}}{\text{jp}}. \tag{10.4}$$

This measure is zero for no deviation from expectation, positive in the presence of more coincidences than expected, and negative if the measurement is lower than the expected count. Large values of S indicate significance (e.g., significance at the 1% level results in a surprise measure of 2.0).

10.2.4 Capturing Dynamics of Correlation

In order to capture time dependent changes of the correlation between neurons, we formulate a time-resolved version of the UE analysis defined in the previous section, using a sliding-window approach. To improve the statistics, neuronal data are typically recorded multiple times under the same stimulus presentation or the same behavioral condition ("trials"). The underlying assumption is that the same neuronal computation is performed across the trials. Therefore, data are cut into trials and aligned on the corresponding stimulus or behavioral event. Then we decide on the width T_{w} (expressed in the number of bins h) of a time window which we slide along the data (Fig. 10.2). At each window position, we separately carry out the UE analysis restricted to this window and simultaneously for all trials (Grün et al. 2002b).

The result of the analysis of a window is represented at the center of the window. The time series of results from subsequent windows provides us with a time-resolved analysis. The offset of successive sliding window positions defines the time resolution of the analysis. For each pattern k, the empirical and

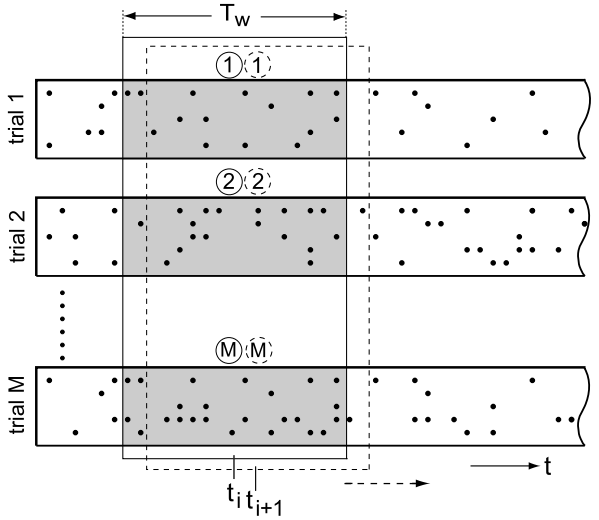

Fig. 10.2 Capturing correlation dynamics by sliding window analysis. After alignment of trials, a window of size T_w is slid along time. At each position of the window, the time segments of all trials covered by the window (*gray*, indicated by *circled trial numbers* from $1, \ldots, M$) are extracted and analyzed for UEs. The procedure is repeated at each new position of the sliding window. The results of each sliding window position are represented at the center of the window, thereby deriving time dependent measures. (Modified from Grün et al. 2002b.)

expected numbers of coincidences are obtained as functions of time ($n_{exp}(t)$, $n_{emp}(t)$, Fig. 10.3B), and the significance is expressed by the surprise measure $S(t)$ (Fig. 10.3C). Thus, the approach enables us to directly relate the dynamics of synchrony (Fig. 10.3D) to the dynamics of the stimulation or the specific behavior of the animal (Riehle et al. 1997, 2000; Grammont and Riehle 2003; Maldonado et al. 2008). In addition, by observing the time-dependent modulation of the occurrence of the different joint-spike patterns we can draw conclusions on the composition of currently active assemblies and the respective participation of the recorded neurons (Grammont and Riehle 1999).

Unitary Event computation

1. Align trials, decide on width of analysis window.
2. Decide on allowed coincidence width.
3. Perform a sliding window analysis. In each window:
   a. Detect and count coincidences.
   b. Calculate expected number of coincidences.
   c. Evaluate significance of detected coincidences.
   d. If significant, the window contains Unitary Events.
4. Explore behavioral relevance of UE epochs.

10.3 Parameter Dependencies

10.3.1 Analysis Window Width

The time scales of the change in firing rate and the modulation of the rate of coincidence events can differ. Even for a temporally stationary firing rate, synchrony may be modulated on a time scale of tens or hundreds of milliseconds (e.g., Vaadia et al. 1995). The UE uses a sliding-window approach to capture such modulation of synchrony (Fig. 10.3). The proper choice of the sliding window width required to identify excess synchrony as significant depends on two factors, the width of the time interval T_c containing excess synchrony ("hot region") and the rate of coincidences λ_c relative to the independent background rate. In the following we analytically study an injection model where coincidences of rate λ_c are injected into independent background activity of rate λ_b in a predefined time interval, yielding a total rate of $\lambda = \lambda_c + \lambda_b$. The background activity is adjusted so that the total rate λ exactly matches the spike rate outside the injection interval, where it consists only of independent activity.

Let us first consider the case of homogeneously inserted coincidences. Trivially, the larger the analysis window T_w, the more coincidences are expected and detected. In the case of two parallel neurons, we expect $n_{exp} = (\lambda \cdot h)^2 \cdot \frac{T_w}{h}$, while observing $n_{emp} = \lambda_c \cdot T_w + (\lambda_b \cdot h)^2 \cdot \frac{T_w}{h}$ coincidences. The larger the coincidence rate, the larger the increase in detected empirical coincidences with increasing T_w (Fig. 10.4A, top, black curves). The expected number of coincidences also increases with T_w (horizontal axis, bottom panel), however, in the same way for all λ_c (Fig. 10.4A, top, dashed curve). The graphs in the bottom panel show the corresponding surprise values as functions of the size of T_w for the different injection rates. Note that for a larger coincidence rate λ_c, the significance level of 1% (dashed line in bottom panels) is reached at a smaller T_w. The values chosen for the coincidence rate were taken in the range detected in experimental data (Grün et al. 1999; Denker et al. in press), i.e., in the range of a few Hz. We find that a minimal size of the analysis window T_{min} is required for detecting excess coincidences as significant. If the analysis window is chosen smaller, the difference of n_{emp} and n_{exp} is too small to be detected as significant. On the other hand, the required difference is not a constant value but scales with the expected number of coincidences: the larger n_{exp}, the more excess coincidences are required, as is indicated by the minimal number of coincidences required for significance n_α (Fig. 10.4A, top, gray). Note that n_α increases in discrete fashion due to the discrete nature of the Poisson distribution, thereby also reflecting changes of the effective significance level (see also Pauluis and Baker 2000).

In case the time interval containing excess coincidences is limited ("hot region"), additional constraints on the detectability of excess synchrony are imposed. Consider a hot region of duration T_c (gray bar in Fig. 10.4B, bottom) and an analysis window of width T_w centered on T_c. For increasing T_w, the empirical count increases linearly up to $T_w = T_c$ due to the increasing amount of injected coincidences. As described for the homogeneous case above, depending on the injected

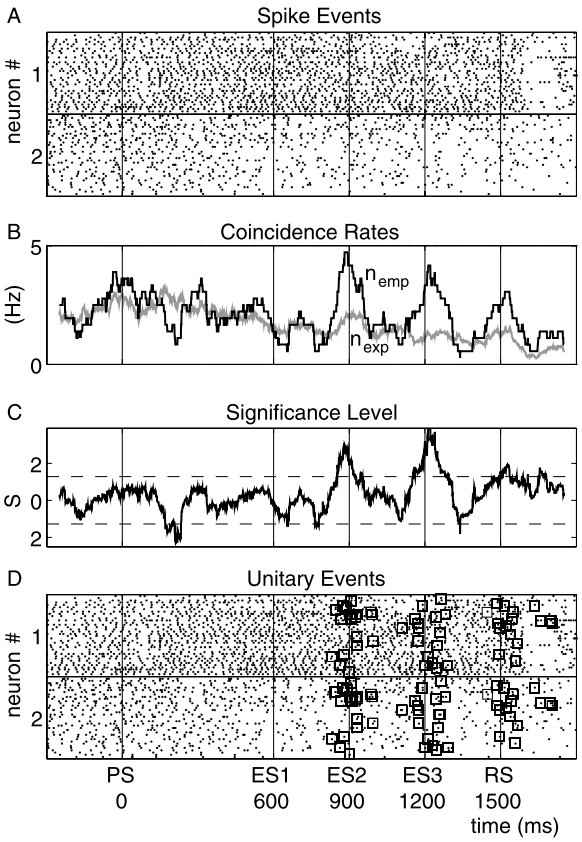

Fig. 10.3 Temporal modulation of synchronous activity. UE analysis of two simultaneously recorded single neurons from motor cortex of awake behaving monkey. The monkey was involved in a delayed pointing task, where the duration of the preparation period (after the preparatory signal (PS) up to the reaction signal (RS)) for the movement was selected randomly from four possible durations (PP; 600, 900, 1200, 1500 ms) from trial to trial. The 36 trials with longest PP duration (1500 ms) were pooled in this example. Thus the monkey could expect the RS to occur at three successive moments (ES1, ES2, ES3) before it actually occurred at RS. **A**. Raster displays of spike discharges of two neurons. **B**. Comparison of measured (*black*) and expected (*gray*) coincidence rates. Allowed coincidence width ± 2 ms. **C**. Surprise as a function of trial time. **D**. Dot display with Unitary Events (*squares*) detected based on a significance level of $\alpha = 0.05$, in sliding windows of $T_w = 100$ ms. (Modified from Riehle et al. 1997.)

coincidence rate, n_{emp} may become significant at a minimal analysis window size T_{min}. A further increase of T_w beyond T_c only leads to an accumulation of chance coincidences from outside the hot region; therefore, the slope of n_{emp} is reduced to the slope of n_{exp}, only maintaining the offset. As a consequence, the distance to the minimal number of coincidences necessary to reach the significance threshold (n_α) shrinks, and the surprise decreases until it falls below the significance level at T_{max}. Thus, the detection of injected coincidences requires $T_{min} < T_w < T_{max}$. The

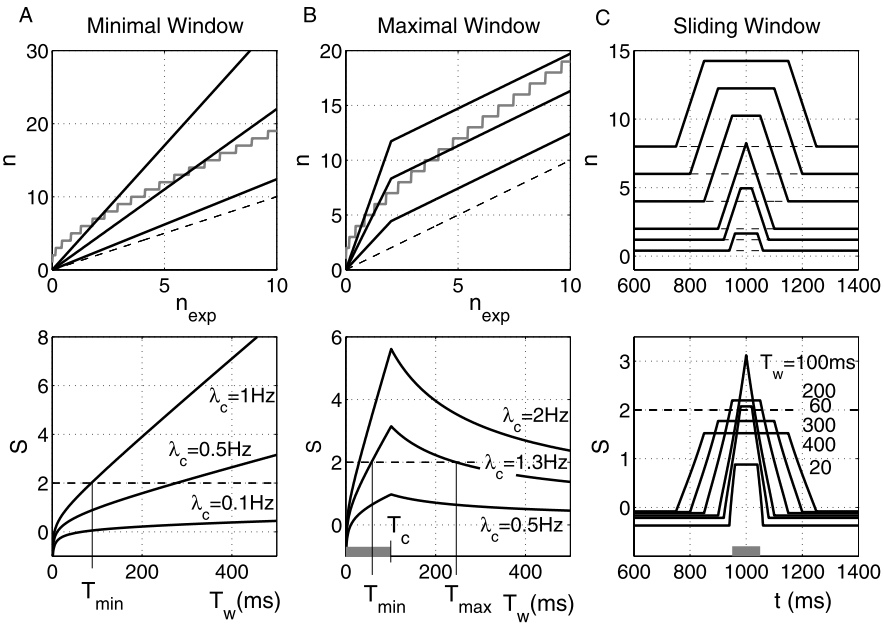

Fig. 10.4 Impact of analysis window size. All *subfigures* show in the *top panels* the coincidence count, and in the *bottom panels* the corresponding surprise value. Two parallel neurons are considered, both of a total firing rate of $\lambda = 20$ Hz, in $M = 50$ trials, significance level is $\alpha = 0.01$ corresponding to $S_\alpha = 2$ (*dashed lines* in *bottom panels*). The background rate during a coincidence injection is $\lambda_b = \lambda - \lambda_c$. **A**.–**B**.: Measures as functions of the analysis window width T_w (*horizontal axis* in *bottom panel*) that can also be interpreted as functions of n_{exp} (*horizontal axis top panels*) for given parameters. n_α as a function of n_{exp} is shown in *gray*, and the diagonal $n(T_w) = n_{exp}(T_w)$ as *dashed*. **A**. Homogeneously injected coincident events. Three curves of n_{emp} as functions of T_w for $\lambda_c = 0.1$, 0.5, 1.0 Hz (*top*) and the corresponding surprise curves (*bottom*). The crossings of n_{emp} with n_α indicate T_{min} (marked for $\lambda_c = 1.0$ Hz) values for the different coincidence rates, or correspondingly the crossings of S with the significance threshold S_α for $\alpha = 0.01$ (*dashed*, *bottom*). **B**. Injected coincidences within a hot region. Curves for injection rates $\lambda_c = 0.5$, 1.3, 2.0 Hz are shown for increasing T_w centered at the hot region ($T_c = 100$ ms, *gray bar*). The crossings of n_{emp} with n_α (or correspondingly S (*bottom*) with S_α (*dashed*)) before the bend in slope indicate T_{min} (marked for $\lambda_c = 1.3$ Hz), the crossings after the bend indicate T_{max} (marked for $\lambda_c = 1.3$ Hz). **C**. Time-resolved sliding window analysis. A hot region of width $T_c = 100$ ms (*gray bar*) with fixed $\lambda_c = 1.3$ Hz is centered at 1000 ms. $n_{emp}(t)$ and $S(t)$ are shown for different analysis window widths (from *bottom* to *top*: $T_w = 20$, 60, 100, 200, 300, 400 ms). The corresponding expected number of coincidences are constant throughout the trial (*dashed lines*)

larger λ_c, the larger T_{max} and the smaller T_{min}, and, consequently, the larger the range of possible T_w in which excess coincidences are detected as significant. For any $\lambda_c > 0$, the surprise peaks at $T_w = T_c$, even for nonsignificant outcomes, i.e., $T_{min} > T_c$.

Figure 10.4C illustrates the situation for an analysis window sliding along the data. The three measures n_{emp}, n_{exp}, and surprise S resulting from the analysis are indicated at the center position of the respective analysis window. The top panel

shows $n_{\mathrm{emp}}(t)$ for different analysis window widths ($T_{\mathrm{w}} = 400$, 300, 200, 100, 60, 20 ms, top to bottom curves). The corresponding $n_{\mathrm{exp}}(t)$ curves (dashed) are stationary but have a different offset depending on T_{w}. The empirical coincidence count starts to increase when the sliding window reaches the hot region (gray bar), assumes a peak value or plateau, and decays again when the window leaves the hot region. The larger T_{w}, the larger the maximum value of $n_{\mathrm{emp}}(t)$. However, as discussed above, significance does not depend on the absolute number of empirical coincidences but on its amount relative to the expected number, which increases for larger T_{w}. Thus, for the example values of T_{w} shown in Fig. 10.4C, only a few curves become significant, i.e., in the cases where $T_{\mathrm{min}} < T_{\mathrm{w}} < T_{\mathrm{max}}$, which for $T_{\mathrm{c}} = 100$ ms is fulfilled at $T_{\mathrm{w}} = 200$, 100, 60 ms. The shape of the surprise curve only depends on the size of T_{w} relative to T_{c}: for $T_{\mathrm{w}} \leq T_{\mathrm{c}}$, the plateau is as large as T_{c}, then becomes narrower with increasing T_{w}, until it exhibits a cusp at $T_{\mathrm{w}} = T_{\mathrm{c}}$. This process is accompanied by an increase in the maximum surprise value. For $T_{\mathrm{w}} > T_{\mathrm{c}}$, the plateau broadens at the cost of a declining amplitude.

In practice we do not know the width of any hot region in the data beforehand. We therefore suggest to vary the analysis window and observe the shape and height of the joint-surprise curve: at a triangular shape, the width of the window corresponds to the duration of the period of excess synchrony, and the analysis is adjusted to maximum sensitivity.

10.3.2 Firing Rate

Next, we study the detection reliability of UEs as a function of the firing rate of the neurons and the injected coincidence rate. We choose λ_{c} in the range of a few Hz as found in experimental data (Grün et al. 1999; Denker et al. in press). The total rate of the neurons is varied between $\lambda = 1, \ldots, 100$ Hz. Figure 10.5A shows the detection rate (true positives) of the coincidence pattern with the largest possible number of spikes ("complexity" $\xi = N$) for systems with different numbers of neurons N. The curves of a particular color correspond to specific total number of neurons ($N = 2, \ldots, 5$) at four different injection rates ($\lambda_{\mathrm{c}} = 0, 1, 2, 3$ Hz). The lower the number of neurons and the lower the injected coincidence rate, the earlier detectability is corrupted by increasing background rate. The reason is that the relative amount of excess synchrony compared to the predicted level is decreasing with increasing firing rate, leading to a nonlinear decrease in the surprise value (compare Fig. 10.4A). For systems of more neurons but the same coincidence rate, this decay is slower because the expected number of occurrences of a synchronous pattern drops with increasing complexity ξ like

$$n_{\mathrm{exp},\xi} = \frac{T_{\mathrm{w}}}{h} \cdot M \cdot P_{\mathrm{exp},\xi} = \frac{T_{\mathrm{w}}}{h} \cdot M \cdot p^{\xi} \cdot (1-p)^{N-\xi}. \tag{10.5}$$

Thus, for an $N = 5$-neuron system, coincidences of $\xi = 5$ injected at a rate of $\lambda = 3$ Hz are perfectly detected even at high firing rates (Fig. 10.5A, red curve with diamonds), whereas pair coincidences are down to a detection rate of about 50% at a background rate of about 50 Hz.

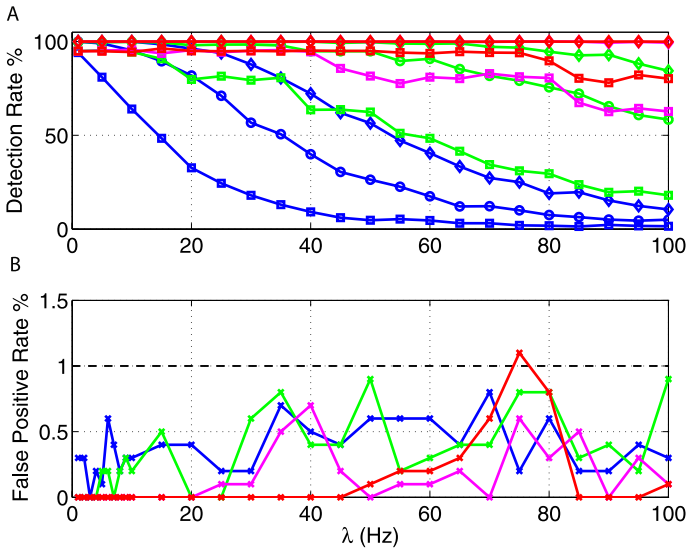

Fig. 10.5 True positive and false positive rates as functions of firing rate for systems with different numbers of neurons. Each data point is the result of 1,000 Bernoulli process realizations of N parallel neurons at stationary rate (*horizontal axis*, varied from 1 to 100 Hz), with injected coincidences (**A**. $\lambda_c = 1, 2, 3$ Hz) and without (**B**. $\lambda_c = 0$ Hz), of total duration $T = 3$ s (corresponding e.g. to $M = 30$ trials and $T_w = 100$ ms). The background rate is $\lambda_b = \lambda - \lambda_c$. The range of firing rates corresponds to spike counts of a single neuron ranging from a mean of 3 to 300. Each realization is evaluated for its empirical n_{emp} and expected n_{exp} coincidence count, and its significance ($\alpha = 0.01$). The number of neurons in the evaluated system is varied from $N = 2, \ldots, 5$ indicated by *blue*, *green*, *magenta*, and *red*, respectively. For each N, the statistics is shown for the spike coincidences of the largest complexity ($\xi = N$)

The false positive rate, i.e., the rate of significant outcomes in independent data (without injected coincidences) is around 1% as expected by a significance level of $\alpha = 1\%$ (Fig. 10.5B). The variation of the false positive levels, which are typically lower than the significance level of 1%, corresponds to the changes in the effective significance levels described by Roy et al. (2000). Due to the discreteness of the coincidence counts, in particular for low expected values, the effective significance level may be much lower than the prescribed α, and the significance tests are more conservative. As a result, for the experimentally realistic range of parameters studied here, we find a low false positive rate independent of the number of neurons considered.

10.3.3 Temporal Precision of Joint-Spike Events

One way to capture potential jitter in coincident spike events is to adjust the bin width accordingly. The choice of the bin width w (in units of time steps h) for detecting coincidences is optimal if the width just catches the temporal jitter of the

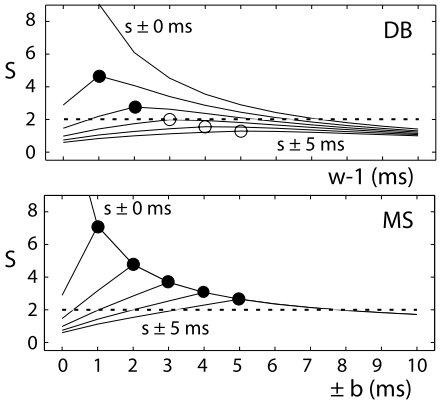

Fig. 10.6 Detection of temporally jittered coincidences. **A**. Comparison of the disjunct binning method (DB, *top*) and the multiple-shift method (MS, *bottom*) using analytical descriptions. The parallel spike trains are assumed as stationary Poisson processes ($\lambda_1 = \lambda_2 = 30$ Hz) with inserted coincident events ($\lambda_c = 1$ Hz) of a given temporal jitter ($s = \pm 0, 1, \ldots, 5$ ms, different curves). The graphs show the surprise for increasing analysis widths (*top*: for bin sizes w ranging from 1 to 11 ms; *bottom*: for maximal shifts from $b = \pm 0, \ldots, \pm 10$ ms; both in steps of $h = 1$ ms). For better visualization, the *horizontal axes* are aligned to correspond to the same maximal spike distances ($b = w - 1$). Each of the *surprise curves* shows a distinct peak, marked by *filled circles* if the values are above the significance level of $\alpha = 0.01$, i.e., $S = 2$, and marked by an *empty circle* if below. The values of S for $b = w - 1 = 0$ (not shown) are 16.76 (DB) and 16.73 (MS). (Modified from Grün et al. 1999.)

coincidences. Of course, also this width is not known in advance, but predictions may be available based on results from other analyses or on the biophysical properties of the neuronal system under study. One way of optimally adjusting the bin width is to systematically vary w. Using simulated data, generated by the injection of coincident events in otherwise independent data, one can show that the significance is largest at the optimal bin width (Fig. 10.6A) (Grün et al. 1999). The peak in the surprise S can be understood as follows: Up to the optimal bin width, more and more of the existing coincidences are detected. At the optimal width, the maximal number is reached. For larger allowed coincidence widths, the number of coincidences only increases because of chance coincidences thereby reducing the relative contribution of excess coincidences and, consequently, significance.

However, binning has a considerable drawback; there is a high probability that coincidences are split by the bin borders such that contributing spikes fall into neighboring bins and, thus, are not detected as coincidences. The division of the time axis into disjunct bins (DB) can lead to a considerable loss of the originally existing coincidences of up to 60% for bin sizes equal to the temporal jitter of the coincidences (Grün et al. 1999). One way to avoid this is to leave the data on a rather fine temporal resolution and shift the spike trains against each other up to the allowed coincidence width b (multiple-shift method (MS), Grün et al. 1999). In this case, the analytical expression for the expected number of coincidences needs to be adjusted to account for the various shifts l. For two parallel spike trains $i = 1, 2$ and M trials, this yields:

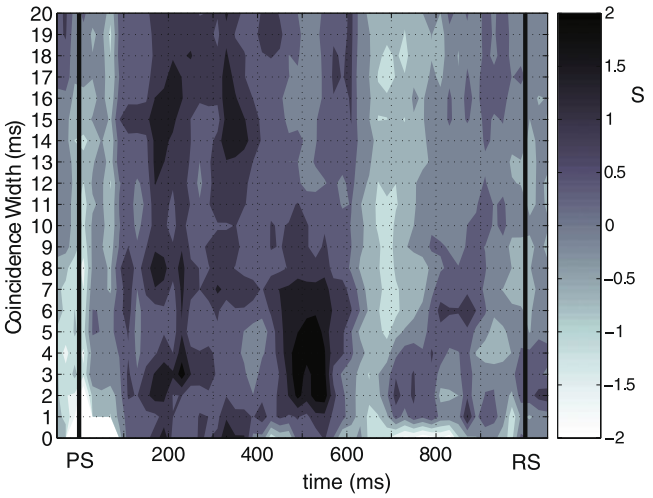

Fig. 10.7 Dynamic changes in coincidence precision. Two simultaneously recorded neurons recorded from monkey motor cortex while the animal was involved in a movement preparation task (for details, see Riehle et al. 2000). PS indicates the time of the preparatory signal, and RS the response signal. The data are analyzed for UEs by the sliding window approach ($T_w = 100$ ms, offset 20 ms). To account for jittered synchronous spikes, the multiple-shift analysis is applied, with variable coincidence widths ranging from $1, \ldots, 20$ ms. The surprise function is calculated separately for each coincidence width and then displayed in matrix form (*horizontal axis*: trial time, *vertical axis*: coincidence width). The surprise values ranges from -2.19 to 2.42, displayed here in *gray scale* (see *color bar*, display clipped below -2 to *white* and above 2 to *black*). (Modified from Riehle et al. 2000.)

$$n_{\text{exp}}^{MS} = \sum_{j=1}^{M} \sum_{l=1}^{2b+1} p_{i=1,j}\, p_{i=2,j,l}\, T_w, \tag{10.6}$$

where $p_{i,j,l}$ is the firing probability of neuron i in trial j at shift l. In this calculation, the number of coincidences resulting from the various shifts is assumed to be independent, which is not necessarily the case in real data. The multiple-shift approach is more sensitive to excess coincidences, and the significance at the optimal coincidence width is considerably larger compared to disjunct binning.

By systematic variation of the maximal shift, the MS method enables us to uncover the typical temporal precision of excess synchronous events in experimental data as the coincidence width with the largest significance. Grün et al. (1999) and Pazienti et al. (2008) found, in two different cortical systems and with different analysis approaches, very similar typical coincidence widths of a few ms. A time-resolved analysis enables us to observe also the dynamics of the temporal precision of coincident events and its relation to the behavioral protocol (Fig. 10.7). In the example shown here, the spike synchrony changes during the trial not only in strength (change in surprise values) but also in temporal precision (Riehle et al. 2000). With a latency of about 200 ms, after the preparatory signal, neuronal activity becomes significantly synchronized at a low temporal precision (jitter of 15 ms). The precision

then increases to a maximum (jitter 2–4 ms) in the middle of the preparatory period at 500 ms. During the remaining time in the preparatory period, neurons become desynchronized and fire independently.

10.4 Impact of Nonstationarities and Other Violations of Assumptions

The basic Unitary Event analysis relies on assumptions, which are typically not fulfilled in experimental neuronal data. The most obvious violation is that firing rates change as functions of time (nonstationarity in time). Another type of nonstationarity is that the base level of firing may change across trials (cross-trial nonstationarity). Latency variability, i.e., variability in the onsets of the rate changes across trials (see, e.g., Richmond 2009; Nawrot et al. 2003), may be seen as a combination of the two. Furthermore, the spike train interval statistics of experimental data (e.g., shape of interval distribution Baker and Lemon 2000, spike-count variability and spike-train irregularity Nawrot et al. 2008a, serial correlations between successive intervals Nawrot et al. 2007) often indicate that the data do not follow Poisson statistics. In the following, we discuss the impact of these features on the UE analysis, in how far they can be tolerated, and how we can cope with biologically realistic properties by incorporating them into the statistical test.

10.4.1 Nonstationary Rates

In the basic UE analysis it is assumed that the firing rates of the neurons are stationary within the analysis time window T_w. If rates change as functions of time, the average rate is not a good description of the rate profile. As a consequence, the expected number of coincidences may not be correct and could give rise to false positive outcomes. The most intuitive way to treat such cases would be to cut the data into stationary pieces and perform the analysis separately in each of those. However, this requires to find time segments that are jointly stationary for all the neurons under consideration. This, in turn, first requires reliable rate estimation and, second, detection of joint-stationary regions (see Appendix D in Grün et al. 2002b). While the idea is appealing, it heavily depends on reliable rate estimation, which is not a trivial task, in particular, if the rate is to be estimated in single trials (e.g., Nawrot et al. 1999; Ventura et al. 2002; Shimazaki and Shinomoto 2010 and Chap. 2 by Shimomoto). In addition, temporal segments corresponding to the joint-stationary regions need to be analyzed independently from each other, and, thus, a smooth transition in time is not given. For these reasons, this route has not been further explored.

An alternative idea is to slide a window of predefined width along the data and carry out the UE analysis separately in each of the windows (Grün et al. 2002b) as introduced in Sect. 10.2.4 to capture the potential dynamics of synchrony. Within

the window, data are assumed to be stationary. We now address the question which consequences we have to expect in terms of false positives (FP) if the rates are not stationary within the analysis window and how much nonstationarity can be tolerated before leading to FPs. For doing this, we consider the worst-case scenario: (1) neurons change their rates in a step-like fashion, and (2) neurons change their rates simultaneously.

For simplicity, we consider here the situation of two neurons changing their rate at the same time from λ_1 to λ_2. The duration of the interval of rate λ_1 is $t_1 = f \cdot T_\mathrm{w}$, and the duration of the second interval of rate λ_2 is $t_2 = T_\mathrm{w} - f \cdot T_\mathrm{w} = (1-f) \cdot T_\mathrm{w}$. We assume that this holds for all trials M. Ignoring the rate change implies to calculate the expected number of coincidences based on the average rate $\bar{\lambda} = \lambda_1 \cdot f + \lambda_2 \cdot (1-f)$ within T_w:

$$\tilde{n} = \bar{\lambda}^2 h^2 \cdot T_\mathrm{w} \cdot M = \big(\lambda_1 \cdot f + \lambda_2 \cdot (1-f)\big)^2 h^2 \cdot T_\mathrm{w} \cdot M. \tag{10.7}$$

However, the correct number of expected coincidences is obtained by calculating the expected number of coincidences separately for each of the stationary rate periods and then taking the sum of the two:

$$n_* = n_{*,1} + n_{*,2} = (\lambda_1 \cdot h)^2 f T_\mathrm{w} M + (\lambda_2 \cdot h)^2 (1-f) T_\mathrm{w} M. \tag{10.8}$$

Note that summation and multiplication are exchanged in the two expressions. Thus, for $\lambda_1 \neq \lambda_2$, n_* is larger than $\tilde{n}$, since the latter underestimates the expected number of coincidences. As a consequence, we tend to overestimate the significance of the empirically found coincidences n_* and obtain false positives. The amount of FPs for a given rate difference can be computed as the part of the area of the coincidence distribution with mean n_* above the minimal coincidence count n_α required for significance derived on the basis of the coincidence distribution with mean $\tilde{n}$:

$$FP = \sum_{n=n_\alpha}^{\infty} \frac{(n_*)^n}{n!} \cdot \exp(-n_*) \tag{10.9}$$

(see Grün et al. 2003 for details and Fig. 5 therein for illustration). Figure 10.8A shows the FP percentage as a function of the rate difference $\Delta\lambda = \lambda_2 - \lambda_1$ for a fixed λ_1. For increasing $\Delta\lambda$, FPs increase and at $\Delta\lambda_\alpha$ surpass the significance level $\alpha = 0.01$. Each curve (color coded) shows this dependence for a different relation f of rate level durations. The curves are nonmonotonic due to changes in the effective significance level (cf. Sect. 10.3.1).

Figure 10.8B shows the minimal rate level difference $\Delta\lambda_\alpha$ leading to FPs as functions of f for different λ_1 (color coded). For small and large f, quite large rate level differences are tolerated before FPs are induced. For intermediate values of f, the tolerated rate level difference decreases, and $\Delta\lambda_\alpha$ exhibits a minimum around $f \approx 0.6$–0.7. The exact minimum is difficult to derive due to the discretized sampling and the nonmonotonicity of FP($\Delta\lambda$). The curves for different λ_1 are similar in shape but have a systematically higher minimum for larger λ_1. For larger number of trials (or, equivalently, longer time window), the tolerated rate difference is lower, e.g., for $M = 100$, $\Delta\lambda_\alpha$ is about half as compared to $M = 30$ (not shown here).

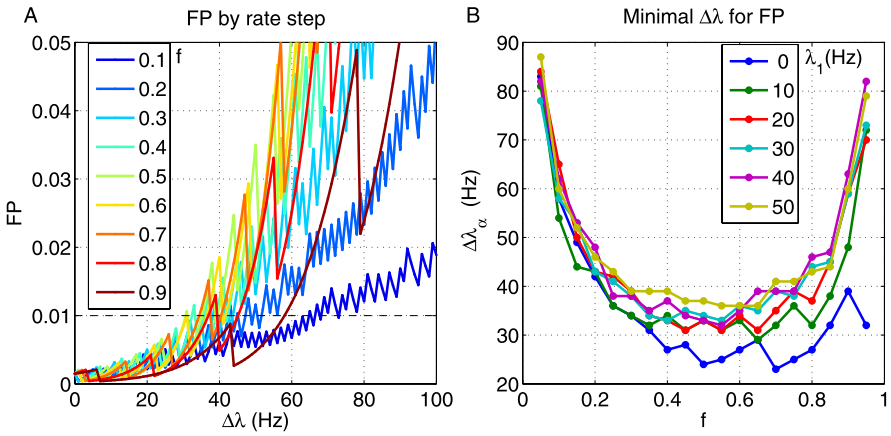

Fig. 10.8 Probability of false positives generated by coherent stepwise rate change. Two neurons are considered which change their rates stepwise in parallel from rate level λ_1 to λ_2. The relative durations of the two rate levels within the time window (here $T_w = 100$ ms) are parameterized by f. The number of trials is $M = 30$. **A**. Percentage of FPs as a function of rate difference $\Delta\lambda = \lambda_2 - \lambda_1$. λ_1 is fixed at 20 Hz, and λ_2 is varied from 20 to 120 Hz in steps of 1 Hz. Each curve is parameterized by f (color coded, see legend). The *dashed black line* shows the significance level of $\alpha = 0.01$. **B**. Critical rate relation $\Delta\lambda_\alpha$ that leads to false positives $\geq \alpha$ as a function of f (here varied in steps of 0.05 in the range $0.05, \ldots, 0.95$). Each curve is retrieved for a different λ_1. The curve for $\lambda_1 = 20$ Hz corresponds to the data shown in **A**

For the analysis of experimental data, these results imply that UE analysis shows a certain robustness to nonstationarity of rate, even if the rates change coherently and in stepwise fashion. The lower the rate level, the smaller the tolerated rate difference. In experimental data, however, rates typically do not perform such instantaneous rate jumps but rather change with finite rise times, reducing the risk of false positives even further.

10.4.2 Cross-Trial Nonstationarity

Next to nonstationarity of firing rate in time, experimental data may also exhibit nonstationarity across trials. For instance, in the simplest case, the offset of the firing rate profile may be different from trial to trial. This may be a consequence of a variation in the depth of the anesthesia or a change in the attention of the animal, etc. In analyses that involve the evaluation of first moments only (e.g., the trial-averaged firing rate), this aspect is often neglected. However, for analysis approaches that involve higher-order statistical moments, as in correlation analysis, ignorance of such cross-trial nonstationarity may lead to false positive results (Brody 1999a, 1999b; Ben-Shaul et al. 2001; Grün et al. 2003; Pauluis and Baker 2000; Ventura et al. 2005).

Here we study false positive outcomes in UE analysis if cross-trial nonstationarity of firing rates is neglected in the calculation of the predictor (Grün et al. 2003).

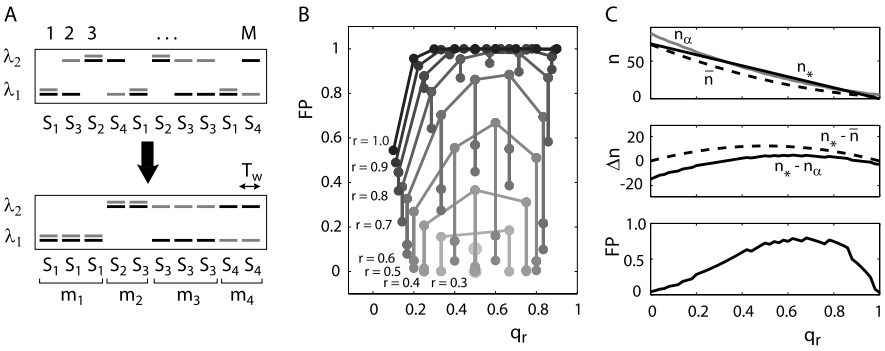

Fig. 10.9 Effect of cross-trial nonstationarity on occurrence of false positives. **A**. Example of an individual M-trial sequence and composition of rate states s_1, s_2, s_3, s_4. *Gray* refers to the rate level of neuron 1, *black* to neuron 2. Below is the reordered sequence of rate states, with $m_1 + m_2$ states forming a coherent rate step, and the remaining $m_3 + m_4$ form opposing rates. **B**. False positives *FP* as a function of rate step duration r and rate step proportion q_r (*horizontal*) for all macrostates of an example system ($\Delta\lambda = 70$ Hz, $q = 0.5$, $M = 10$, $T_w = 1000$ ms, $\alpha = 0.05$). Curves connect data (*dots*) for macrostates with identical r (as labeled, *dark gray* encodes large r, *large dots* for $r = 0.2$). Multiple values of FP at identical (r, q_r) reflect macrostates distinguished by m_3 and m_4 (*vertically connected dots*). **C**. Analysis of a coherent rate step corresponding to $r = 1$, $M = 10$, $T_w = 1000$ ms, $\alpha = 0.05$. The *upper graph* shows n_* (*black solid*), $\bar{n}$ (*dashed*), and n_α (*gray*) as functions of q_r. The *middle graph* shows the difference of the correct expected number n_* to the expected number derived by averages $\bar{n}$ (*dashed*) and to the minimal number at threshold n_α (*solid*). The *lower graph* shows the false positives FP for each pair n_* and $\bar{n}$ as a function of q_r. Steps are due to the discrete nature of n_α. (Modified from Grün et al. 2003.)

For simplicity, we again assume a system of two neurons, however, with stationary firing rates within the analysis window but with rate levels changing from trial to trial. Based on a two-rate-state model, rates are randomly drawn from two possible rate levels λ_1 (low rate) and λ_2 (high rate), independently for each trial and each neuron. Their distance $\Delta_{CT} = \lambda_1 - \lambda_2$ gives a measure for the degree of cross-trial nonstationarity. An additional parameter of the model is the probability to select one of the two rate levels ("occupation probability") termed q for λ_1 and $(1 - q)$ for λ_2. Thus, the degree of nonstationarity across trials and the occupation probability of the rate levels can systematically be varied.

One realization of M trials is composed of a sequence of four possible rate combinations ("rate states"), $[\lambda_1, \lambda_1]$ called s_1, $[\lambda_2, \lambda_2]$ called s_2, $[\lambda_1, \lambda_2]$ called s_3, and $[\lambda_2, \lambda_1]$ called s_4 (Fig. 10.9A, top panel). The occurrence counts of the respective rate states $[m_1, m_2, m_3, m_4]$ depend on the occupation probabilities of the rate levels defined by q (see derivations in Grün et al. 2003). As in the foregoing section, we are interested in the FP probability originating if nonstationarity is ignored. The number of coincidences expected under the assumption that rates are identical across trials is given by the rate averages $\bar{\lambda}_1$ and $\bar{\lambda}_2$ across trials:

$$\bar{n} = \bar{\lambda}_1 h \cdot \bar{\lambda}_2 h \cdot T_w \cdot M = \left(\frac{1}{M}\sum_{i=1}^{M} \lambda_{1,i}\right) h \cdot \left(\frac{1}{M}\sum_{i=1}^{M} \lambda_{2,i}\right) h \cdot T_w \cdot M. \qquad (10.10)$$

The correct expected number of coincidences is the sum of the expected number of coincidences per trial $n_{*,i}$:

$$n_* = \sum_{i=1}^{M} n_{*,i} = \left(\sum_{i=1}^{M} \lambda_{1,i} h \cdot \lambda_{2,i} h \right) \cdot T_{\mathrm{w}}. \tag{10.11}$$

Note, as in the foregoing section, the interchange of sum and product in the two expressions. With these two expressions, we are able to calculate the FP rate as above by calculating the area of the Poisson coincidence distribution with mean n_* starting at $\bar{n}_\alpha$ derived from the coincidence distribution with mean $\bar{n}$ (cf. (10.9)). However, the number of possible rate state realizations across M trials ("microstates") is huge because the four possible rate states s_i may occur in different sequences and numbers of occurrences m_i. Fortunately, the expressions for the number of coincidences are not dependent on the specific sequence of the rate states, but just on their nature. Therefore, we can reduce the complexity by combining all microstates with the same number of rate state occurrences $[m_1, m_2, m_3, m_4]$ into one macrostate. The coincidence counts of a macrostate are then calculated as

$$\bar{n}_m = \left(\frac{1}{M} \sum_{i=1}^{4} m_i \lambda_{1,i} h \right) \cdot \left(\frac{1}{M} \sum_{i=1}^{4} m_i \lambda_{2,i} h \right) \cdot T_{\mathrm{w}} \cdot M \tag{10.12}$$

and

$$n_{*,m} = \left(\sum_{i=1}^{4} m_i \cdot \lambda_{1,i} h \cdot \lambda_{2,i} h \right) \cdot T_{\mathrm{w}}. \tag{10.13}$$

We can think of the consecutive trials $1, \ldots, M$ as a temporal sequence of time segments. By ordering the rate states as a sequence of first low–low rates (m_1 times s_1), followed by a region where both neurons have high rates (m_2 times s_2), and two regions with opposing rates (m_3 times s_3, m_4 times s_4), the temporal sequence can be viewed as a coherent rate step followed by anticorrelated rate levels (Fig. 10.9A, bottom panel). This situation is very similar to the one treated for covarying rate steps in Sect. 10.4.1, with the difference that there the rate steps lasted for the whole duration of time considered. Here, a part of the time sequence of duration $(m_3 + m_4) \cdot T_{\mathrm{w}}$ contains opposing rates (states s_3 and s_4).

We characterize a macrostate more compactly by the relative length of the rate step $r = \frac{m_1+m_2}{M}$ and the relative duration of the low-rate and high-rate regimes $q_r = \frac{m_1}{m_1+m_2}$, a relation comparable to the relation f in the foregoing. In Fig. 10.9B we use the variables r and q_r to structure the set of macrostates of a system with $M = 10$ trials in terms of false positives. Macrostates containing a rate step generate a high fraction of false positives. (The absolute FP rate is higher than in the example shown in Sect. 10.4.1, because here T_{w} is 10 times larger than in the former example.) The longer the rate step (large r), the larger the fraction of false positives. For constant r, the FPs reach a maximum at q_r close to ≤ 0.7. Figure 10.9C demonstrates the relation of the expected number of coincidences n_* and $\bar{n}$ to the minimal number required to be significant (n_α) (top panel). The numbers n_* and $\bar{n}$ decrease

in different manner with the expansion of the low-rate regime, i.e., increasing q_r. The difference of the two (middle panel) exhibits a maximum at $q_r = 0.5$. However, the false positive rate for each pair of $n_*, \bar{n}$ (bottom panel) reaches a maximum at $q_r \approx 0.7$ due to the nonlinearity involved in the calculation of the significance (cf. Fig. 10.4A).

10.4.2.1 Intermediate Summary on Nonstationarities

In conclusion, the most effective generator of false positives turns out to be covariation of firing rates across trials, i.e., trials where both neurons are in the same of the two rate states. The rate of false positives is maximal, if in approximately 70% of the trials, the neurons are jointly in the low-rate state, the remainder in the high-rate state. However, directly including the trial-by-trial spike counts and calculating the expected number of coincidences in a trial-by-trial manner (10.2) avoids the occurrence of FPs.

This consideration of cross-trial nonstationarity can also be interpreted in the context of nonstationarity in time, by reinterpreting the trials as time segments. In this view, the most efficient generator of false positives is a step of coherent firing rates, with 70% of the time spent in the low–low rate configuration, and 30% in the high–high rate configuration. The longer the duration of the coherent rate step, compared to the full duration of the data set, the more false positives are generated. In the extreme, the complete data set represents a coherent rate step as studied in Sect. 10.4.1 (see also Kass and Ventura 2006 for a related aspect). Unfortunately, there is no such simple solution for nonstationarity in time as we had for the cross-trial nonstationarity. Theoretically, the solution is straightforward if the bin-by-bin firing probabilities $p_{i,k,j}$ are known, with i being the neuron index:

$$n_{\mathrm{exp}} = \sum_{j=1}^{M} \sum_{k=1}^{T_{\mathrm{w}}/h} p_{1,k,j} \cdot p_{2,k,j}. \tag{10.14}$$

This would also be a way of calculating the expected number of coincidences for combinations of nonstationarities in time and cross-trial, even in the absence of coherent onsets of rate changes (latency variability). However, the problem is the *estimation* of the firing probabilities, which typically involves taking averages (Nawrot et al. 1999; Ventura et al. 2002; Shimazaki and Shinomoto 2010) either across trials or over time and thus is not as instantaneous as required.

A practical solution is to choose a nonparametric approach, i.e., the generation of the coincidence distribution for the significance test on the basis of surrogate data. The idea is to specifically destroy in the original data the feature one is going to test for, in our case the exact temporal coordination of spikes across neurons, but to conserve all other statistical properties of the original data (see Sect. 10.5.1 below for details). The coincidence distribution results from the different coincidence counts of a sufficiently large number of surrogate data sets.

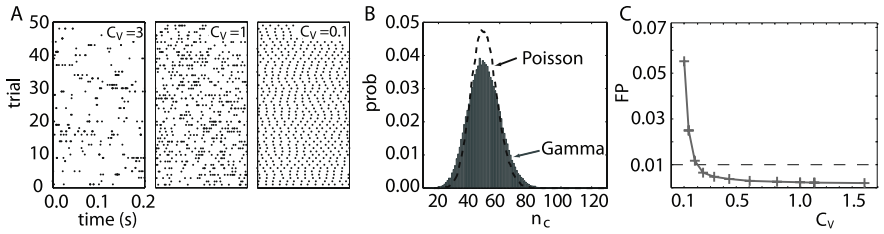

Fig. 10.10 Impact of non-Poisson spike trains on significance estimation. **A**. *Dot* displays of three different examples of realizations of Gamma processes (*left to right*: $C_V = 3$ ("bursty"); $C_V = 1$ (Poisson); $C_V = 0.1$ (regular)). **B**. Comparison of coincidence distributions of two parallel spike trains (*dashed*: Poisson; *gray*: Gamma process with $C_V = 0.1$; firing rates are 50 Hz, $M = 10^5$ trials of 5-s duration, bin width $h = 4$ ms, no clipping). **C**. Coincidences are evaluated based on binning ($h = 4$ ms, clipped). Both processes are parameterized by the product of bin width and firing rate, here 0.1. Probability of false positives (UE analysis, Poisson assumption; significance level $\alpha = 0.01$) for coincidences detected in pairs of independent Gamma processes (rates $\lambda = 50$ Hz, 10^4 realizations of 5 s for each C_V). (Modified from Grün 2009.)

10.4.3 Non-Poisson Processes

The UE analysis method as presented so far assumes the individual spike trains to have Poisson statistics. However, neuronal spike trains often deviate from this assumption. This poses the question to what extent UE analysis is affected by the properties of the individual processes. In particular, whether an error in the assumed process type (here: Poisson) influences the significance estimation of spike coincidences. To investigate this, we model spike trains as renewal processes with interspike intervals (ISIs) drawn from a Gamma distribution (see, e.g., Chap. 16 by Cardanobile and Rotter), which is presently considered to be a reasonable model for experimental spike trains (see references in Nawrot et al. 2008a).

Figure 10.10A shows dot displays of example realizations of a Gamma process for three parameter settings. The processes are parameterized by their coefficient of variation $C_V = \frac{\sigma(ISI)}{\mu(ISI)}$. Processes with higher ISI variability than Poisson ($C_V = 1$) have a C_V larger than 1, and processes with more regular ISIs than Poisson have values smaller than 1 (see Chap. 3 by Nawrot). Figure 10.10B illustrates the distribution of coincidence counts derived from simulations of parallel, stationary, and independent Gamma processes. The shape of the coincidence distribution changes with C_V (or, correspondingly, with the shape of the ISI distribution). For the investigated process type, the distribution is unimodal and becomes wider for $C_V > 1$. For $C_V < 1$, the distribution first gets narrower than Poisson and for even smaller C_V widens again. For unclipped processes (i.e., bins may contain more than 1 spike), the mean of the distribution is the same as for the Poisson process, fully defined by the firing rates of the neurons (Pipa et al. under revision).

Yet, in spite of these obvious deviations from the simple Poisson assumption, UE analysis is in large parts unaffected if the spike trains have Gamma-renewal statistics. Only for very regular processes ($C_V \ll 1$), UE analysis leads to an increased number of false positives. Nevertheless, the significance of coincident events of pro-

cesses with moderate $C_V < 1$ or bursty processes ($C_V > 1$) tends to be underestimated (Fig. 10.10C). The reason why UE analysis does not generate false positives for bursty processes is twofold. First, it operates on binned and clipped spike trains, which considerably reduces the burstiness and leads to a Poisson-like coincidence distribution. Second, UE analysis adjusts the mean of the distribution used for the significance evaluation according to the spike counts in the data under evaluation (Pipa et al. in preparation). Most cortical data show a $C_V > 0.2$ (Nawrot et al. 2008a) leading to the conclusion that when Poisson processes are wrongly assumed, the test is more conservative and does not generate FPs, as was also found in Pipa et al. (2007). A further study extending the scope to nonrenewal processes with first-order negative serial correlation, as found in cortical data (Nawrot et al. 2008a), shows the same general findings (Nawrot et al. 2008b).

Taken together, these results show that the temporal structure of the spike trains influences the significance evaluation of coincident spike events. If Poisson statistics are assumed, the significance may be under- or overestimated, depending on the C_V of the analyzed processes. Obviously, it is preferable to use the proper coincidence distribution for the significance test. Unfortunately, to the best of our knowledge, there is currently no analytical expression for the coincidence distribution, given the C_V, available.

One solution would be to model the simultaneous processes as independent processes and to generate the coincidence distribution from simulated data. However, this requires knowledge about the statistics of the measured data for proper model selection. This is not a trivial task, since ISI distributions of experimental data are often confounded by changes of the firing rates (Baker and Lemon 2000; Johnson 1996). This problem may be solved by time-rescaling approaches, making the firing stationary prior to constructing the ISI distribution and performing parameter estimation (Nawrot et al. 1999, 2008a; Brown et al. 2002). This procedure relies on the assumption that the process parameters do not change in time. If they do, parameters have to be estimated in a time-dependent manner. Another way to create the proper distribution without having to assume a certain point process model is to generate the distribution directly from the measured data with the help of surrogate data (Pipa et al. 2007, 2008; Ito 2007; Louis et al. in press, see Sect. 10.5.1).

10.5 Discussion

Unitary Event analysis is a tool for the analysis of time-dependent spike correlations. Its application has provided important insights into principles of information processing in the cortex in a number of studies. In the visual (Maldonado et al. 2008), the prefrontal (Grün et al. 2002b), and the motor cortex (Riehle et al. 1997, 2000; Grammont and Riehle 1999, 2003; Kilavik et al. 2009), Unitary Events occur at distinct and relevant points in time during the behavior, which could not have been detected by conventional cross-correlation analysis due to averaging over time. Changes in firing rates and spike synchrony often occur in complementary fashion

and may even show complementary tuning. The precision of UEs is typically found in the range of a few ms (Grün et al. 1999) but may also change as a function of time during the trial.

The existence and functional role of spike synchrony are still debated, although both theoretical (Abeles 1982; Bernander et al. 1994; Diesmann et al. 1999; Goedeke and Diesmann 2008; Marsalek et al. 1997; Kumar et al. 2008) and experimental (e.g., Reyes 2003; Rodriguez-Molina et al. 2007) studies have shown that neurons are much more likely to exhibit output spikes if the input activity is synchronized. Reliable tools to decide between rate fluctuations and spike synchrony are required. In this chapter we have outlined the properties and assumptions of UE analysis and have also demonstrated the limitations of the analysis based on analytical expressions for the expected number, the coincidence distribution, and, thus, the significance of coincident events. As we showed, UE analysis can tolerate a reasonable range of nonstationarities and deviations from Poisson before causing false positives. Still, to avoid a wrong interpretation of experimental data, we suggest the following approach: Start the analysis of the data with the UE analysis based on the null hypothesis expressed by (10.2). If there are UEs at time instances were firing rates change abruptly, or the spike trains are extremely regular, perform additional tests by intentionally destroying spike coincidences, or by modeling neuronal activity (Maldonado et al. 2008; Pazienti et al. 2008), or by comparing the analysis results with the results of other analyses (e.g., as done in Pipa et al. 2007), or directly apply nonparametric tests based on surrogates (see below). We stress that these concerns are not specific to UE analysis but apply to any correlation analysis method since any method has specific assumptions that may be violated by the data (Grün 2009).

In the following, we will discuss three further issues of interest in connection to UE analysis. First, we discuss the use of surrogate data as an alternative approach in the analysis of significance. Second, we discuss the use of UE analysis as a time-dependent measure of synchronization in population activity. Finally, we briefly describe the relation between UE analysis and other correlation-based methods of neuronal ensemble activity.

10.5.1 Surrogates

UE analysis rests on a number of assumptions, like stationarity of the processes within the analysis time window and Poisson statistics of the individual processes involved. Under these conditions, the analysis can be performed computationally fast since the distribution for the statistical test (Poisson distribution) is known analytically. Also the parameter for the distribution, the expected number of coincidences, can be derived from the experimental data by simply taking into account the spike counts of the neurons. Nonstationarity across trials can be directly and easily incorporated into that measure by considering the spike counts of each neuron in each trial.

Although UE analysis based on Poisson-distributed coincidence counts exhibits a certain robustness to violations of the assumptions, it is desirable to have a method to construct a coincidence distribution by taking into account more details of the experimental data. The potential advantage is not only an increased reliability (reduction of false positives, FPs) but also an increased sensitivity (reduction of false negatives, FNs). The price is an increased computation time.

One approach is to model the simultaneous processes as independent processes and to realize the coincidence distribution on this basis. This, however, also requires precise knowledge about the statistics of the measured data and reliable methods of model selection. Therefore we suggest to make use of surrogate data to directly generate the proper distribution for the significance test. Surrogate data are artificial data that are generated from the original data by specific manipulations. The idea is to specifically destroy the feature one is going to test for, while conserving all other properties of the original data. In our case the property of interest is the exact temporal coordination of the spikes across the neurons. Thus we intend to destroy the exact timing of spikes. Meanwhile there is a large body of surrogate generation methods available, each with somewhat different characteristics. Some are designed to preserve the ISI distribution of the spike trains, and others preserve the spike counts or rates but at the same time often ignore other features and destroy these. Thus, one has to carefully check the applicability of the surrogate for the question being studied. Chapter 17 by Louis et al. illustrates how to select proper surrogates for particular features present in the data (e.g., strong rate modulations, cross-trial nonstationarity, etc.) in the framework of correlation analysis. A recent review by Grün (2009) gives an overview of currently available surrogate methods, the features they preserve and those they destroy.

The way surrogates are employed for UE analysis is the following: For each position of the analysis window, we construct the coincidence distribution from surrogate data. If the significance level is specified up to two decimals, typically 1000 surrogate data sets are sufficient. Each surrogate is analyzed for the occurrence of coincidences in the same way and with the same parameters (e.g., coincidence width, number of trials) as the original data. For example, if we are interested in the significance of the spike pattern [1 0 1 1 0 0] of neurons $1, \ldots, 6$, we need to count the occurrences of the same pattern in the surrogate data. The coincidence count distribution is constructed from the different counts in the surrogates. Subsequently, the empirical count is compared to the distribution by computing the p-value or surprise. For the next sliding window position, the procedure is repeated. In practice a surrogate is typically created for the whole data set and then evaluated in sliding window fashion in order to avoid border effects.

The contribution by Louis et al. (Chap. 17) illustrates proper selection of surrogates in the context of pairwise correlation analysis of coincident spike events. The study is directly related to UE analysis, since it basically performs UE analysis between a pair of neurons for one position of the analysis window. The only difference is that in their case the analysis window spans the whole trial which includes considerable nonstationarities. Thus their results on the performances of different surrogates result from averaging across the whole trial. The study shows that proper

surrogates can well account for nonstationarities in time and across trials, and for non-Poisson ISI statistics.

Obviously, the surrogate approach is costly in computation time and memory requirements. However, with the help of today's computer clusters and high-level programming languages, this is not a practical problem. Chapter 20 by Denker et al. provides a hands-on introduction into the use of distributed and parallel computing techniques for the analysis of neuronal data.

10.5.2 Population Measures

UE analysis allows one to examine the occurrence of excess spike synchrony within a set of simultaneously recorded neurons and its relation to sensory stimulation and/or to behaviorally relevant events. For example, Riehle et al. (1997) found that neurons in monkey motor cortex synchronize their activity at instances in time when the monkey expects a go signal to initiate an arm movement, even if the signal does not occur. This leads to the interpretation that the cortical network is preparing for the upcoming movement by the activation of neuronal assemblies.

A next step is to confirm that such time- and behavior-dependent phenomena are consistent across the population of neurons. Such questions are often approached by performing many recordings of different sets of neurons from the same network, session by session, under the assumption that the recordings form representative samples. The idea is to capture phenomena reappearing consistently across the different recordings. A simple averaging of the time-dependent surprise functions (or p-value functions) across sessions is not a meaningful approach, since these measures are nonlinear and would need complicated statistical treatment in order to be averaged. An alternative is to indicate time instances that exhibit significant spike synchrony and to calculate the percentage of sessions which show excess spike synchrony at a specific instant in time. For example, if pairs of neurons are analyzed for UEs, the center bins of the sliding analysis window can be marked as "1", if the window contains excess coincidences, otherwise "0", resulting in a binary vector. Doing this for all pairs of neurons under the same behavioral condition yields binary vectors that can be averaged across the population bin-by-bin. The result is a measure of the probability of finding significant synchronization across the population at any instant in time (Riehle et al. 2000; Grammont and Riehle 2003). This time-dependent population measure can then be compared, e.g., to the firing rate averaged across the population of recorded neurons. Grammont and Riehle (2003) found by such an analysis directional tuning of spike synchrony at a specific instant in the preparatory period without any sign of rate tuning. However, the finding was reversed during movement onset which was taken as evidence that coherent activation of cell assemblies may trigger the increase in firing rate in large groups of neurons.

Kilavik et al. (2009) evaluated whether intensive practice induces long-term modifications in the temporal structure of synchrony and firing rate at the population

level. Monkeys were trained in a delayed pointing task in which the selection of movement direction depended on correct time estimation. The goal was to quantify the evolution of synchrony in time and the temporal precision at the level of the population. The observation in individual neuron pairs is that spike synchrony typically occurs in relation to an expected event, but not in a strictly time-locked fashion. Thus the approach to sum the binary vectors is not able to capture this observation. An intuitive approach would be to simply smooth the summed binary vector to allow some temporal spread of the synchrony responses. Unfortunately, the resulting values of the filtered vector cannot be interpreted as the percentage of significant pairs since they are perturbed by multiple testing. An alternative, however, is to slide an integration window along the binary vector of each pair and evaluate if the count of "1"s per window is significant compared to surrogate data. Using this approach, Kilavik et al. (2009) found that the timing of the task is represented in the temporal structure of significant spike synchronization at the population level. During practice, the temporal structure of synchrony is shaped, with synchrony becoming stronger and more localized in time during later experimental sessions, in parallel with an improvement in behavioral performance. Concurrently, the average population firing rate mainly decreases, which is interpreted as performance optimization through practice by boosting the computation via spike synchrony, allowing an overall reduction in population activity.

Another approach used in Kilavik et al. (2009) extracts the strength of the significance of spike correlation of the whole population of pairs. For a particular behavioral condition, the trial-by-trial empirical and expected numbers of coincidences are derived in each sliding window for each neuron pair. Instead of calculating the significance on a pair-by-pair basis and combining the results in terms of UE rate (Maldonado et al. 2008; Ito et al. submitted), one first sums the respective numbers of all trials and all pairs and computes the significance of the total empirical counts given the total sum of expected numbers. By performing this computation at each sliding window position, one obtains a time-dependent surprise function of the population data. All these computations are not restricted to coincidence events between pairs of neurons only but can be performed for any kind of spike pattern across multiple neurons of interest.

10.5.3 Relation to Other Analysis Methods

The standard tool of spike correlation analysis is the cross-correlation histogram (CCH; Perkel et al. 1967, Chaps. 5, 6). It analyzes spike correlation between pairs of spike trains by retrieving the probability for spike occurrence in one spike train relative to the spikes of a reference spike train. It therefore extracts delayed and near-coincidences. Normalization of the CCH by subtracting the expected number of coincidences given the firing rates and dividing by the products of the standard deviations of the spike counts provides a correlation coefficient for each time delay. The approach assumes stationarity of the firing rates and needs to integrate over

relatively long data stretches of time. A typical result of a CCH analysis is a peak around a certain delay, for cortical data typically around zero delay. However, if firing rates are nonstationary, the CCH may also exhibit a peak solely reflecting the nonstationarity of the rates and not the coordination of spike times (see illustration in Chap. 17). The fact that experimental data typically reveal synchronized or near-coincident spike events led to the focus of UE analysis on spike synchrony. Thus, UE analysis is basically evaluating the significance of coincidences with zero or small delay, which corresponds to an evaluation of the central bin entries of the CCH only. In addition, UE is performed in a sliding window manner, which can in principle also be done for the CCH but restricts the maximal delay.

The Joint Peri-Stimulus Histogram (JPSTH) analysis method provides a time-resolved spike correlation analysis (Aertsen et al. 1989). Spike coincidences of any delay of a pair of neurons are entered in a matrix with zero-delay coincident spikes along the diagonal and delayed coincidences at corresponding time delays off the diagonal. The statistics is evaluated across trials separately within each bin. Within each bin, corrections for chance coincidences are performed by computing the correlation coefficient. Along the diagonal, the tool captures the dynamics of coincidences. The applicability of the JPSTH requires a large number of trials (typically, tens to hundreds) to retrieve reliable statistics within the individual bins. In comparison, the UE analysis basically integrates the coincidence counts in a window sliding along the diagonal, thereby requiring fewer trials, but for the price of smoothed results. An approach related to the JPSTH was used for the evaluation of spatio-temporal spike patterns between three neurons (Prut et al. 1998; Abeles and Gat 2001 and Chap. 9). Here the spikes of one neuron are used as the triggers relative to which the temporal relations of the spikes of two further neurons are entered into the matrix. The matrix is actually a section of the snowflake plot (Perkel et al. 1975; Czanner et al. 2005) which was developed as a natural extension of the pairwise to a tripplewise analysis of parallel spike trains. In all these tools, except in the JPSTH, the time dependence of the patterns is lost.

As outlined in Chap. 9, the calculation of the expected number of coincidences for $N > 2$ becomes more difficult, since considerations of higher-order correlations come into play. The null hypothesis may not only be based on full independence of the firing, but should also enable us to test whether tripple events are not trivially explained by the presence of pairwise correlations. However, this makes the formulation of the null hypothesis considerably more complicated (see Chap. 12 by Staude et al.). UE analysis in its present formulation tests for deviations from full independence only. Thus a significant triplet may be the result of a significant pairwise correlation, i.e., pair coincidences coinciding with background spikes. Shimazaki et al. (2009) tackle this problem and develop a state-space-based time-resolved higher-order analysis method using the framework of information geometry (see Chap. 11 by Amari) to identify periods expressing higher-order correlations. In such an approach, the number of parameters to be estimated faces the danger of a combinatorial explosion, thereby reducing the possible number of parallel processes to be analyzed simultaneously or requires huge numbers of trials. Another approach to the analysis of synchronization in the spiking activities of larger ensembles of

neurons is given by the 'gravitational clustering' method (Gerstein et al. 1985; Gerstein and Aertsen 1985). The basic ideas of this method, later improvements, and results of its application to physiological multiple-neuron recordings are described in Chap. 8 by Gerstein.

Finally, two variants of UE analysis, each one with its specific interesting properties, should be mentioned. Gütig et al. (2002) reformulated the statistical test underlying the UE method using a coincidence count distribution based on empirical spike counts, rather than on estimated spike probabilities. This led to the hypergeometric distribution, rather than the binomial distribution, as the test distribution of interest. By analytical calculations of the test power curves of the original and the revised method, they demonstrated that the test power could be increased by a factor of two or more in physiologically realistic regimes. Moreover, in the case of two neurons, they showed that the requirement of stationary firing rates, originally imposed on both neurons, could be relaxed; only the rate of one neuron needs to be stationary, while the other may follow any arbitrary time course. A second variant, called NeuroXidence (Pipa et al. 2008), detects coincident spike events in multiple neurons by a method comparable to the multiple-shift approach and directly uses surrogate data for the generation of the null hypothesis. The method proposes a particular surrogate to use, i.e., to shift the spike trains as a whole against each other by a small amount of time to destroy spike synchrony (see also Harrison and Geman 2009 and Chap. 17). Significance of the empirical number of coincidences is evaluated by comparison with the counts resulting from the shifted versions using a t-test. A comparison of analysis results based on UE and NeuroXidence (Pipa et al. 2007) of experimental data from motor cortex of monkey confirmed the results derived using the standard UE, however with slightly higher significance. This may reflect the fact that the method is accounting for the spike train autostructure.

10.5.4 Conclusion

- The UE method provides a tool to analyze multiple parallel spike trains for time-dependent spike synchrony, enabling the study of the relation between spike synchrony and behavioral context.
- The basic UE method relies on the assumption of trial-by-trial stationary Poisson processes to enable rapid analysis on the basis of an analytical expressions for the significance estimation.
- The basic UE method tolerates reasonable amounts of nonstationarity, as well as non-Poisson ISI statistics for experimentally found C_Vs without generating false positive results.
- Concerns with respect to false positives can be alleviated by cross-checking the results using count distributions constructed from surrogate data.

The software is available at http://www.apst.spiketrain-analysis.org.

Acknowledgements Partial funding by the Helmholtz Alliance on Systems Biology, by the German Federal Ministry of Education and Research (BMBF) to DIP F1.2 and to BCCN Freiburg (01GQ0420), and by the EU-FP6 (15879, FACETS).

References

Abeles M (1982) Role of cortical neuron: integrator or coincidence detector?. Israel J Med Sci 18:83–92

Abeles M (1991) Corticonics: neural circuits of the cerebral cortex. Cambridge University Press, Cambridge

Abeles M, Gat I (2001) Detecting precise firing sequences in experimental data. J Neurosci Methods 107(1–2), 141–154

Aertsen A, Gerstein G, Habib M, Palm G (1989) Dynamics of neuronal firing correlation: modulation of "effective connectivity". J Neurophysiol 61(5), 900–917

Baker S, Lemon R (2000) Precise spatiotemporal repeating patterns in monkey primary and supplementary motor areas occur at chance levels. J Neurophysiol 84(4):1770–1780

Barlow HB (1972) Single units and sensation: a neuron doctrine for perceptual psychology?. Perception 1:371–394

Ben-Shaul Y, Bergman H, Ritov Y, Abeles M (2001) Trial to trial variability in either stimulus or action causes apparent correlation and synchrony in neuronal activity. J Neurosci Methods 111(2):99–110

Bernander Ö, Koch C, Usher M (1994) The effect of synchronized inputs at the single neuron level. Neural Comput 6:622–641

Brody CD (1999a) Correlations without synchrony. Neural Comput 11:1537–1551

Brody CD (1999b) Disambiguating different covariation types. Neural Comput 11:1527–1535

Brown E, Barbieri R, Ventura V, Kass R, Frank L (2002) The time-rescaling theorem and its application to neural spike train data analysis. Neural Comput 14:325–346

Czanner G, Grün S, Iyengar S (2005) Theory of the snowflake plot and its relations to higher-order analysis methods. Neural Comput 17(7):1456–1479

Denker M, Riehle A, Diesmann M, Grün S (in press) Estimating the contribution of assembly activity to cortical dynamics from spike and population measures. J Comput Neurosci. doi:10.1007/s10827-010-0241-8

Diesmann M, Gewaltig MO, Aertsen A (1999) Stable propagation of synchronous spiking in cortical neural networks. Nature 402(6761):529–533

Gerstein G, Aertsen A (1985) Representation of cooperative firing activity among simultaneously recorded neurons. J Neurophysiol 54:1513–1528

Gerstein G, Perkel D, Dayhoff J (1985) Cooperative firing activity in simultaneously recorded populations of neurons: detection and measurement. J Neurosci 5:881–889

Gerstein G, Bedenbaugh P, Aertsen A (1989) Neuronal assemblies. IEEE Trans Biomed Eng 36(1):4–14

Goedeke S, Diesmann M (2008) The mechanism of synchronization in feed-forward neuronal networks. New J Phys 10:015007. doi:10.1088/1367-2630/10/1/015007

Grammont F, Riehle A (1999) Precise spike synchronization in monkey motor cortex involved in preparation for movement. Experimental Brain Res 128:118–122

Grammont F, Riehle A (2003) Spike synchronization and firing rate in a population of motor cortical neurons in relation to movement direction and reaction time. Biol Cybernet 88(5):360–373

Grün S (2009) Data-driven significance estimation of precise spike correlation. J Neurophysiol 101(3):1126–1140 (invited review)

Grün S, Diesmann M, Grammont F, Riehle A, Aertsen A (1999) Detecting unitary events without discretization of time. J Neurosci Methods 94(1):67–79

Grün S, Diesmann M, Aertsen A (2002a) 'Unitary Events' in multiple single-neuron spiking activity. I. Detection and significance. Neural Comput 14(1):43–80

Grün S, Diesmann M, Aertsen A (2002b) 'Unitary Events' in multiple single-neuron spiking activity. II. Non-stationary data. Neural Comput 14(1):81–119

Grün S, Riehle A, Diesmann M (2003) Effect of cross-trial nonstationarity on joint-spike events. Biol Cybernet 88(5):335–351

Gütig R, Aertsen A, Rotter S (2002) Significance of coincident spikes: count-based versus rate-based statistics. Neural Comput 14:121–153

Harris K (2005) Neural signatures of cell assembly organization. Nature Neurosci Rev 5(6):339–407
Harrison M, Geman S (2009) A rate and history-preserving resampling algorithm for neural spike trains. Neural Comput 21(5):1244–1258
Hebb DO (1949) The organization of behavior: a neuropsychological theory. Wiley, New York
Ito H (2007) Bootstrap significance test of synchronous spike events – a case study of oscillatory spike trains. Statistical Med 26:3976–3996
Ito J, Maldonado P, Singer W, Grün S (submitted) Saccade-related LFP modulations support synchrony of visually elicited spikes
Johnson DH (1996) Point process models of single-neuron discharges. J Comput Neurosci 3(4):275–299
Kass R, Ventura V (2006) Spike count correlation increases with length of time interval in the presence of trial-to-trial variation. Neural Comput 18(11):2583–2591
Kilavik B, Roux S, Ponce-Alvarez A, Confais J, Grün S, Riehle A (2009) Long-term modifications in motor cortical dynamics induced by intensive practice. J Neurosci 29(40):12653–12663
Kumar A, Rotter S, Aertsen A (2008) Conditions for propagating synchronous spiking and asynchronous firing rates in a cortical network model. J Neurosci 28:5268–5280
Louis S, Gerstein GL, Grün S, Diesmann M (in press) Surrogate spike train generation through dithering in operational time. Front Comput Neurosci
Maldonado P, Babul C, Singer W, Rodriguez E, Berger D, Grün S (2008) Synchronization of neuronal responses in primary visual cortex of monkeys viewing natural images. J Neurophysiol 100:1523–1532
Marsalek P, Koch C, Maunsel J (1997) On the relationship between synaptic input and spike output jitter in individual neurons. Proc Natl Acad Sci 94:735–740
Nawrot M, Aertsen A, Rotter S (1999) Single-trial estimation of neuronal firing rates: from single-neuron spike trains to population activity. J Neurosci Methods 94:81–92
Nawrot M, Aertsen A, Rotter S (2003) Elimination of response latency variability in neuronal spike trains. Biol Cybernet 88:321–334
Nawrot M, Boucsein C, Rodriguez-Molina V, Aertsen A, Grün S, Rotter S (2007) Serial interval statistics of spontaneous activity in cortical neurons in vivo and in vitro. Neurocomputing 70:1717–1722
Nawrot M, Boucsein C, Rodriguez Molina V, Riehle A, Aertsen A, Rotter S (2008a) Measurement of variability dynamics in cortical spike trains. J Neurosci Methods 169:374–390
Nawrot M, Farkhooi F, Grün S (2008b) Significance of coincident spiking considering inter-spike interval variability and serial interval correlation. In: Frontiers in computational neuroscience. Conference abstract: Bernstein symposium 2008. doi:10.3389/conf.neuro.10.2008.01.017
Palm G (1981) Evidence, information and surprise. Biol Cybernet 42:57–68
Pauluis Q, Baker SN (2000) An accurate measure of the instantaneous discharge probability with application to unitary joint-event analysis. Neural Comput 12(3):647–669
Pazienti A, Maldonado P, Diesmann M, Grün S (2008) Effectiveness of systematic spike dithering depends on the precision of cortical synchronization. Brain Res 1225:39–46
Perkel DH, Gerstein GL, Moore GP (1967) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophys J 7(4):419–440
Perkel DH, Gerstein GL, Smith MS, Tatton WG (1975) Nerve-impulse patterns: a quantitative display technique for three neurons. Brain Res 100:271–296
Pipa G, Grün S, van Vreeswijk C (under revision) Impact of spike-train autostructure on probability distribution of joint-spike events. Neural Comput
Pipa G, Riehle A, Grün S (2007) Validation of task-related excess of spike coincidences based on NeuroXidence. Neurocomputing 70:2064–2068. Published online 2006: doi:10.1016/j.neucom.2006.10.142
Pipa G, van Vreeswijk C, Grün S (in preparation) Impact of spike-train autostructure on Unitary Events
Pipa G, Wheeler D, Singer W, Nikolic D (2008) NeuroXidence: reliable and efficient analysis of an excess or deficiency of joint-spike events. J Comput Neurosci 25(1):64–88

Prut Y, Vaadia E, Bergman H, Haalman I, Hamutal S, Abeles M (1998) Spatiotemporal structure of cortical activity: properties and behavioral relevance. J Neurophysiol 79(6):2857–2874
Reyes A (2003) Synchrony-dependent propagation of firing rate in iteratively constructed networks in vitro. Nature Neurosci 6:593–599
Richmond B (2009) Stochasticity spikes and decoding: sufficiency and utility of order statistics. Biol Cybernet 100(9):447–457
Riehle A, Grün S, Diesmann M, Aertsen A (1997) Spike synchronization and rate modulation differentially involved in motor cortical function. Science 278(5345):1950–1953
Riehle A, Grammont F, Diesmann M, Grün S (2000) Dynamical changes and temporal precision of synchronized spiking activity in monkey motor cortex during movement preparation. J Physiol (Paris) 94:569–582
Rodriguez-Molina V, Aertsen A, Heck D (2007) Spike timing and reliability in cortical pyramidal neurons: Effects of EPSC kinetics input synchronization and background noise on spike timing. PLoS ONE 2(3):e319. doi:10.1371/journal.pone.0000319
Roy A, Steinmetz PN, Niebur E (2000) Rate limitations of unitary event analysis. Neural Comput 12:2063–2082
Shadlen MN, Movshon AJ (1999) Synchrony unbound: a critical evaluation of the temporal binding hypothesis. Neuron 24:67–77
Shimazaki H, Amari S, Brown E, Grün S (2009) State-space analysis on time-varying correlations in parallel spike sequences. In: IEEE international conference on acoustics, speech, and signal processing (ICASSP), pp 3501–3504
Shimazaki H, Shinomoto S (2010) Kernel bandwidth optimization in spike rate estimation. J Comput Neurosci. doi:10.1007/s10827-009-0180-4
Singer W (1999) Neuronal synchrony: a versatile code for the definition of relations?. Neuron 24(1):49–65
Softky WR, Koch C (1993) The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs. J Neurosci 13(1):334–350
Vaadia E, Haalman I, Abeles M, Bergman H, Prut Y, Slovin H, Aertsen A (1995) Dynamics of neuronal interactions in monkey cortex in relation to behavioral events. Nature 373:515–518
Ventura V, Carta R, Kass R, Gettner S, Olson C (2002) Statistical analysis of temporal evolution in single-neuron firing rates. Biostatistics 3(1):1–20
Ventura V, Cai C, Kass R (2005) Trial-to-trial variability and its effect on time-varying dependency between two neurons. J Neurophysiol 94(4):2928–2939
Von der Malsburg C (1981) The correlation theory of brain function. Internal report 81-2, Max-Planck-Institute for Biophysical Chemistry, Göttingen, FRG

Chapter 11
Information Geometry of Multiple Spike Trains

Shun-ichi Amari

Abstract Information geometry studies a family probability distributions by using modern geometry. Since a stochastic model of multiple spike trains is described by a family of probability distributions, information geometry provides not only intuitive understanding, but also useful tools to analyze complex spike trains. A stochastic model of neuronal spikes represents average firing rates and correlations of spikes. We separate correlations of spikes from their firing rates orthogonally. We further separate higher-order correlations from lower-order ones, and thus the effect of correlations is decomposed orthogonally. However, a general model is too complicated and is not adequate for practical use. So we study characteristics of various tractable models. We study among them a mixture model, which is simple and tractable and has many interesting properties. We also study a marginal model and its characteristics.

11.1 Introduction

Information is encoded by trains of spikes in a neuron pool. They have large variability so that such spikes may be regarded as samples from a stochastic model (see Chap. 1 and also Kass et al. 2005; Dayan and Abbott 2005; Salinas and Sejnowski 2001). Information processing takes place through dynamics of mutual interactions of spikes. It is, hence, important to understand characteristics of stochastic models of neural firing and their capabilities and limitations of representing probability distributions. There are a vast number of papers on this topic as is seen in the present book and references cited therein.

Information geometry (Amari and Nagaoka 2000) studies the geometrical structure of a family of probability distributions which forms a manifold. Its geometry

S.-i. Amari (✉)
RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan
e-mail: amari@brain.riken.jp
url: http://www.brain.riken.jp/labs/mns/amari/home-E.html

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_11, © Springer Science+Business Media, LLC 2010

gives various relations among probability distributions. For example, we can discuss divergences or discrepancies between two probability distributions, orthogonality of two small deviations of distributions, and flatness of the manifold. Information geometry provides a mathematical tool to study intrinsic features of stochastic phenomena. The present chapter is devoted to studies of various models of correlated spikes from the point of view of information geometry.

We begin with a general model of multiple spikes in Sect. 11.2. It consists of all joint probability distributions of firing of neurons, where temporal aspects are neglected for the moment. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ be a random vector variable, where $x_i = 1$ represents emission of a spike from neuron i, and $x_i = 0$ otherwise. We discretize the continuous time axis into bins of duration Δt, so that $\text{Prob}\{x_i = 1\}/\Delta t$ represents the firing rate of neuron i. Hereafter, we simply use the firing probability and firing rate interchangeably, although they are different in the exact sense (see Chap. 2). The full model consists of all the probability distributions $p(\boldsymbol{x})$. We first study characteristic features of the full model by defining a Riemannian metric due to Fisher information matrix and two types of special coordinates systems. They are m-affine and e-affine coordinate systems that are mutually orthogonal.

The structure of the full model (Sect. 11.2) is simple and beautiful, but its dimensionality is extremely high when n is large, so that such a model is not tractable for analyzing experimental data. We need to use a tractable model of multiple spike trains, which is a submodel (submanifold) of the full model. Some of the properties of the full model are inherited by these submodels. This shows the importance of the geometrical study of the full model, although the full model itself is seldom used.

By using the full model, we show in Sect. 11.3 how measures of correlations can be separated orthogonally from the firing rates of component neurons (Amari 2009a). We further show how higher-order correlations (see Chap. 12) are separated from lower-order correlations orthogonally (Amari 2001; Nakahara and Amari 2002). This gives an orthogonal decomposition of the Kullback–Leibler divergence used as a measure of difference between two probability distributions into the terms due to firing rates and respective orders of correlations.

In Sect. 11.4, we study various tractable models. One topic treats the problem how higher-order correlations emerge from a simple model receiving common inputs (Amari et al. 2003). Emphasis is put on the mixture model in Sect. 11.5, which is simple but has interesting properties. We use the mixture model to study the dynamic transition of the hidden state of a neuron pool and demonstrate how the firing rates and correlations change in the intermediate state (Amari 2010).

In Sect. 11.6, we then investigate the temporal structure of correlations existing in a spike train emitted from a single neuron. In addition to the full model, we study a Markov process as an example. The second topic is estimation of the gamma parameter of a neuron in a renewal process under the condition that the firing rate is changing over time (Miura et al. 2006). The third topic is the discrimination of ISI distribution from data and how the discrimination power depends on the stochastic model (Kang and Amari 2008).

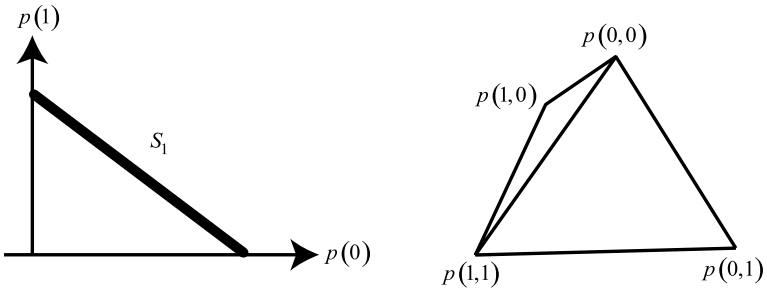

Fig. 11.1 (*Left*) Manifold S_1; (*right*) Manifold S_2

11.2 Joint Probability Distributions of Neural Firing

11.2.1 Joint Firing Coordinates of Full Statistical Model

Let us begin with a simple case where an ensemble of neurons emits spikes stochastically. Here, we focus on the joint probability distributions of firing of n neurons. Temporal aspects of neural firing will be considered in later sections. Let us consider n neurons, and let $x_1, \ldots, x_n$ be random variables taking a value 0 or 1 representing firing or nonfiring of neurons in a time bin. When neuron i fires, $x_i = 1$, and 0 otherwise. Firings of neurons are correlated in general, so we need to study a joint probability distribution $p(x_1, \ldots, x_n)$.

We consider the set of all joint probability distributions,

$$S_n = \{p(x_1, \ldots, x_n)\}, \tag{11.1}$$

which is called the full model. Let $\boldsymbol{x} = (x_1, \ldots, x_n)$ be a vector representing a firing pattern of the n neurons. Since there are 2^n firing patterns, probabilities are assigned to each of them, and they should satisfy

$$\sum_{\boldsymbol{x}} p(x_1, \ldots, x_n) = 1. \tag{11.2}$$

Hence, the degrees of freedom to specify a probability distribution is $2^n - 1$. The set S_n is regarded geometrically as a $(2^n - 1)$-dimensional manifold. When $n = 1$, this is a line (see Fig. 11.1, left), where $p(1) + p(0) = 1$. When $n = 2$, $S_2 = \{p(x_1, x_2)\}$, which includes four probabilities $p(0, 0)$, $p(0, 1)$, $p(1, 0)$, and $p(1, 1)$ with the constraint $p(0, 0) + p(0, 1) + p(1, 0) + p(1, 1) = 1$. Hence, it is a three-dimensional space. It is a 3-simplex (see Fig. 11.1, right).

We have interest in the firing rates and correlations of neural spikes. The firing rate of neuron i is given by its firing probability, which in our case is the expectation of x_i,

$$\eta_i = E[x_i] = \text{Prob}\{x_i = 1\}, \tag{11.3}$$

where E denotes expectation. Similarly, the pairwise joint firing rate of x_i and x_j is given by

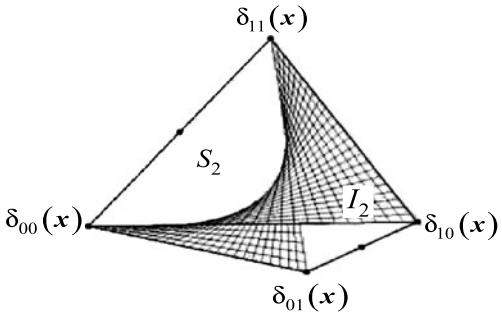

Fig. 11.2 Submanifold I_2 in S_2; $\delta_{ij}(\boldsymbol{x})$ denotes the deterministic distribution of $x_1 = i$ and $x_2 = j$, $i, j = 0, 1$

$$\eta_{ij} = E[x_i x_j] = \text{Prob}\{x_i = x_j = 1\}. \tag{11.4}$$

We may continue this way to give joint firing rates of many neurons. For example, for k neurons $i_i, \ldots, i_k$, their joint firing rate is

$$\eta_{i_1 \cdots i_k} = E[x_{i_1} \cdots x_{i_k}] = \text{Prob}\{x_{i_1} = \cdots = x_{i_k} = 1\}. \tag{11.5}$$

Finally, the joint firing rate of all n neurons is

$$\eta_{12\cdots n} = E[x_1 \cdots x_n] = \text{Prob}\{x_1 = x_2 = \cdots = x_n = 1\}. \tag{11.6}$$

All of the above probabilities η's are $2^n - 1$ in number, and they can be used to specify a probability distribution $p(\boldsymbol{x})$. This plays the role of a coordinate system of S_n, which is called the joint firing coordinate system. We summarize them in a vector having $2^n - 1$ components,

$$\boldsymbol{\eta} = (\eta_i; \eta_{ij}; \ldots; \eta_{1\cdots n}), \quad i, j, \ldots = 1, \ldots, n. \tag{11.7}$$

In information geometry (Amari and Nagaoka 2000), this is called the m-coordinates (mixture coordinates), $\boldsymbol{\eta}$-coordinates, or joint firing coordinates.

When $x_1, \ldots, x_n$ fire independently, we have

$$p(\boldsymbol{x}) = p_1(x_1) \cdots p_n(x_n). \tag{11.8}$$

Let I_n be the set of all independent probability distributions. Because of (11.8), a probability distribution in I_n can be specified by n firing rates $\eta_1, \ldots, \eta_n$. Hence, I_n is an n-dimensional submanifold of S_n (see Fig. 11.2 for the case of $n = 2$). All other $\boldsymbol{\eta}$-coordinates are determined as

$$\eta_{i_1 \cdots i_k} = \eta_{i_1} \eta_{i_2} \cdots \eta_{i_k}. \tag{11.9}$$

11.2.2 Log Interaction Coordinates

We have another coordinate system, emerging from the additive decomposition of the log probability function,

$$\log p(\boldsymbol{x}) = \sum_i \theta_i x_i + \sum_{i<j} \theta_{ij} x_i x_j + \cdots + \theta_{12\cdots n} x_1 \cdots x_n - \psi. \tag{11.10}$$

Since x_i takes values on 0 and 1, this expansion is exact, and this is known as the log-linear model (see Chap. 12). A coefficient, for example, θ_{ij}, represents pairwise interaction of x_i and x_j, and θ_{ijk} triplewise interaction of x_i, x_j, and x_k.

The coefficients of the above expansion can be summarized into a vector of $2^n - 1$ dimensions,

$$\boldsymbol{\theta} = (\theta_i; \theta_{ij}; \ldots; \theta_{1\cdots n}), \quad i, j, \ldots = 1, \ldots, n, \tag{11.11}$$

and they can be used as another coordinate system of S_n. This is called the log-linear coordinate system or log-interaction coordinate system, and the e-coordinate system (exponential coordinate system) in information geometry (Amari and Nagaoka 2000; see also Amari 2001). The parameters $\boldsymbol{\theta}$ are also called the natural or canonical parameters of the exponential family in statistics.

The full model S_n is an exponential family of distributions, where distribution (11.10) is written together with parameters $\boldsymbol{\theta}$ as

$$p(\boldsymbol{x}, \boldsymbol{\theta}) = \exp\Big\{\sum \theta_i x_i + \sum \theta_{ij} x_i x_j + \cdots + \theta_{1\cdots n} x_1 \cdots x_n - \psi(\boldsymbol{\theta})\Big\}, \tag{11.12}$$

where $\psi(\boldsymbol{\theta})$ corresponds to the normalization factor given by

$$\psi(\boldsymbol{\theta}) = \log\Big\{\sum_{\boldsymbol{x}} \exp\Big(\sum \theta_i x_i + \cdots + \theta_{1\cdots n} x_1 \cdots x_n\Big)\Big\}. \tag{11.13}$$

This is the cumulant-generating function and is called the free energy in physics. When $p(\boldsymbol{x}, \boldsymbol{\theta})$ is an independent distribution belonging to I_n, all $\theta_{i_1\cdots i_k}$ $(k \geq 2)$ vanish except for the first-order terms $\theta_1, \ldots, \theta_n$, so that

$$p(\boldsymbol{x}, \boldsymbol{\theta}) = \exp\Big\{\sum \theta_i x_i - \psi\Big\}. \tag{11.14}$$

As is shown in (11.10), $\theta_{i_1\cdots i_k}$ represents the intensity of interaction among k variables $x_{i_1}, \ldots, x_{i_k}$.

11.2.3 Coordinate Transformation Between θ and η

The $\boldsymbol{\eta}$-coordinates are given by the expectations of the random variables $x_{i_1} \cdots x_{i_k}$,

$$\eta_{i_1\cdots i_k} = E[x_{i_1} \cdots x_{i_k}], \quad k = 1, \ldots, n. \tag{11.15}$$

It is known that the transformation of the coordinates from $\boldsymbol{\theta}$ to $\boldsymbol{\eta}$ is given by

$$\boldsymbol{\eta} = \nabla \psi(\boldsymbol{\theta}), \tag{11.16}$$

where ∇ is the gradient, and hence

$$\eta_i = \frac{\partial}{\partial \theta_i} \psi(\boldsymbol{\theta}), \tag{11.17}$$

$$\cdots \tag{11.18}$$

$$\eta_{i_1\cdots i_k} = \frac{\partial}{\theta_{i_1\cdots i_k}}\psi(\boldsymbol{\theta}), \tag{11.19}$$

$$\cdots \tag{11.20}$$

$$\eta_{12\cdots n} = \frac{\partial}{\partial\theta_{1\cdots n}}\psi(\boldsymbol{\theta}). \tag{11.21}$$

The inverse transformation is given by

$$\boldsymbol{\theta} = \nabla\varphi(\boldsymbol{\eta}), \tag{11.22}$$

where $\varphi(\boldsymbol{\eta})$ is the negative entropy,

$$\varphi(\boldsymbol{\eta}) = \sum_{\boldsymbol{x}} p(\boldsymbol{x},\boldsymbol{\eta})\log p(\boldsymbol{x},\boldsymbol{\eta}), \tag{11.23}$$

where, in slight abuse of notation, $p(\boldsymbol{x},\boldsymbol{\eta})$ is the probability distribution whose m-coordinates are $\boldsymbol{\eta}$. The negative entropy is given by the Legendre transformation,

$$\varphi(\boldsymbol{\eta}) = \max_{\boldsymbol{\theta}}\big\{\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \psi(\boldsymbol{\theta})\big\}. \tag{11.24}$$

These show that $\boldsymbol{\theta}$ and $\boldsymbol{\eta}$ are connected by the Legendre transformation, and both $\psi(\boldsymbol{\theta})$ and $\varphi(\boldsymbol{\eta})$ are convex functions (Amari and Nagaoka 2000).

11.2.4 Fisher Information

We consider a general statistical model $M = \{p(\boldsymbol{x},\boldsymbol{\xi})\}$ parameterized by $\boldsymbol{\xi}$, which is not necessarily the full model S_n but can be any model. In the case of the full model S_n, $\boldsymbol{\xi}$ represents $\boldsymbol{\theta}$, $\boldsymbol{\eta}$ or any others. The model M is a statistical manifold, where $\boldsymbol{\xi}$ is its coordinate system. The Fisher information matrix of a statistical model $M = \{p(\boldsymbol{x},\boldsymbol{\xi})\}$ is defined by

$$g_{ij} = E\left[\frac{\partial}{\partial\xi_i}\log p(\boldsymbol{x},\boldsymbol{\xi})\frac{\partial}{\partial\boldsymbol{\xi}_j}\log p(\boldsymbol{x},\boldsymbol{\xi})\right]. \tag{11.25}$$

The Fisher information is given equivalently by

$$g_{ij} = -E\left[\frac{\partial^2}{\partial\xi_i\partial\xi_j}\log p(\boldsymbol{x},\boldsymbol{\xi})\right]. \tag{11.26}$$

This is the Hessian of log-probability. Since

$$E\left[\frac{\partial}{\partial\xi_i}\log p(\boldsymbol{x},\boldsymbol{\xi})\right] = 0 \tag{11.27}$$

holds by simple calculations, the function

$$D\big(\boldsymbol{\xi},\boldsymbol{\xi}'\big) = E\big[\log p\big(\boldsymbol{x},\boldsymbol{\xi}'\big)\big], \tag{11.28}$$

where E denotes the expectation with respect to $p(\boldsymbol{x},\boldsymbol{\xi})$, has the peak at $\boldsymbol{\xi}' = \boldsymbol{\xi}$. Hence, when n independent observations $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n$ are given, for estimating the parameter, we can use the maximum likelihood estimator (mle) $\hat{\boldsymbol{\xi}}$ that maxi-

mizes

$$\hat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}'} \sum_{i=1}^{n} \log p(\boldsymbol{x}_i, \boldsymbol{\xi}'). \tag{11.29}$$

This is because we may replace the expectation of (11.28) by the empirical mean of observed data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. The mle is its maximizer. Since the Fisher information (11.26) is the second derivative of the log-probability at its maximum, it represents how easy it is to find the peak. Hence, it is interpreted as the amount of information that one observation $\boldsymbol{x}_i$ carries for estimating the true $\boldsymbol{\xi}$.

The following Cramér–Rao theorem shows the role and meaning of the Fisher information.

Cramér–Rao theorem *Let $\hat{\boldsymbol{\xi}}$ be any unbiased estimator. The covariance matrix of the estimation error $\boldsymbol{e} = (e_i)$,*

$$\boldsymbol{e} = \hat{\boldsymbol{\xi}} - \boldsymbol{\xi}, \tag{11.30}$$

is bounded by the inverse of the Fisher information

$$E[e_i e_j] \geq \frac{1}{n}(g_{ij})^{-1}. \tag{11.31}$$

Moreover, the equality is asymptotically attained by the mle $\hat{\boldsymbol{\xi}}$ when n is large. The theorem shows that the error is approximately the inverse of the information.

In our case of the full model, the parameters are $\boldsymbol{\theta}$ or $\boldsymbol{\eta}$. It should be noted that the indices in the full model have hierarchical structure so that they run over multiple indices $i, ij, ijk, \ldots$, as $\theta_i, \theta_{ij}, \ldots$. To simplify the notation, we will from now on use capital letters $K, L, \ldots$ to denote indices that run over $i, ij, ijk, \ldots$.

We then have

$$p(\boldsymbol{x}, \boldsymbol{\theta}) = \exp\left\{\sum_K \theta_K X_K - \psi(\boldsymbol{\theta})\right\}, \tag{11.32}$$

where

$$X_K = x_{i_1} \cdots x_{i_k} \tag{11.33}$$

for $K = (i_1 \cdots i_k)$. It is clear that

$$\eta_K = E[X_K]. \tag{11.34}$$

Since S_n is an exponential family, the Fisher information matrix $G = (g_{KL})$ is directly obtained as

$$G = \nabla\nabla\psi(\boldsymbol{\theta}) = \frac{\partial^2}{\partial\theta_K \partial\theta_L}\psi(\boldsymbol{\theta}) \tag{11.35}$$

in terms of the $\boldsymbol{\theta}$-coordinates. This is equal to the covariance of X_K,

$$g_{KL} = \mathrm{Cov}[X_K, X_L]. \tag{11.36}$$

The Fisher information matrix in terms of the $\boldsymbol{\eta}$-coordinates is given by the inverse G^{-1} of G.

11.2.5 Geometry of S_n and Orthogonal Parameters

The length between two nearby points of S_n is defined by using the Fisher information matrix. This is given by considering the Fisher information as the Riemannian metric on S_n. Let us consider two nearby probability distributions $P = p(\boldsymbol{x}, \boldsymbol{\theta})$ and $P + dP = p(\boldsymbol{x}, \boldsymbol{\theta} + d\boldsymbol{\theta})$, each represented by the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\theta} + d\boldsymbol{\theta}$. Then, the square of their distance is given by

$$ds^2 = \langle d\boldsymbol{\theta}, d\boldsymbol{\theta} \rangle = d\boldsymbol{\theta}^T \, G \, d\boldsymbol{\theta}, \tag{11.37}$$

where $\langle d\boldsymbol{\theta}, d\boldsymbol{\theta} \rangle$ is the inner product defined by the right-hand side of (11.37), and $d\boldsymbol{\theta}^T$ is the transposition of column vector $d\boldsymbol{\theta}$. This can be represented also by

$$ds^2 = \langle d\boldsymbol{\eta}, d\boldsymbol{\eta} \rangle = d\boldsymbol{\eta}^T \, G^{-1} \, d\boldsymbol{\eta}, \tag{11.38}$$

when we use the $\boldsymbol{\eta}$-coordinate system. The Fisher information matrix is the unique matrix that is invariant, that is, its structure does not change, under both transformations of parameters and of data (Chentsov 1972; Amari and Nagaoka 2000).

It is shown from (11.38) that $d\boldsymbol{\theta}$ and $d\boldsymbol{\eta}$ are related by

$$d\boldsymbol{\eta} = G \, d\boldsymbol{\theta}. \tag{11.39}$$

Next, we define the orthogonality of two small deviations, dP and $d'P$, that is, the two deviations from P to $P + dP$ and $P + d'P$, respectively. Here, the deviations are represented by two small changes in coordinates $d\boldsymbol{\theta}$ and $d'\boldsymbol{\theta}$. The two small deviations are said to be orthogonal if

$$\left\langle d\boldsymbol{\theta}, d'\boldsymbol{\theta} \right\rangle = d\boldsymbol{\theta}^T \, G \, d'\boldsymbol{\theta} = 0. \tag{11.40}$$

Similarly, two curves $\boldsymbol{\theta}_1(t)$ and $\boldsymbol{\theta}_2(t)$ specified by parameter t, which intersect at $t = 0$, $\boldsymbol{\theta}_1(0) = \boldsymbol{\theta}_2(0)$, are said to be orthogonal at $t = 0$ if their tangent vectors $\dot{\boldsymbol{\theta}}_1$ and $\dot{\boldsymbol{\theta}}_2$ are orthogonal,

$$\langle \dot{\boldsymbol{\theta}}_1, \dot{\boldsymbol{\theta}}_2 \rangle = 0, \tag{11.41}$$

where

$$\dot{\boldsymbol{\theta}} = \frac{d}{dt} \boldsymbol{\theta}(t) \bigg|_{t=0}. \tag{11.42}$$

For a statistical model $M = \{p(\boldsymbol{x}, \boldsymbol{\xi})\}$, let us consider two small deviations $d\boldsymbol{\xi}$ and $d'\boldsymbol{\xi}$ of the parameter $\boldsymbol{\xi}$. Assume that the first deviation is caused by a small change $d\xi_i$ of the coordinate curve ξ_i, all the other coordinates being fixed, and the second one by a small change $d'\xi_j$ of ξ_j. Then, the inner product of the two small changes is

$$\left\langle d\boldsymbol{\xi}, d'\boldsymbol{\xi} \right\rangle = g_{ij}(\boldsymbol{\xi}) \, d\xi_i \, d'\xi_j. \tag{11.43}$$

Two parameters ξ_i and ξ_j are orthogonal if $g_{ij}(\boldsymbol{\xi}) = 0$ for all $\boldsymbol{\xi}$.

It is desirable to use orthogonal parameters when it is possible. Intuitively speaking, when the angle of two coordinates curves ξ_i, ξ_j is very small, it is difficult to

distinguish a change caused by an increment of ξ_i and that of ξ_j. More technically, the score functions

$$s_i(\boldsymbol{x}, \boldsymbol{\xi}) = \frac{\partial}{\partial \xi_i} \log p(\boldsymbol{x}, \boldsymbol{\xi}), \quad i = 1, \ldots, n, \tag{11.44}$$

are correlated,

$$E\big[s_i(\boldsymbol{x}, \boldsymbol{\xi}) s_j(\boldsymbol{x}, \boldsymbol{\xi})\big] = g_{ij} \neq 0, \tag{11.45}$$

when ξ_i and ξ_j are not orthogonal. This implies that the errors e_i and e_j of estimation are correlated. When ξ_i and ξ_j are orthogonal, one can easily test if ξ_i has changed or not, even when ξ_j is not fixed but is fluctuating, since the effects of their changes are orthogonal.

We show in the next section how we separate the firing rates and correlations orthogonally.

11.3 Separation of Correlations from Firing Rates

11.3.1 Orthogonal Measure of Correlation

Given a joint probability distribution $p(\boldsymbol{x})$, one can calculate both firing rates of neurons and their correlations. Since firing rates may change over time, it is important to have a measure of correlation which does not explicitly depend on the firing rates. Geometrically speaking, such a measure of correlation is desirable that is orthogonal to the firing rates.

We begin with a simple case of two neurons. A probability distribution is given by $p(x_1, x_2)$, and the set S_2 of all probability distributions is three-dimensional. Among three degrees of freedom, two are the firing rates of the two neurons,

$$\eta_i = E[x_i], \quad i = 1, 2, \tag{11.46}$$

and the third is correlation of x_1 and x_2. There are a number of measures to represent correlation. The covariance is

$$\mathrm{cov} = \eta_{12} - \eta_1 \eta_2, \tag{11.47}$$

and the correlation coefficient is

$$\rho = \frac{\eta_{12} - \eta_1 \eta_2}{\sqrt{\eta_1 \eta_2 (1 - \eta_1)(1 - \eta_2)}}. \tag{11.48}$$

Mutual information is another one,

$$I(X_1 : X_2) = \sum p(x_1, x_2) \log \frac{p(x_1, x_2)}{p_1(x_1) p_2(x_2)}, \tag{11.49}$$

where $p_i(x_i)$ is the marginal probability of x_i. The coefficient θ_{12} in additive log-decomposition (11.10), which is abbreviated as θ in the present section, is also a

measure of correlation,

$$\theta = \log \frac{\eta_{12}(1-\eta_1-\eta_2+\eta_{12})}{(\eta_1-\eta_{12})(\eta_2-\eta_{12})}. \tag{11.50}$$

All of them vanish when x_1 and x_2 are independent.

Which is adequate as a measure of correlation? It is desirable to use the one that is orthogonal to changes in firing rates. In order to obtain an orthogonal measure, let us denote by c an arbitrary measure of correlation which satisfies $c = 0$ when x_1 and x_2 are independent. Then, probability distributions $p(x_1, x_2)$ in S_2 are parameterized by $\boldsymbol{\xi} = (\eta_1, \eta_2, c)$. A small change in the firing rate is represented by $d\eta_i$, where firing rate changes from η_i to $\eta_i + d\eta_i$ but the other parameters do not change. Similarly, a small change in correlation is represented by dc. They are orthogonal when

$$\langle dc, d\eta_i \rangle = E\left[\frac{\partial \log p(\boldsymbol{x}, \boldsymbol{\xi})}{\partial c} \frac{\partial \log p(\boldsymbol{x}, \boldsymbol{\xi})}{\partial \eta_i}\right] = 0, \tag{11.51}$$

since a change of parameter $d\xi_i$ in the coordinate system $\boldsymbol{\xi}$ is represented by the score function

$$s_i(\boldsymbol{x}, \boldsymbol{\xi}) = \frac{\partial \log p(\boldsymbol{x}, \boldsymbol{\xi})}{\partial \xi_i}. \tag{11.52}$$

This is a random variable representing the corresponding change of log-likelihood when ξ_i changes. The inner product (11.51) is the same as that defined by using the Fisher information matrix in the coordinates (η_1, η_2, c).

By using the coordinate system $\boldsymbol{\xi} = (\eta_1, \eta_2, c)$, let us consider the set of probability distributions $M(\eta_1, \eta_2)$ having fixed firing rates η_1 and η_2. For any (η_1, η_2), this set forms a line in S_2 in which c changes but η_1 and η_2 are fixed. There are many lines $M(\eta_1, \eta_2)$ depending on (η_1, η_2), all of which fill S_2 densely (Fig. 11.3). As for correlation c, we consider the set $E(0)$ consisting of all independent probability distributions where η_1 and η_2 are free. This is a two-dimensional subspace of S_2. We further consider the subspace $E(c)$ which consists of all the probability distributions having the same fixed measure of correlation c but firing rates change freely. The measure c defines a scale of correlations in $M(\eta_1, \eta_2)$ (see Fig. 11.3).

We search for an orthogonal measure c such that $E(c)$ and $M(\eta_1, \eta_2)$ are orthogonal to each other. That is, a small change dc is orthogonal to small changes $d\eta_i$.

Theorem 1 (Amari 2009a, 2009b, 2010). *The log-linear measure θ is orthogonal to the firing rates η_1, η_2.*

Proof We can show from direct but careful calculations that

$$\langle d\eta_i, d\theta \rangle = E\left[\frac{\partial \log p(\boldsymbol{x}, \eta_1, \eta_2, c)}{\partial \eta_i} \frac{\partial \log p(\boldsymbol{x}, \eta_1, \eta_2, c)}{\partial c}\right] = 0 \tag{11.53}$$

for $c = \theta$. □

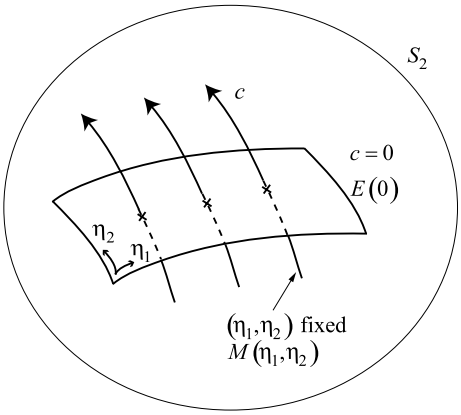

Fig. 11.3 Dual foliations by $\{E(c)\}$ and $\{M(\boldsymbol{\eta})\}$

The covariance, correlational coefficient, or other measures are not orthogonal, so that, when firing rates change, the scale of correlations changes depending on the firing rates.

From now on, we use the firing rates η_1, η_2 and the orthogonal correlation θ as a new coordinate system to specify a distribution. We call $\boldsymbol{\xi} = (\eta_1, \eta_2, \theta)$ the mixed coordinate system in which e-coordinate and m-coordinates are mixed (see Amari 2001; Nakahara and Amari 2002; Amari 2009a). Because of the orthogonality property, for two probability distributions $P(\eta_1, \eta_2, \theta)$ and $P'(\eta_1', \eta_2', \theta')$, we may say that the firing rates change from (η_1, η_2) to (η_1', η_2') and the correlation changes from θ to θ'.

11.3.2 Kullback–Leibler Divergence

We study the Kullback–Leibler divergence (KL-divergence) to decompose the difference of two probability distributions into one due to firing rates and the other due to correlation. This is a unique divergence having both invariance and flatness, see Amari (2009b).

Let $P = \{p(\boldsymbol{x})\}$ and $P' = \{p'(\boldsymbol{x})\}$ be two probability distributions. The KL-divergence between the two is defined by

$$KL\big[P : P'\big] = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})}. \tag{11.54}$$

This is not symmetric, because

$$KL\big[P : P'\big] \neq KL\big[P' : P\big] \tag{11.55}$$

in general, but it is nonnegative and vanishes if and only if $P = P'$,

$$KL[P : P] = 0, \tag{11.56}$$

$$KL\big[P : P'\big] > 0 \quad \text{for } P \neq P'. \tag{11.57}$$

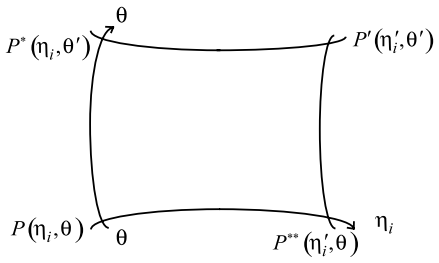

Fig. 11.4 Pythagorean decomposition of KL-divergence

When P and $P' = P + dP$ are infinitesimally close, we have

$$KL[P : P + dP] = \frac{1}{2}\sum g_{ij}\, d\xi_i\, d\xi_j \tag{11.58}$$

in the general $\boldsymbol{\xi}$-coordinates, where $G = (g_{ij})$ is the Fisher information matrix with respect to $\boldsymbol{\xi}$. Because of invariance, $KL[P : P + dP]$ is the same in any coordinate systems. Hence, it is locally a half of the square of the Riemannian distance.

Let P and P' be two distributions whose mixed coordinates are $\boldsymbol{\xi} = (\eta_1, \eta_2, \theta)$ and $\boldsymbol{\xi}' = (\eta_1', \eta_2', \theta')$. They are different both in the firing rates and correlation. Their difference can be decomposed into a sum of two terms, one due to the difference in firing rates and the other in correlation. To show this, we use the two associated distributions P^* and P^{**}. The coordinates of P^* are $(\eta_1, \eta_2, \theta')$, that is, its firing rates are the same as P, and its correlation is the same as P'. The coordinates of P^{**} are $(\eta_1', \eta_2', \theta)$, where the firing rates are the same as P', and correlation is the same as P. See Fig. 11.4.

Theorem 2 (Amari and Nagaoka 2000). *The KL-divergence is decomposed as*

$$KL\big[P' : P\big] = KL\big[P' : P^{**}\big] + KL\big[P^{**} : P\big], \tag{11.59}$$

$$KL\big[P : P'\big] = KL\big[P : P^{*}\big] + KL\big[P^{*} : P'\big]. \tag{11.60}$$

Here, $KL[P' : P^{**}]$ represents the divergence due to the change of correlation from θ' to θ, where (η_1, η_2) is fixed, and $KL[P^{**} : P]$ is that due to firing rates from (η_1', η_2') to (η_1, η_2), where θ is fixed. We have similar interpretation for the second equation.

Let P be $\boldsymbol{\xi} = (\eta_1, \eta_2, \theta)$, and P_0 be the uniform distribution without correlation, $\boldsymbol{\xi}_0 = (0.5, 0.5, 0)$, that is, $p_0(x_1, x_2) = 1/4$ for all x_1, x_2. Then, we have

$$KL[P : P_0] = KL\big[P : P^{**}\big] + KL\big[P^{**} : P_0\big], \tag{11.61}$$

where P^{**} is the independent distribution having the same firing rates (η_1, η_2) with P. Here, $KL[P : P^{**}]$ represents how far P is from the independent distribution, that is, the effect of correlation, and $KL[P^{**} : P_0]$ represents the deviation of firing rates of P from the uniform distribution.

11.3.3 Higher-Order Correlations

The theories we have studied in the case of $n = 2$ can be generalized to a neuron pool with more than two neurons. In the case of S_n, $n \geq 3$, there are higher-order correlations that cannot be reduced to the lower-order correlations. For example, in the case of three neurons, even when three pairwise correlations are 0, this does not mean that x_1, x_2, and x_3 are independent. There may exist triplewise correlation which is not reducible to pairwise correlations. Therefore, it is required to separate higher-order correlations from lower ones orthogonally.

To this end, consider marginal distributions of k random variables $x_{i_1}, \ldots, x_{i_k}$ among n variables $x_1, \ldots, x_n$. The marginal distribution of the first k variables $x_1, \ldots, x_k$, for example, is given by summing up $p(\boldsymbol{x})$ over all irrelevant random variables $x_{k+1}, \ldots, x_n$ as

$$p(x_1, \ldots, x_k) = \sum_{x_{k+1}, \ldots, x_n} p(x_1, \ldots, x_n). \tag{11.62}$$

The marginal distribution is represented by the m-coordinates of the related random variables,

$$\eta_i, \eta_{ij}, \ldots, \eta_{1\cdots k}, \quad i, j, \ldots = 1, \ldots, k. \tag{11.63}$$

Any marginal distribution $p(x_{i_1}, \ldots, x_{i_k})$ is represented similarly by the joint firing probabilities $\eta_{i_1 \cdots i_p}$, $1 \leq p \leq k$, of the related variables, no more than k variables. We represent the joint firing rates of all combinations of no more than k variables by

$$\boldsymbol{\eta}_k = (\eta_i, \eta_{ij}, \ldots, \eta_{i_1 \cdots i_k}), \tag{11.64}$$

which consist of a part of the coordinates $\boldsymbol{\eta}$. This includes n firing rates η_i, ${}_nC_2$ joint firing rates η_{ij} of two neurons x_i and x_j, until ${}_nC_k$ joint firing rates of any k variables, where ${}_nC_k$ is the binomial coefficient,

$${}_nC_k = \frac{n!}{k!(n-k)!}. \tag{11.65}$$

In other words, $\boldsymbol{\eta}_k$ is the first s components of

$$\boldsymbol{\eta} = (\eta_i, \eta_{ij}, \ldots, \eta_{i_1 \cdots i_k};\ \eta_{i_1 \cdots i_{k+1}}, \ldots, \eta_{1\cdots n}), \tag{11.66}$$

where

$$s = \sum_{p=1}^{k} {}_nC_p. \tag{11.67}$$

In the case of $n = 3$, when $k = 1$, the marginal distributions of neurons are given by the firing rates

$$\boldsymbol{\eta}_1 = (\eta_1, \eta_2, \eta_3). \tag{11.68}$$

The marginal distributions of pairs of neurons ($k = 2$) are given by

$$\boldsymbol{\eta}_2 = (\eta_1, \eta_2, \eta_3;\ \eta_{12}, \eta_{23}, \eta_{31}). \tag{11.69}$$

Let us consider the submanifold $M(\boldsymbol{\eta}_k)$ in which any distributions of n random variables $\boldsymbol{x}$ have the same marginal distributions of k variables, given by $\boldsymbol{\eta}_k$. They differ only in higher-order correlations.

In a similar way, we divide the $\boldsymbol{\theta}$ coordinates into two parts

$$\boldsymbol{\theta} = (\boldsymbol{\theta}_k, \boldsymbol{\theta}_{n-k}), \tag{11.70}$$

in which the former part $\boldsymbol{\theta}_k$ consists of the components with at most k indices, $i_1, \ldots, i_p$, $p \leq k$, and the latter part having indices more than k. Hence, $\boldsymbol{\theta}_{n-k}$ denote the last $2^n - 1 - s$ coordinates of $\boldsymbol{\theta}$. The indices of $\boldsymbol{\theta}_{n-k}$ are complements to those of $\boldsymbol{\eta}_k$ (but in the $\boldsymbol{\theta}$ coordinates),

$$\boldsymbol{\theta}_{n-k} = (\theta_{i_1 \cdots i_{n-k+1}}, \ldots, \theta_{12\cdots n}). \tag{11.71}$$

They show the log-interaction terms of more than k variables. Let $E(\boldsymbol{\theta}_{n-k})$ be the submanifold where all the distributions have the same fixed $\boldsymbol{\theta}_{n-k}$ but $\boldsymbol{\theta}_k$ are free.

This is an s-dimensional submanifold. In the case of $n = 3$, the complement of $\boldsymbol{\eta}_1$ is

$$\boldsymbol{\theta}_2 = (\theta_{12}, \theta_{23}, \theta_{31}, \theta_{123}), \tag{11.72}$$

and the complement of $\boldsymbol{\eta}_2$ is

$$\boldsymbol{\theta}_3 = (\theta_{123}). \tag{11.73}$$

Now we show that coordinates $\boldsymbol{\eta}_k$ are orthogonal to $\boldsymbol{\theta}_{n-k}$ for any k (see Amari and Nagaoka 2000; Amari 2001).

Theorem 3 *The two families of submanifolds $M(\boldsymbol{\eta}_k)$ and $E(\boldsymbol{\theta}_{n-k})$ are orthogonal to each other.*

All the marginal correlations within k neurons are fixed in $M(\boldsymbol{\eta}_k)$, so that for two distributions P, P' belonging to the same $M(\boldsymbol{\eta}_k)$, their difference is solely due to the higher-order correlations of more than k neurons. The parameters orthogonal to $\boldsymbol{\eta}_k$ are given by $\boldsymbol{\theta}_{n-k}$, so that we may say that $\boldsymbol{\theta}_{n-k}$ represents higher-order correlations of neurons, higher than order k, which are orthogonal to the firing rates and lower-order correlations up to k variables. We may use the mixed coordinates

$$\boldsymbol{\xi} = (\boldsymbol{\eta}_k, \boldsymbol{\theta}_{n-k}) \tag{11.74}$$

for extracting higher-order correlations.

By using the mixed coordinates, we have the orthogonal decomposition of the KL-divergence.

Theorem 4 (Amari 2001). *The KL-divergence between P and P' is decomposed as*

$$KL\left[P' : P\right] = KL\left[P' : P^{**}\right] + KL\left[P^{**} : P\right], \tag{11.75}$$

$$KL\left[P : P'\right] = KL\left[P : P^{*}\right] + KL\left[P^{*} : P'\right], \tag{11.76}$$

where the coordinates of P^* and P^{**} are, respectively, given by the mixed coordinates

$$\boldsymbol{\xi}^* = \left(\boldsymbol{\eta}_k, \boldsymbol{\theta}'_{n-k}\right), \qquad \boldsymbol{\xi}^{**} = \left(\boldsymbol{\eta}'_k, \boldsymbol{\theta}_{n-k}\right). \tag{11.77}$$

Here, $KL[P' : P^{**}]$ represents the divergence responsible for difference only in higher-order correlations, higher than k, and KL, $[P^{**} : P]$ represents the divergence purely responsible for the change in the marginal distributions of degrees less than or equal to k.

11.4 Tractable Models of Probability Distributions

The full model S_n of all the probability distributions includes $2^n - 1$ parameters. Even when $n = 10$, the number is 1,023. Hence, even though the full model is theoretically well defined, this is not useful for the purpose of analyzing experimental data. It is impossible to estimate all the parameters from data. Therefore, we need much simpler models for analyzing real data. We show here some of them (see also Nakahara et al. 2006). We present a mixture model in the next section.

11.4.1 Homogeneous Model

A model is said to be homogeneous if the roles of all the neurons are the same. In other words, the probability distribution $p(x_1, \ldots, x_n)$ of a homogeneous model is invariant under a permutation of variables $x_1, \ldots, x_n$.

In this case, the firing rates of all neurons are the same,

$$\eta_i = \bar{\eta}_1, \quad i = 1, \ldots, n, \tag{11.78}$$

and the joint firing rates of any two neurons are the same, and so on,

$$\eta_{ij} = \bar{\eta}_2, \quad i \neq j,\ i, j = 1, \ldots, n, \tag{11.79}$$

$$\eta_{ijk} = \bar{\eta}_3, \quad i, j, k = 1, \ldots, n, \tag{11.80}$$

$$\cdots \tag{11.81}$$

$$\eta_{12\cdots n} = \bar{\eta}_n. \tag{11.82}$$

Therefore, in order to specify a distribution, we need only n parameters,

$$\bar{\boldsymbol{\eta}} = (\bar{\eta}_1, \bar{\eta}_2, \ldots, \bar{\eta}_n). \tag{11.83}$$

This implies that the homogeneous model is a subspace of S_n specified by (11.78)–(11.82) that are linear in the $\boldsymbol{\eta}$-coordinates. Hence, the model is a linear subspace of S_n in the $\boldsymbol{\eta}$-coordinates. In other words, it is m-flat in terms of information geometry. The m- and e-flat structures play a fundamental role in information geometry (Amari and Nagaoka 2000).

Correspondingly, a homogeneous model satisfies

$$\theta_i = \bar{\theta}_1, \tag{11.84}$$

$$\theta_{ij} = \bar{\theta}_2, \tag{11.85}$$

$$\cdots \tag{11.86}$$

$$\theta_{1\cdots n} = \bar{\theta}_n \tag{11.87}$$

in the $\boldsymbol{\theta}$-coordinates system, too, so that it is also an exponential family and e-flat. The distributions is written as

$$p(\boldsymbol{x}; \bar{\boldsymbol{\theta}}) = \exp\left\{ \frac{1}{n}\bar{\theta}_1 \sum x_i + \frac{2}{n(n-1)}\bar{\theta}_2 \sum x_i x_j + \cdots + \bar{\theta}_n x_1 \cdots x_n - \bar{\psi}(\bar{\boldsymbol{\theta}}) \right\}. \tag{11.88}$$

The $\bar{\boldsymbol{\theta}}$ coordinates and $\bar{\boldsymbol{\eta}}$-coordinates correspond to each other, and their relation is given by

$$\bar{\eta}_i = \frac{\partial}{\partial \bar{\theta}_i}\bar{\psi}(\bar{\boldsymbol{\theta}}). \tag{11.89}$$

Theoretical analysis of such a model is given in Bohte et al. (2000) and Amari et al. (2003).

Montani et al. (2009) analyzed experimental data by using the homogeneous model to demonstrate the effects of higher-order correlations. However, the assumption of homogeneity might not hold in many cases. It is possible to extend the model to include two types of neurons such that homogeneity holds only within each type.

11.4.2 Boltzmann Machine

A Boltzmann machine is a neural model having a Markovian state transition rule (Ackley et al. 1985). Its stationary probability distribution is written as

$$p(\boldsymbol{x}) = \exp\left\{ \sum \theta_i x_i + \sum_{i<j} \theta_{ij} x_i x_j - \psi(\boldsymbol{\theta}) \right\}. \tag{11.90}$$

This is the same as the Ising spin model used in physics. One can easily see that the set of distributions is of exponential type and is a submodel of S_n such that all higher-order correlations, higher than two, vanish,

$$\theta_{ijk} = \cdots = \theta_{1\cdots n} = 0. \tag{11.91}$$

Hence, firing rates are arbitrary, and pairwise correlations exist, but higher-order correlations of order higher than two do not exist.

When the model includes extra neurons called hidden neurons whose states are denoted by random variables $y_1, \ldots, y_m$, even when the entire distribution is of

Boltzmann type, the marginal distributions are

$$p(\boldsymbol{x}) = \sum_{y_1,\ldots,y_m} \exp\Big\{\sum \theta_i x_i + \sum \theta_{ij} x_i x_j + \sum \theta'_i y_i + \sum \theta'_{ij} y_i y_j + \sum \theta''_{ij} x_i y_j - \psi\Big\}. \tag{11.92}$$

This is no more an exponential family. Moreover, they include higher-order correlations. However, the analysis of such an extended model is not easy in general (Amari et al. 1992).

11.4.3 Marginal Models

It is difficult to handle a firing pattern $\boldsymbol{x}$ of n neurons as a whole. We often pick up a small number of neurons at one time and study its characteristics. When we pick up single neurons, all the correlations are discarded. Hence, the firing rates of neurons are the only quantities to be studied. This is trivial, and it is the marginal model M_1 of order 1. We have marginal probability distributions $p_i(x_i)$ of neuron i, and they are characterized by the firing rates $\boldsymbol{\eta}_1 = (\eta_1, \ldots, \eta_n)$. For each neuron, say neuron i, we have the firing rate η_i, and the dual quantity in model $M_1 = \{p_i(x_i)\}$ is

$$\theta_i^* = \log \frac{\eta_i}{1-\eta_i}. \tag{11.93}$$

Note that θ_i^* is different from θ_i of the full model S_n, because θ_i depends on all $(\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n)$. This is shown from the relation

$$(g_{ij})^{-1} = \frac{\partial \theta_i}{\partial \eta_j}, \tag{11.94}$$

which is not a diagonal matrix in general. The explicit form of θ^i in terms of $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_n$ is complicated.

We then consider a pair of neurons (i, j). There are $n(n-1)/2$ pairs of neurons. The firing rates and joint firing rate of neurons i, j are η_i, η_j, and η_{ij}. We marginalize $p(\boldsymbol{x})$ to

$$p_{ij}(x_i, x_j) = \sum p(x_1, \ldots, x_n), \tag{11.95}$$

where the summation is taken over $x_1, \ldots, x_n$ except for x_i and x_j. We then have the $\boldsymbol{\eta}$-coordinates $(\eta_i, \eta_j, \eta_{ij})$, and the corresponding $\boldsymbol{\theta}$-coordinates are calculated from the marginal model M_2 consisting of $p_{ij}(x_i, x_j)$. We denote the second-order interaction term of this marginal distribution by θ_{ij}^*,

$$\theta_{ij}^* = \log \frac{\eta_{ij}(1-\eta_i-\eta_j+\eta_{ij})}{(\eta_i-\eta_{ij})(\eta_j-\eta_{ij})}. \tag{11.96}$$

We cannot write θ_{ij} of S_n in terms of the marginal quantities η_i, η_j, and η_{ij}.

We further consider triplets of neurons i, j, k and the marginal distributions $p_{ijk}(x_i, x_j, x_k)$. Its $\boldsymbol{\eta}$-coordinates are the corresponding part of $\boldsymbol{\eta}_3$, but the $\boldsymbol{\theta}$-coordinates are different from those of $\boldsymbol{\theta}$. Let θ^*_{ijk} be the third-order interaction term of the marginal model M_3. We then have new coordinates $(\theta^*_i, \theta^*_{ij}, \theta^*_{123})$ in M_3.

In this way, we consider a sequence of marginal models $M_1, M_2, \ldots, M_n$. We have the same sequence of $\boldsymbol{\eta}$-coordinates

$$\boldsymbol{\eta} = (\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \ldots, \boldsymbol{\eta}_n), \tag{11.97}$$

representing joint firing rates. We also have a sequence

$$\boldsymbol{\theta}^* = \left(\boldsymbol{\theta}^*_1, \boldsymbol{\theta}^*_2, \boldsymbol{\theta}^*_3, \ldots, \boldsymbol{\theta}^*_n\right), \tag{11.98}$$

where $\boldsymbol{\theta}^*_k$ are the kth-order interactions of the marginal distributions M_k of k neurons. We should note that $\boldsymbol{\theta}^*_1, \ldots, \boldsymbol{\theta}^*_{k-1}$ are different from the interaction terms of the marginal model M_k of k neurons. For example, $\boldsymbol{\theta}^*_{k-1}$ are the interactions of the marginal distributions of $k-1$ neurons in M_{k-1} and are not in M_k nor in M_n.

We call $\boldsymbol{\theta}^*$ the marginal interaction coordinates. They have the following properties.

Theorem 5

(1) $\boldsymbol{\theta}^*_k$ *are functions of* $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_k$.
(2) $\boldsymbol{\theta}^*_k$ *are orthogonal to* $(\boldsymbol{\theta}^*_1, \ldots, \boldsymbol{\theta}^*_{k-1})$.

Proof Since $\boldsymbol{\theta}^*_k$ are the interactions of the marginal distributions of k neurons, they can be written as functions of $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_k$. The interactions $\boldsymbol{\theta}^*_k$ in M_k are orthogonal to $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{k-1}$. Since $\boldsymbol{\theta}^*_1, \ldots, \boldsymbol{\theta}^*_{k-1}$ are functions of $\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_{k-1}$, $\boldsymbol{\theta}^*_k$ is orthogonal to them. □

The marginal model of order k is useful when we neglect interactions of orders higher than k. Even when interactions higher than k exist, their effects are not simply neglected, but included in $\boldsymbol{\theta}^*_k$, because of marginalization. For example, consider the marginal model M_2 consisting of three neurons, S_3. The full model is represented by $\boldsymbol{\eta} = (\eta_i, \eta_{ij}, \eta_{123})$ and the interaction coordinates $(\theta_i, \theta_{ij}, \theta_{123})$. Pairwise interactions $\boldsymbol{\theta}^*_2$ are different from $\boldsymbol{\theta}_2$. Even when $\theta_{12} = 0$, θ^*_{12} is not necessarily equal to 0. This is because the pairwise interaction of neurons exists through the third neuron x_3 and also θ^*_{12} includes term due to θ_{123}. Hence, one may say that $\boldsymbol{\theta}^*_k$ includes the functional connectivity even when there are no direct connections.

The marginal model provides a reasonable framework when we disregard higher-order interactions but take their effect into account. The marginal model is new, and its details will appear soon (Shimazaki et al. 2010).

11.4.4 Higher-Order Correlations Generated from Common Inputs

We use a simple toy model in order to show how higher-order correlations emerge. We also study the probability distribution of the ensemble activity in a pool of neurons and show how it is related to higher-order correlations. The model is a pool of McCulloch–Pitts neurons receiving a common input. Let s_i be the ith input, and suppose that neuron j receives it with synaptic weight w_{ji}. Then the total weighted sum of the inputs to neuron j is

$$u_j = \sum w_{ji} s_i - h, \tag{11.99}$$

where we subtracted a common threshold h. The output of neuron j is

$$x_j = 1(u_j), \tag{11.100}$$

where 1 is the Heaviside function

$$1(u) = \begin{cases} 1, & u > 0, \\ 0, & u \le 0. \end{cases} \tag{11.101}$$

The outputs $x_1, \ldots, x_n$ are correlated because they use common inputs s_i.

When the connections w_{ij} are randomly assigned independently and identically, because of the central limit theorem, we see that u_j are subject to a joint Gaussian distribution, in which they have the same variance $\sigma_i^2 = \sigma^2$ and the same covariance ρ for any pair u_j, u_k. There are no higher-order correlations because they are Gaussian. We normalize the variance of u_i equal to 1, and let ρ be the normalized common covariance. Then we may put

$$u_i = \sqrt{(1-\rho)}v_i + \sqrt{\rho}\varepsilon - h, \tag{11.102}$$

where $v_1, \ldots, v_n$ and ε are independent Gaussian random variables subject to the standard Gaussian distribution $N(0, 1)$. Here, u_i are indeed subject to a joint Gaussian distribution with mean $-h$ and variance–covariance matrix, $\mathrm{V}[u_i] = 1$, $\mathrm{Cov}[u_i, u_j] = \rho$.

The joint distribution of $x_1, \ldots, x_n$ is rather complicated, because it includes higher-order correlations in spite that the inputs u_i do not include higher-order correlations. The $\boldsymbol{\theta}$-coordinates of the probability distribution $p(\boldsymbol{x}, \rho, h)$ of $\boldsymbol{x}$ are analyzed in Amari et al. (2003), and shown to include higher-order correlations even in this simple case. Let us present the central arguments.

We consider the activity of the pool, that is, the ensemble firing rate,

$$a = \frac{1}{n} \sum x_i, \tag{11.103}$$

which shows the ratio of neurons excited in the pool. This is a random variable. When all x_i are independent, due to the law of large numbers, the distribution of a is concentrated at its mean value $\bar{a}$. In such a case, the activity of the pool is always the same, although different combinations of neurons fire each time. When

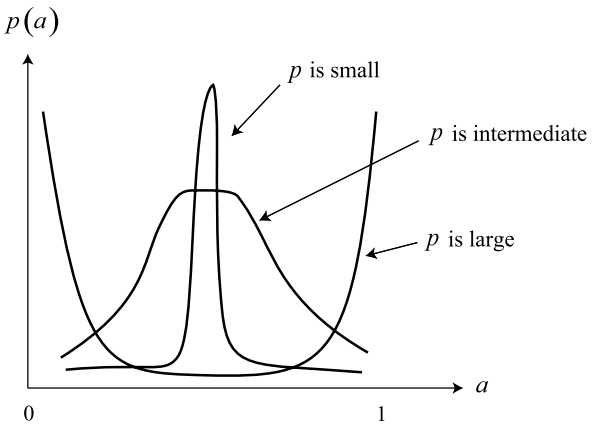

Fig. 11.5 Distributions $p(a)$ of activities of a pool for various p

the distribution $p(a)$ of a is widely spread, or even is bimodal, a large number of neurons fire in one time, while only a small number of neurons fire in another time. This type of widely spread distribution is a characteristic of a neuronal ensemble to have synchronized firing behavior. See Montani et al. (2009) for the study of the distribution $p(a)$, both theoretically and experimentally.

The distribution of a is calculated, when n is large, by the following theorem (Amari et al. 2003).

Theorem 6 *The distribution of a is given asymptotically by*

$$p(a;\rho,h)=c\exp\left[\frac{2\rho-1}{2(1-\rho)}\left\{F^{-1}(a)-\frac{\sqrt{\rho}}{2\rho-1}h\right\}^2\right], \tag{11.104}$$

where c is a normalization constant, F is the error function of the form

$$F(u)=\frac{1}{\sqrt{2\pi}}\int_{(h-\rho u)/\sqrt{1-\rho}}^{\infty}\exp\left\{-\frac{x^2}{2}\right\}dx, \tag{11.105}$$

and F^{-1} is its inverse function.

From the theorem we see how the shape of $p(a;\rho,h)$ changes as the correlation ρ increases. In the range of $0<\rho<1/2$, $p(a;\rho,h)$ is unimodal, having a peak at $a=F(\frac{\sqrt{\rho}}{2\rho-1}h)$. When $\rho=1/2$, it is a uniform distribution, and when $\rho>1/2$, it is bimodal, having peaks at $a=0$ and $a=1$ (Fig. 11.5).

11.5 Mixture Model and Its Dynamics

Let $\boldsymbol{u}=(u_1,\ldots,u_n)$ be a vector to show firing rates of neurons, that is, $u_i=\eta_i$. We call $\boldsymbol{u}$ a firing rate vector. Let us denote by $p(\boldsymbol{x},\boldsymbol{u})$ a probability distribution with firing vector $\boldsymbol{u}$,

$$\boldsymbol{u}=E[\boldsymbol{x}]=\sum_{\boldsymbol{x}}\boldsymbol{x}\,p(\boldsymbol{x},\boldsymbol{u}). \tag{11.106}$$

When $p(\boldsymbol{x}, \boldsymbol{u})$ is independent, we have

$$p(\boldsymbol{x}, \boldsymbol{u}) = \prod_{i=1}^{n} p(x_i, u_i), \tag{11.107}$$

where $p(1, u) = u$ and $p(0, u) = 1 - u$.

We begin with a mixture of two independent distributions $p(\boldsymbol{x}, \boldsymbol{u})$ and $p(\boldsymbol{x}, \boldsymbol{v})$. We consider their mixture

$$p(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}, t) = tp(\boldsymbol{x}, \boldsymbol{u}) + (1 - t)p(\boldsymbol{x}, \boldsymbol{v}), \tag{11.108}$$

where t is the mixing rate. This model chooses $p(\boldsymbol{x}, \boldsymbol{u})$ with probability t and $p(\boldsymbol{x}, \boldsymbol{v})$ with probability $1 - t$ each time and then generates $\boldsymbol{x}$ from the chosen distribution. Even when the component distributions are independent, $p(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}, t)$ is not.

Let us introduce a new random variable y which takes 1 with probability t and 0 with probability $1 - t$. We consider a conditional distribution $p(\boldsymbol{x}|y)$ given by

$$p(\boldsymbol{x}|y = 1) = p(\boldsymbol{x}, \boldsymbol{u}), \tag{11.109}$$

$$p(\boldsymbol{x}|y = 0) = p(\boldsymbol{x}, \boldsymbol{v}). \tag{11.110}$$

Then, the joint distribution of $\boldsymbol{x}$ and y is

$$p(\boldsymbol{x}, y) = p(\boldsymbol{x}|y)p(y), \tag{11.111}$$

and the mixture model (11.108) is the marginal distribution of $p(\boldsymbol{x}, y)$,

$$p(\boldsymbol{x}) = \sum_{y} p(\boldsymbol{x}, y). \tag{11.112}$$

The mixture model is hence considered as a model including a hidden variable y which affects the probability of $\boldsymbol{x}$ conditionally.

We study the various quantities of the mixture model. The firing rate vector is simply given by the mixture of the two,

$$\eta_i = tu_i + (1 - t)v_i. \tag{11.113}$$

Furthermore, the $\boldsymbol{\eta}$-coordinates of the mixture model are simply the mixtures of the respective components,

$$\eta_{i_1 \cdots i_k} = t\eta^1_{i_1 \cdots i_k} + (1 - t)\eta^2_{i_1 \cdots i_k}, \tag{11.114}$$

where $\eta^1_{i_1 \cdots i_k}$ and $\eta^2_{i_1 \cdots i_k}$ are the $\boldsymbol{\eta}$-coordinates of the component distributions.

The behavior of pairwise covariance c_{ij} of x_i and x_j is interesting, as is shown in the following theorem. Here, we do not assume that $p(\boldsymbol{x}, \boldsymbol{u})$ and $p(\boldsymbol{x}, \boldsymbol{v})$ are independent distributions.

Theorem 7 *The covariances among two neurons in the mixture model are given by*

$$c_{ij} = tc^1_{ij} + (1 - t)c^2_{ij} + t(1 - t)(u_i - v_i)(u_j - v_j), \tag{11.115}$$

where c^1_{ij} and c^2_{ij} are those of the component distributions $p(\boldsymbol{x}, \boldsymbol{u})$ and $p(\boldsymbol{x}, \boldsymbol{v})$. They are 0 for independent distributions.

The proof is straightforward. The theorem shows that the covariances are not mere mixtures of those of the components. The correlations emerge even when the component distributions are independent. They are positive or negative, depending on the differences of their firing rates. It is not easy to obtain explicit forms of the $\boldsymbol{\theta}$-coordinates, but it is sure that the higher-order correlations exist in a mixture model even when the component distributions are independent.

We give a simple but interesting interpretation of (11.115). Let $p_A(\boldsymbol{x})$ and $p_B(\boldsymbol{x})$ be two distributions, which are not necessarily independent. The distributions are assumed to represent stochastic activities of two Hebbian assemblies A and B of neurons. A Hebbian assembly A is a set of neurons such that, when A is active, the neurons in A have high firing rates. Its distribution is given by $P_A(\boldsymbol{x})$. Assume that there are two Hebbian assemblies A and B and that the state of the neuron pool changes from $p_A(\boldsymbol{x})$ to $p_B(\boldsymbol{x})$ gradually by a dynamical process. We assume that the intermediate state is the mixture, depending on the transition time t, $0 \le t \le 1$. The intermediate state at time t is given by the mixture model

$$p(\boldsymbol{x}, t) = (1-t)p_A(\boldsymbol{x}) + tp_B(\boldsymbol{x}). \tag{11.116}$$

The pairwise covariances of neurons in the initial and last states are denoted by c_{ij}^A and c_{ij}^B, respectively. The firing rate vectors are $\boldsymbol{u}^A$ and $\boldsymbol{u}^B$. It is interesting to see how the covariances change in the process of state transition. From (11.115) we have

$$c_{ij}(t) = (1-t)c_{ij}^A + tc_{ij}^B + t(1-t)\left(u_i^A - u_i^B\right)\left(u_j^A - u_j^B\right). \tag{11.117}$$

Hence, for two neurons i and j belonging to A and not to B, the firing rates of both neurons decrease, $u_i^A > u_i^B$ and $u_j^A > u_j^B$ by the state transition. Hence, their covariance increases in the process of state transition more than their mixture. For neurons i and j belonging to B and not to A, the situation is the opposite. But their covariance also increases in the process of state transition. However, for neuron i belonging to A and neuron j belonging to B, $u_i^A - u_i^B > 0$ and $u_j^A - u_j^B < 0$. Hence, their covariance decreases in the middle of the state transition. This shows that the covariances of two neurons belonging to the same Hebbian assembly increase in the intermediate state, while those of two neurons belonging to different assemblies decrease, even becoming negative.

We generalize the mixture model so that it consists of more than two components. Let $\boldsymbol{u}^1, \ldots, \boldsymbol{u}^m$ be m firing vectors, and let $p\ (\boldsymbol{x}, \boldsymbol{u}^i)$, $i = 1, \ldots, m$, be the corresponding independent distributions. We introduce a random variable y which takes values $1, \ldots, m$. Let w_i be the probability of $y = i$. The general mixture model is defined by

$$p\left(\boldsymbol{x}, \boldsymbol{u}^1, \ldots, \boldsymbol{u}^m, \boldsymbol{w}\right) = \sum w_i\, p\left(\boldsymbol{x}, \boldsymbol{u}^i\right), \tag{11.118}$$

where $\sum w_i = 1$. We define the joint probability distributions of $\boldsymbol{x}$ and y by

$$p(\boldsymbol{x}, y) = w_i\, p\left(\boldsymbol{x}, \boldsymbol{u}^i\right) \quad \text{when } y = i. \tag{11.119}$$

Then, $p(\boldsymbol{x}, \boldsymbol{u}^i)$ is regarded as the conditional probability,

$$p\left(\boldsymbol{x}, \boldsymbol{u}^i\right) = p(\boldsymbol{x}|y = i), \tag{11.120}$$

and the mixture model is the marginal model

$$p\left(\boldsymbol{x}, \boldsymbol{u}^1, \ldots, \boldsymbol{u}^m, \boldsymbol{w}\right) = \sum_y p(\boldsymbol{x}, y). \tag{11.121}$$

Let $\boldsymbol{r}$ be the expectation of the firing rate vector of a mixture model,

$$\boldsymbol{r} = E[\boldsymbol{x}] = \sum w_i \boldsymbol{u}^i, \tag{11.122}$$

and define the vectors

$$\boldsymbol{\delta}^i = \boldsymbol{u}^i - \boldsymbol{r} \tag{11.123}$$

showing the deviations of $\boldsymbol{u}^i$ from the average.

In order to show higher-order interactions, we define the kth-order central moments of $\boldsymbol{x}$,

$$c_{i_1 \cdots i_k} = E\left[(x_{i_1} - r_{i_1}) \cdots (x_{i_k} - r_{i_k})\right]. \tag{11.124}$$

The covariance is a special case of $k = 2$. We then have the following theorem.

Theorem 8 (Amari 2010). *The kth-order central moment of a mixture model is*

$$c_{i_1 \cdots i_k} = \sum w^i \delta^i_{i_1} \cdots \delta^i_{i_k}. \tag{11.125}$$

We can use a general mixture model to study the dynamics of Hebbian assemblies (see Chap. 10). Let us consider the case where a pool of neurons is divided into m Hebbian assemblies $H_1, \ldots, H_m$, where a neuron may belong to a number of assemblies. When H_i is active, we assume that its firing probability is $p(\boldsymbol{x}, \boldsymbol{u}^i)$. Then, the overall probability is their mixture,

$$p(\boldsymbol{x}, \boldsymbol{w}) = \sum w_i \, p\left(\boldsymbol{x}, \boldsymbol{u}^i\right), \tag{11.126}$$

where $p(\boldsymbol{x}, \boldsymbol{u}^i)$ may be an independent distribution or a more general one.

If the mixture vector $\boldsymbol{w}$ changes dynamically, depending on the activities of neurons and external stimuli I, we have the control equation

$$\frac{d\boldsymbol{w}}{dt} = \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{x}, I). \tag{11.127}$$

The mixture model changes in this way dynamically. Such a model is proposed in Amari (2010).

We finally remark that many existing models for correlated spike generation are special cases of the mixture model. For example, the additive interaction model and the eliminating interaction model (Kuhn et al. 2002; Feng and Brown 2000; Kuhn et al. 2003), and a more general replacement model (Niebur 2007) belong to the mixture model with $m = 2$. The Bretta model (Brette 2009) is also a special case with large m.

11.6 Temporal Correlations of Spikes in a Single Neuron

A train of spikes of a neuron are not usually temporally independent but have temporal correlations. We study its stochastic nature from the information-geometric viewpoint. We first give the full model and then various simple models.

11.6.1 Full Model of Train of Spikes and Stationary Model

Let $x(1), \ldots, x(T)$ be random variables representing spikes at time bin $t = 1, \ldots, T$ of a neuron. Let $\boldsymbol{X}^T = (x(1), \ldots, x(T))$ be its vector notation, and let $p(\boldsymbol{X}^T)$ be its probability distribution. We then have the following two types of coordinates: joint firing coordinates (m-coordinates) are

$$\eta_i = E\big[x(i)\big], \tag{11.128}$$

$$\eta_{ij} = E\big[x(i)x(j)\big], \tag{11.129}$$

$$\eta_{ijk} = E\big[x(i)x(j)x(k)\big], \tag{11.130}$$

$$\cdots \tag{11.131}$$

and interaction coordinates (e-coordinates) are

$$\theta_i, \theta_{ij}, \theta_{ijk}, \ldots, \tag{11.132}$$

where $\theta_{i_1 \cdots i_k}$ are the coefficients of the expansion of the log-probability,

$$\log p\big\{\boldsymbol{X}^T\big\} = \sum \theta_i x(i) + \sum \theta_{ij} x(i)x(j) + \cdots + \theta_{1\ldots T} x(1) \cdots x(T) - \psi. \tag{11.133}$$

This is the full model which includes so many parameters and is too general for practical use. There are two ways of simplification: One is the stationarity assumption under which the stochastic properties do not change over time. In this case, all the firing rates are the same,

$$\eta_1 = \eta_2 = \cdots \eta_T = \eta^{(1)}; \qquad \theta_1 = \theta_2 = \cdots = \theta_T = \theta^{(1)}. \tag{11.134}$$

Further, the pairwise firing rates η_{ij} depend only on the difference of two time bins i, j,

$$\eta_{ij} = \eta^{(2)}_{|j-i|}. \tag{11.135}$$

Higher-order joint firing probabilities also have this type of stationarity.

11.6.2 Markov Chain

When the probability $p\{x(t)\}$ at time t is determined based on the history of firing at k past time bins, $\boldsymbol{x}^k(t) = \{x(t-k), \ldots, x(t-1)\}$, it is a kth-order Markov chain.

It is given by the conditional probability distribution $p(x(t)|\boldsymbol{x}^k(t))$. We first study the simplest case of the first-order Markov chain, where the conditional probability is given by $p(x(t)|x(t-1))$ based on Amari (2010). This gives a (2×2)-state transition matrix,

$$a_{ij} = p\big(x(t) = i \big| x(t-1) = j\big), \quad i, j = 0, 1, \tag{11.136}$$

and obviously

$$\sum_{i=1}^{2} a_{ij} = 1. \tag{11.137}$$

Hence, the number of free parameters is 2, and the set of all first-order Markov chains is a two-dimensional manifold M.

The Markov chain has a stationary distribution $\bar{p}(x)$, $\bar{p}_i = \text{Prob}\{x = i\}$, such that

$$\sum_{x'} p\big(x\big|x'\big)\bar{p}\big(x'\big) = \bar{p}(x), \tag{11.138}$$

$$\sum_{j} a_{ij}\bar{p}_j = \bar{p}_i. \tag{11.139}$$

The firing rate η_1 of the neuron is given by

$$\eta_1 = E\big[x(t)\big] = \text{Prob}\big\{x(t) = 1\big\} = \bar{p}_1, \tag{11.140}$$

which is solved as

$$\bar{p}_1 = \frac{a_{10}}{a_{10} + a_{01}}. \tag{11.141}$$

We next consider the probability of successive spikes,

$$\eta_{11} = E\big[x(t)x(t-1)\big] = \text{Prob}\big\{x(t) = x(t-1) = 1\big\}. \tag{11.142}$$

This is given by

$$\eta_{11} = \bar{p}_1 a_{11} = \frac{a_{10}a_{11}}{a_{10} + a_{01}}. \tag{11.143}$$

This is responsible for the temporal correlation of spikes. The temporal covariance

$$c = \text{Cov}\big[x(t), x(t-1)\big] \tag{11.144}$$

is given by

$$c = \eta_{11} - \eta_1^2 = \frac{a_{10}a_{01}(a_{11} - a_{10})}{(a_{10} + a_{01})^2}. \tag{11.145}$$

The two quantities η_1 and η_{11} play the role of a coordinate system of M. We now consider a sequence $\boldsymbol{X}^T = x(1), \ldots, x(T)$ of length T. Then, its probability is

$$p\big(\boldsymbol{X}^T\big) = p\big\{x(1)\big\} \prod_{t=1}^{T-1} p\big(x(t+1)\big|x(t)\big) \tag{11.146}$$

$$= p\big\{x(1)\big\} \prod_{t=1}^{T-1} \frac{p\{x(t+1)x(t)\}}{p(x(t))}. \tag{11.147}$$

Its logarithm divided by T is asymptotically given by

$$\frac{1}{T} \log p\big(\boldsymbol{X}^T\big) = \frac{1}{T} \sum \big\{\log p\big\{x(t+1)x(t)\big\} - \log p\big(x(t)\big)\big\}, \tag{11.148}$$

where the small order term of $\log p\{x(1)\}/T$ is neglected.

Now we introduce the random variables

$$n_1 = \sum_{t=1}^{T} x(t), \tag{11.149}$$

$$n_{11} = \sum_{t=1}^{T-1} x(t)x(t+1) \tag{11.150}$$

that represent the number of spikes and the number of successive spikes. Their rates are, for large T,

$$f_1 = \frac{n_1}{T}, \tag{11.151}$$

$$f_{11} = \frac{n_{11}}{T}. \tag{11.152}$$

We show that the Markovian model M is an exponential family for large T:

Theorem 9 (Amari 2001). *The probability distribution of Markov chain is written as*

$$p\big(\boldsymbol{X}^T; \theta_1, \theta_2\big) = \exp\{\theta_1 f_1 + \theta_2 f_{11} - \psi\}, \tag{11.153}$$

where f_1 and f_{11} are random variables determined from $\boldsymbol{X}^T$, ψ corresponds to the normalization factor, and θ_1, θ_2 are given, respectively, by

$$\theta_1 = \log \frac{a_{01}a_{10}}{a_{00}^2}, \tag{11.154}$$

$$\theta_2 = \log \frac{a_{11}a_{00}}{a_{10}a_{01}}. \tag{11.155}$$

The transformation between $\boldsymbol{\theta}$- and $\boldsymbol{\eta}$-coordinates is

$$\theta_1 = \log \frac{(\eta_1 - \eta_{11})^2}{(1 - 2\eta_1 + \eta_{11})^2} \frac{1 - \eta_1}{\eta_1}, \tag{11.156}$$

$$\theta_2 = \log \frac{\eta_{11}(1 - 2\eta_1 + \eta_{11})}{(\eta_1 - \eta_{11})^2}. \tag{11.157}$$

We also have the relation

$$a_{11} = \frac{\eta_{11}}{\eta_1}, \qquad a_{01} = \frac{\eta_1 - \eta_{11}}{\eta_1}, \qquad a_{10} = \frac{\eta_1 - \eta_{11}}{1 - \eta_1}, \qquad a_{00} = \frac{1 - 2\eta_1 + \eta_{11}}{1 - \eta}. \tag{11.158}$$

Theorem 10 (Amari 2010). *The coordinate θ_2 is a measure of temporal correlation orthogonal to the firing rate η_1.*

We can generalize the above theory to the kth-order Markov chain. Here, we state the case with $k = 2$, where the state transition is given by $p\{x(t)|x(t-1)x(t-2)\}$. We have the stationary distribution $\bar{p}\{x(t-1)x(t-2)\}$ with which the stationary distribution of three consecutive variables is given by

$$\begin{aligned} &\bar{p}\big\{x(t)x(t-1)x(t-2)\big\} \\ &\quad = p\big\{x(t)\big|x(t-1)x(t-2)\big\}\bar{p}\big\{x(t-1)x(t-2)\big\}. \end{aligned} \tag{11.159}$$

A Markov chain is determined by the stationary distribution

$$\bar{p}\big\{x(t)x(t-1)x(t-2)\big\}, \tag{11.160}$$

from which we can recover the stationary distribution $\bar{p}\{x(t)x(t-1)\}$ and the transition probability $p\{x(t)|x(t-1)x(t-2)\}$.

The manifold of the second-order Markov chain is a four-dimensional exponential family. Its $\boldsymbol{\eta}$-coordinates are

$$\eta_1 = E\big[x(t)\big], \tag{11.161}$$

$$\eta_{11} = E\big[x(t)x(t-1)\big], \tag{11.162}$$

$$\eta_{1\cdot1} = E\big[x(t)x(t-2)\big], \tag{11.163}$$

$$\eta_{111} = E\big[x(t)x(t-1)x(t-2)\big]. \tag{11.164}$$

The first one, η_1, is the firing rate, and $(\eta_{11}, \eta_{1\cdot1})$ are joint firing rates of two time bins, one neighboring and the other separated by one bin, and η_{111} is the joint firing rate of three consecutive time bins. By expanding $\log p(\boldsymbol{X}^T)$, we have the corresponding $\boldsymbol{\theta}$-coordinates $(\theta_1; \theta_{11}, \theta_{1\cdot1}, \theta_{111})$. Here, $(\theta_{11}, \theta_{1\cdot1}, \theta_{111})$ are orthogonal to the firing rate η_1, and θ_{111} represents the intrinsic triplewise interaction orthogonal to the firing rate η_1 and pairwise correlations η_{11} and $\eta_{1\cdot1}$.

11.6.3 Estimation of Shape Parameter in Renewal Process

The gamma distribution gives a good approximation to the interspike interval (ISI) of a real cortical neuron. It is believed that the value κ of the shape parameter is fixed for a neuron depending on its type, although the firing rate changes over time

(Shinomoto et al. 2003). It is important to estimate κ robustly from experimental data of spike trains. Let $\{\boldsymbol{T}\} = \{T_1, \ldots, T_m\}$ be m ISIs observed from a neuron when $m+1$ spikes are observed. Then, the likelihood of such data is given by

$$p(\boldsymbol{T}; \lambda, \kappa) = \prod_{i=1}^{m} f(T_i, \lambda, \kappa), \tag{11.165}$$

where $f(T, \lambda, \kappa)$ is the density of the gamma distribution with fixed unknown parameters λ and κ. We can calculate the maximum likelihood estimators $\hat{\lambda}$, $\hat{\kappa}$ that maximize the likelihood. This is a consistent and Fisher-efficient estimator of the pair (λ, κ) when m is large.

However, the firing rate $\lambda(t)$ changes over time in the real situation while κ is fixed. See Miura et al. (2006) for this problem. The number of the unknown parameters is $T+1$ in this case, because $\lambda(1), \ldots, \lambda(T)$ are additional unknown parameters. Hence, it is difficult to estimate κ. Such a problem is known as a semiparametric estimation since function $\lambda(t)$ is unknown. The maximum likelihood estimator is not necessarily consistent, because this is known as one of the Neymann–Scott problems.

The method of estimating function (Godambe 1991) was developed for the purpose of obtaining a semiparametric estimator, and information geometry plays a fundamental role for elucidating the structure of estimating functions (Amari and Kawanabe 1997). Let us consider a function $y(T, \kappa)$ of T depending on κ but not on λ. When it satisfies, for any λ,

$$E\big[y(T, \kappa)\big] = 0, \tag{11.166}$$

where expectation is taken with respect to $p(T, \kappa, \lambda)$, the function y is said to be an unbiased estimating function. When such a function exists, we have the estimating equation

$$\sum_{i=1}^{m} y(T_i, \kappa) = 0. \tag{11.167}$$

Its solution $\hat{\kappa}$ is a consistent estimator of κ, in whatever way $\lambda(t)$ changes, because the above sum divided by m, $\sum y(T_i, \kappa)/m$, converges to 0 as m increases when κ is the true value.

The best unbiased estimating function is the projection of the κ-score function to the information subspace in the fiber bundle of the manifold of semiparametric probability distributions (Amari and Kawanabe 1997).

However, we can prove unfortunately that such an estimation function does not exist in the present case. We modify the situation that λ may change at least after two consecutive ISIs such as T_t and T_{t+1} (Miura et al. 2006). Hence, we divide the whole time into N subintervals and assume that λ has a fixed value λ_l for the lth interval, $l = 1, \ldots, N$, where each interval includes $k+1$ spikes and hence k observations $T_1^{(l)}, \ldots, T_k^{(l)}$ $(k \geq 2)$. Then, an unbiased estimating function exists

for k observations $T_1, \ldots, T_k$, where $k \geq 2$. It is given by

$$y(T_1, \ldots, T_k; \kappa) = \sum \log T_i - k \log\Big(\sum T_i\Big) \tag{11.168}$$

$$-k\big\{\phi(k\kappa) - \phi(\kappa)\big\}, \tag{11.169}$$

where

$$\phi(\kappa) = \frac{\Gamma'(\kappa)}{\Gamma(\kappa)}. \tag{11.170}$$

Hence, the estimating equation is

$$\sum_{l=1}^{N} y\big(T_1^{(l)}, \ldots, T_k^{(l)}; \kappa\big) = 0. \tag{11.171}$$

This gives a consistent estimator which is Fisher efficient and is different from the maximum likelihood estimator, which is not consistent.

A number of measures of irregularity for spike trains have been proposed so far. The coefficient of variation

$$c_V = \frac{\hat{\sigma}_T}{\bar{T}}, \tag{11.172}$$

which is the ratio of the standard deviation $\hat{\sigma}_T$ of $\{T_1, \ldots, T_m\}$ divided by the mean $\bar{T}$ is one of them and is different from 1 when the process is not Poissonian. In the case of the gamma distribution, we have

$$\kappa = \frac{1}{{c_V}^2}. \tag{11.173}$$

Hence, we have an estimator of κ based on (11.172). However, this is biased and worse than our estimator, which is optimal. The Fano factor is another one. Shinomoto et al. (2003) proposed

$$L_V = \frac{1}{m-1} \sum_{i=1}^{m-1} \frac{3(T_i - T_{i+1})}{(T_i + T_{i+1})^2}, \tag{11.174}$$

which works well for many experimental data.

Miura et al. (2006) proposed

$$S_I = -\frac{1}{m-1} \sum_{i=1}^{m-1} \frac{1}{2} \log\left(\frac{4T_i T_{i+1}}{(T_i + T_{i+1})^2}\right), \tag{11.175}$$

which is derived from the theory of estimating function (Amari and Kawanabe 1997). Indeed, the consistent estimator $\hat{\kappa}$ of the gamma distribution is a function of S_I. The relation between S_I and L_V was also shown (Miura et al. 2006), and the difference lies in the arithmetic mean and geometric mean of the inverses of T_i.

11.6.4 Discrimination of ISI Distributions

Let $f_1(T)$ and $f_2(T)$ be two ISI distributions of renewal processes. There are many divergence measures between the two distributions, such as the Kullback–Leibler divergence, Hellinger distance, and the α-divergence (Amari and Nagaoka 2000). Kang and Amari (2008) used the Chernoff divergence

$$D[f_1, f_2] = -\max_{\alpha} \log \int f_1(T)^{\alpha} f_2(T)^{1-\alpha}\, dT. \tag{11.176}$$

This is related to the α-divergence of two distributions

$$D_{\alpha} = \alpha(1-\alpha)\left[1 - \int f_1(T)^{\alpha} f_2(T)^{1-\alpha}\, dT\right] \tag{11.177}$$

and is the KL-divergence as $\alpha \to 1$. The Chernoff divergence is directly related to the error probability of discrimination of the two distributions from data including N observations,

$$p_{\text{error}} \approx \exp\{-N D[f_1, f_2]\}. \tag{11.178}$$

See Cover and Thomas (1991).

Let k^* be the solution of

$$\int \exp(k^* t) f_1(t)_2^{\alpha} f(t)^{1-\alpha}\, dt = 1. \tag{11.179}$$

Then, for large observation time T, the asymptotic form is obtained as

$$D_{\alpha} \approx k^* T. \tag{11.180}$$

This shows that the probability of discrimination error is

$$p_{\text{error}} \approx \exp(-k^* T). \tag{11.181}$$

When f_1 and f_2 are close to each other, we have the Fisher information for discrimination, which is also used for evaluating the estimation error. The Fisher information of various models, such as the gamma distribution, inverse Gaussian distribution, and leaky integration-and-fire distribution, were studied by Kang and Amari (2008). It was shown that the gamma distribution has the minimum Fisher information for estimating the firing rate.

11.7 Conclusions

The present chapter is devoted to an introduction to information-geometrical approach to stochastic spike trains. We began with the full model of probability distributions of multiple spikes, in order to understand its mathematical structure such as orthogonality and flatness. We then studied properties of tractable models such as the homogeneous model, marginal model, and mixture model. We also studied the

structure of temporal correlations and related tractable models from the mathematical point of view. Spatio-temporal models remain still to be a subject of research in future, and we need to study underlying stochastic dynamics of neurons in addition to stochastic approach.

References

Ackley DH, Hinton GE, Sejnowski TJ (1985) A learning algorithm for Boltzmann machines. Cogn Sci 9:147–169

Amari S (2001) Information geometry on hierarchy of probability distributions. IEEE Trans Inform Theory 47:1701–1711

Amari S (2009a) Measure of correlation orthogonal to change in firing rate. Neural Comput 21:960–972

Amari S (2009b) α-Divergence is unique, belonging to both f-divergence and Bregman divergence class. IEEE Transactions on Information Theory 55:November

Amari S (2010) Conditional mixture model for correlated neuronal spikes. Neural Comput 22(7):1718–1736

Amari S, Kawanabe M (1997) Information geometry of estimating functions in semi parametric statistical models. Bernoulli 3:29–54

Amari S, Kurata K, Nagaoka H (1992) Information geometry of Boltzmann machines. IEEE Trans Neural Networks 3:260–271

Amari S, Nagaoka H (2000) Methods of information geometry. Translations of mathematical monographs, vol 191. AMS & Oxford University Press, Providence

Amari S, Nakahara H, Wu S, Sakai Y (2003) Synchronous firing and higher-order interactions in neuron pool. Neural Comput 15:127–142

Bohte SM, Spekreijse H, Roelfsema PR (2000) The effect of pair-wise and higher-order correlations on the firing rate of a postsynaptic neuron. Neural Comput 12:153–159

Brette R (2009) Generation of correlated spike trains. Neural Comput 21:188–215

Chentsov NN (1972) Statistical decision rules and optimal inference. American Mathematical Society, Providence 1982. Originally published in Russian, Nauka, Moscow

Cover TM, Thomas JA (1991) Elements of information theory. Wiley, New York

Dayan P, Abbott LF (2005) Theoretical neuroscience: computational and mathematical modeling of neural systems. MIT Press, Cambridge

Feng J, Brown D (2000) Impact of correlated inputs on the output of the integrate-and-fire model. Neural Comput 12:671–692

Godambe VP (1991) Estimating functions. Oxford University Press, London

Kang K, Amari S (2008) Discrimination with spike times and ISI distributions. Neural Comput 20:1411–1426

Kass RE, Ventura V, Brown EN (2005) Statistical issue in the analysis of neuronal data. J Neurophysiol 94:8–25

Kuhn A, Rotter S, Aertsen A (2002) Correlated input spike trains and their effects on the leaky integrate-and-fire neuron. Neurocomputing 44–46:121–126

Kuhn A, Aertsen A, Rotter S (2003) Higher-order statistics of input ensembles and the response of simple model neurons. Neural Comput 15:67–101

Miura K, Okada M, Amari S (2006) Estimating spiking irregularities under changing environments. Neural Comput 18:2359–2386

Montani F, Ince RAA, Senatore R, Arabzadeh E, Diamond M, Panzeri S (2009) The impact of higher-order interactions on the rate of synchronous discharge and information transmission in somatosensory cortex. Philosophical Trans R Soc A 367:3297–3310

Nakahara H, Amari S (2002) Information-geometric measure for neural spikes. Neural Comput 14:2269–2316

Nakahara H, Amari S, Richmond B (2006) A comparison of descriptive models of a single spike train by information-geometric measure. Neural Comput 18:545–568

Niebur E (2007) Generation of synthetic spike trains with defined pairwise correlations. Neural Comput 19:1720–1738

Salinas E, Sejnowski TJ (2001) Correlated neuronal activity and the flow of neural information. Nature Rev Neurosci 2:539–550

Simazaki H, Gruen S, Amari S (2010) Analysis of subsets of higher-order correlated neurons based on marginal correlation coordinates. Cosyne 10

Shinomoto S, Shima K, Tanji J (2003) Differences in spiking patterns among cortical neurons. Neural Comput 15:2823–2842

Chapter 12
Higher-Order Correlations and Cumulants

Benjamin Staude, Sonja Grün, and Stefan Rotter

Abstract Recent advances in electrophysiological and imaging techniques have highlighted the need for correlation measures that go beyond simple pairwise analyses. In this chapter, we discuss cumulant correlations as natural and intuitive higher-order generalizations of the covariance. In particular, we show how cumulant correlations fit to a frequently used additive model of correlation, an idea that mimics correlations among spike trains that originate from overlapping input pools. Finally, we compare the cumulant correlations to the interaction parameters of the well-known exponential family by computing the respective parameters for two different models. We find that the different frameworks measure entirely different aspects, so that populations can have higher-order correlations in one framework but none in the other.

12.1 Introduction

It has long been suggested that fundamental insight into the nature of neuronal computation requires the understanding of the cooperative dynamics of populations of neurons (Hebb 1949; Gerstein et al. 1989; and Chaps. 5–11 in this book). Indeed, the increasing experimental evidence for computationally relevant neuronal interactions on various temporal and spatial scales supports the concept of cooperative computation (see, e.g., Salinas and Sejnowski 2001; Lestienne 2001; Womelsdorf and Fries 2007; Kohn et al. 2009, and references therein). Direct experimental evidence for coordinated activity on the level of spike trains, however, relies almost exclusively on the correlations between pairs of nerve cells (Gray and Singer 1989; Vaadia et al. 1995; Riehle et al. 1997; Bair et al. 2001; Kohn and Smith 2005;

S. Rotter (✉)
Bernstein Center Freiburg & Faculty of Biology, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany
e-mail: stefan.rotter@biologie.uni-freiburg.de
url: http://www.bcf.uni-freiburg.de/people/details/rotter

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_12, © Springer Science+Business Media, LLC 2010

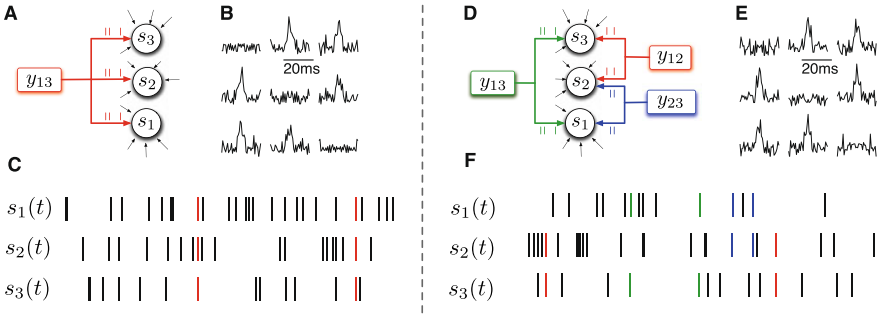

Fig. 12.1 Origin and signature of higher-order correlations. Connection diagrams (**A**, **D**), cross- and autocorrelation functions (**B**, **E**), and raster plots (**C**, **F**) of two hypothetical populations of $N = 3$ neurons (s_1, s_2, and s_3). Additional to independent background (*small black arrows* in **A** and **D**), the three neurons of the *left panel* share one common pool y_{123} (**A**), which induces coordinated spikes in all three spike trains (*red ticks* in **C**). In the right population, only pairs of neurons have overlapping input pools (y_{12}, y_{13}, and y_{23} in **D**). As a consequence, coordinated spikes occur only in pairs of spike trains $s_i(t)$ (*colored ticks* in **F**). The relative strength of the common components in the two populations is adjusted so that cross- and autocorrelation functions are identical (**B**, **E**; estimated from $T = 100$ seconds, central peaks of autocorrelations removed)

Fujisawa et al. 2008). Such pairwise correlations, however, do not resolve the cooperative dynamics of neuronal populations (Martignon et al. 1995; Bohte et al. 2000; Kuhn et al. 2003). To illustrate the limitations of pairwise analyses, consider two hypothetical input scenarios into $N = 3$ neurons. In the first scenario, schematized in the left panel of Fig. 12.1, the three neurons s_1, s_2, and s_3 share one common input pool y_{123} (Fig. 12.1A), additional to independent inputs (black arrows in Fig. 12.1A). In such a scenario, spike events in the common pool y_{123} arrive at the three neurons synchronously and, assuming sufficiently strong synapses, induce coincident events in all three output spike trains $s_1(t)$, $s_2(t)$, and $s_3(t)$ (red marks in Fig. 12.1C). In a different scenario (right panel of Fig. 12.1), the inputs to pairs of neurons overlap without a common input pool to all three neurons. As a consequence, pairs of spike trains do have coincidences (colored spikes in Fig. 12.1F), but this input configuration does not directly generate triplet coincidences. Importantly, the relative strengths of the common components and the independent inputs can be such that firing rates of individual neurons and pairwise correlations in both scenarios are identical, as illustrated by the identical cross- and autocorrelation functions (Fig. 12.1B and E). Thus, the apparent differences between these two populations are not captured by mere pairwise correlations. It is the higher-order correlations that determine whether coincident spikes of two neurons are also coincident with the spikes of the third neuron (see also Chap. 11). The population in the left panel of Fig. 12.1 has positive triplet correlations, while the right population has only pairwise correlations but no correlations of order three.

This contribution presents a mathematical framework of higher-order cumulant correlations that is compatible with the "common component" interpretation presented in Fig. 12.1. We begin by reviewing the covariance as a measure of pairwise correlations and show how it can be interpreted in terms of a common

component. Section 12.2.2 introduces cumulant-based higher-order correlations as the generalization of the covariance to more than two processes. The mathematical description of the common component model for $N > 2$ neurons and its relationship to higher-order cumulant correlations is presented in Sect. 12.2.3. To interpret cumulant correlations for spike trains, we introduce a parametric model of correlated Poisson processes in Sect. 12.2.3 (Holgate 1964; Kawamura 1979; Karlis and Meligkotsidou 2005; Johnson and Goodman 2007; Ehm et al. 2007; Staude et al. 2009, 2010), where we also present tractable models with only few parameters. In Sect. 12.4, we review two recent analysis techniques (Ehm et al. 2007; Staude et al. 2009, 2010) that exploit the common source model. Importantly, the cumulant correlations presented here are not identical to the higher-order parameters of the binary exponential family (see Chap. 11 and, e.g., Martignon et al. 1995; Shlens et al. 2006; Nakahara and Amari 2002; Schneidman et al. 2006; Montani et al. 2009; Shimazaki et al. 2009). We illustrate their difference by computing the respective parameters for two representative examples in Sect. 12.5.

12.2 Correlation

The first step in most correlation analysis procedures is discretization, or binning. Here, the original continuous-time spike trains $s_i(t) = \sum_j \delta(t - t_j^i)$ with spike times $\{t_j^i\} = \{t_1^i, t_2^i, \ldots\}$ are converted into a sequence of integer counting values $S_i(l)$, indexed by the bin number l. The value of S_i at l is the spike count, i.e., the number of spikes the ith neuron emitted in the lth bin of length h,

$$S_i(l) = \int_{lh}^{(l+1)h} s_i(t)\, dt. \tag{12.1}$$

In most correlation studies, the bin size h is chosen in the range of a few milliseconds such that at most one spike falls in each bin and the $S_i(l)$ are binary, spike/no spike variables (see, e.g., Chaps. 10, 11, and 13). In this contribution, we allow also larger bins, so that $S_i(l)$ can have arbitrary integer values. If not stated otherwise, we will assume stationarity in the sense that the statistical properties of $S_i(l)$ do not change with time. We thus regard subsequent values of $S_i(l)$ as i.i.d. random variables and drop the time argument l in the remainder of this study.

We wish to stress that although the focus of this contribution is on spike train analysis, the cumulant correlations presented here are well defined for general real-valued random variables, like, e.g., membrane potentials. The recurrence to spike trains will be made when introducing correlated Poisson processes in Sect. 12.3.

12.2.1 Pairwise Correlation

A common measure for the dependence between two real-valued variables S_1 and S_2 is the covariance

$$\mathrm{Cov}[S_1, S_2] := \mathrm{E}[S_1 S_2] - \mathrm{E}[S_1]\mathrm{E}[S_2], \tag{12.2}$$

or normalized versions thereof. The standard normalization divides the covariance by the geometric mean of the individual variances. The resulting Pearson's correlation coefficient

$$c_{12} := \frac{\mathrm{Cov}[S_1, S_2]}{\sqrt{\mathrm{Var}[S_1]\,\mathrm{Var}[S_2]}} \tag{12.3}$$

is a dimensionless quantity with values between -1 and 1, where $c_{12} = 1$ implies identity ($S_1 = S_2$), and $c_{12} = -1$ implies negatively correlated variables (see, e.g., Chap. 11 for a different normalization).

In terms of spike trains, the standard interpretation assumes that the S_i are binary spike/no spike variables. In this case, the expectation $\mathrm{E}[S_i]$ equals the probability for a spike, and thus $\mathrm{E}[S_1]\mathrm{E}[S_2]$ is the probability for a spike coincidence, or synchrony, under the assumption of independent firing. Furthermore, $S_1(l)S_2(l) = 1$ if and only if both spike trains have a spike in the lth bin, and hence $\mathrm{E}[S_1 S_2]$ is the actual probability for a spike coincidence. Thus, the covariance subtracts the probability of a "chance coincidence" from the actual probability of a coincidence. The result has been termed the probability for "excess synchrony" (see Chaps. 10 and 17).

An alternative interpretation that does not require binary variables (or even spike trains) assumes that the dependence between S_1 and S_2 originates from a common additive component (e.g., overlapping input pools of neurons, compare Fig. 12.1). In this framework, S_1 and S_2 can be expressed with the help of three independent random variables Y_1, Y_2, and Y_{12} as

$$\begin{aligned} S_1 &= Y_1 + Y_{12}, \\ S_2 &= Y_2 + Y_{12}. \end{aligned} \tag{12.4}$$

Given the decomposition of (12.4), the linearity of the covariance implies

$$\begin{aligned} \mathrm{Cov}[S_1, S_2] &= \mathrm{Cov}[Y_1 + Y_{12}, Y_2 + Y_{12}] \\ &= \underbrace{\mathrm{Cov}[Y_1, Y_2]}_{=0} + \underbrace{\mathrm{Cov}[Y_1, Y_{12}]}_{=0} + \underbrace{\mathrm{Cov}[Y_{12}, Y_2]}_{=0} + \underbrace{\mathrm{Cov}[Y_{12}, Y_{12}]}_{=\mathrm{Var}[Y_{12}]} \\ &= \mathrm{Var}[Y_{12}]. \end{aligned} \tag{12.5}$$

Thus, correlation as measured by the covariance captures the strength (the variance) of the common component Y_{12}. This simple interpretation of correlation has made the additive construction of (12.4) a very common tool to model and interpret correlated stochastic signals, both in neuroscience (De la Rocha et al. 2007; Staude et al. 2008; Tetzlaff et al. 2008) and other research areas. In the next section, we introduce cumulant correlations κ as natural higher-order generalizations of the covariance. These higher-order correlations respect in particular the common component interpretation of correlation, as we will show in Sect. 12.2.3.

12.2.2 Higher-Order Correlations ($N > 2$)

The straightforward generalization of the covariance as a measure for dependence among $N = 3$ variables subtracts the product of the expectation values from the expectation value of the product (compare Stratonovich 1967)

$$\mathrm{E}[S_1 S_2 S_3] - \mathrm{E}[S_1]\mathrm{E}[S_2]\mathrm{E}[S_3]. \tag{12.6}$$

If the S_i are binary, this expression computes the probability to observe a coincident spike in all three neurons (first term) and subtracts from it the prediction that assumes complete independence (second term). Thus, (12.6) vanishes if all three neurons are independent. However, the dependence structure between three neurons can be more complicated. If, for instance, S_1 and S_2 are correlated, but both are independent of S_3, we have

$$\mathrm{E}[S_1 S_2 S_3] \overset{S_3 \text{ independent}}{=} \mathrm{E}[S_1 S_2]\mathrm{E}[S_3] \overset{(S_1,S_2) \text{ correlated}}{\neq} \mathrm{E}[S_1]\mathrm{E}[S_2]\mathrm{E}[S_3].$$

In this case, expression (12.6) will not be zero even though the dependence in the population is only pairwise, without any triplet correlations. To quantify triplet correlations, we have to subtract from $\mathrm{E}[S_1 S_2 S_3]$ not only the prediction of complete independence, but also the predictions that assume nonzero covariances between neuron pairs and independence of the third neuron, i.e., all terms of the form $\mathrm{Cov}[S_i, S_j]\mathrm{E}[S_k] = \mathrm{E}[S_i S_j]\mathrm{E}[S_k] - \mathrm{E}[S_i]\mathrm{E}[S_j]\mathrm{E}[S_k]$ $(i, j, k = 1, 2, 3)$. Putting everything together, we obtain the expression

$$\begin{aligned}
\kappa_{1,1,1}[S_1, S_2, S_3] &:= \mathrm{E}[S_1 S_2 S_3] - \mathrm{E}[S_1]\mathrm{E}[S_2]\mathrm{E}[S_3] - \mathrm{Cov}[S_1, S_2]\mathrm{E}[S_3] \\
&\quad - \mathrm{Cov}[S_1, S_3]\mathrm{E}[S_2] - \mathrm{Cov}[S_2, S_3]\mathrm{E}[S_1] \\
&= \mathrm{E}[S_1 S_2 S_3] + 2\mathrm{E}[S_1]\mathrm{E}[S_2]\mathrm{E}[S_3] \\
&\quad - \mathrm{E}[S_1 S_2]\mathrm{E}[S_3] - \mathrm{E}[S_1 S_3]\mathrm{E}[S_2] - \mathrm{E}[S_2 S_3]\mathrm{E}[S_1],
\end{aligned} \tag{12.7}$$

where the second equality results from inserting the definition of the covariance.

The quantity $\kappa_{1,1,1}[S_1, S_2, S_3]$ is the third *connected cumulant*. It measures the dependence in the triplet (S_1, S_2, S_3) that is not already contained in the pairwise correlations. Cumulants κ arise in various fields of theoretical and applied statistics as variants of the more familiar (*raw*) *moments* $\mu_k[S] := \mathrm{E}[S^k]$ (Stuart and Ord 1987; Brillinger 1996; Bell and Sejnowski 1996; Gardiner 2003; Mattner 2004; Blaschke and Wiskott 2004; Zhou et al. 2006; Di Nardo et al. 2008; and Chap. 1).

While the above "corrective" construction has its intuitive advantages, defining kth-order correlations by subtracting appropriate corrections from $\mathrm{E}[X_1 \cdots X_k]$ is exceedingly complicated for large k. Therefore, the standard construction of higher-order connected cumulants follows an alternative approach. Recall that the expectation operator is additive-linear, i.e., $\mathrm{E}[\sum_{i=1}^N S_i] = \sum_{i=1}^N \mathrm{E}[S_i]$. The variance, however, is only additive-linear if the individual variables S_i are pairwise uncorrelated. If not, we have the well-known variance–covariance relationship

$$\mathrm{Var}\left[\sum_{i=1}^N S_i\right] = \sum_{i=1}^N \mathrm{Var}[S_i] + \sum_{i \neq j} \mathrm{Cov}[S_i, S_j].$$

Thus, alternative to the "corrective" interpretation of (12.2), the covariance can also be regarded as a measure for the degree to which pairwise dependencies in a population compromise the linearity of the variance of their sum. This is the basis of the standard definition of higher-order connected cumulants (e.g., Gardiner 2003).

The formal construction in the multivariate case is quite involved, however, and the details are therefore postponed to Appendix A. The basic ingredient is the logarithm of the Fourier transform $\mathcal{F}[f_S]$ of the distribution function, the so-called log-characteristic function $\psi_S(u) = \log \mathrm{E}[e^{iuS}] = \log \int e^{ius} f_S(s)\, ds = \log \mathcal{F}[f_S](u)$. The (univariate) cumulants $\kappa_k[S]$ are the coefficients of the Taylor-series expansion of $\psi_S(u)$,

$$\psi_S(u) = \sum_{k=1}^{\infty} i^k \frac{u^k}{k!} \kappa_k[S],$$

provided that the series converges and all coefficients are finite.

The rationale behind this construction are the facts that the distribution of a sum of *independent* variables is the convolution of the individual distributions and that the Fourier transform maps convolutions to products. The log-characteristic function of the sum of an independent family $\{S_i\}$ therefore fulfills

$$\begin{aligned}\psi_{S_1+\cdots+S_N} &= \log \mathcal{F}[f_{S_1+\cdots+S_N}] \\ &= \log\bigl(\mathcal{F}[f_{S_1} * \cdots * f_{S_N}]\bigr) \\ &= \log\bigl(\mathcal{F}[f_{S_1}] \cdots \mathcal{F}[f_{S_N}]\bigr) \\ &= \psi_{S_1} + \cdots + \psi_{S_N}.\end{aligned}$$

As this equality has to hold in all powers of the Taylor-series expansion, the cumulants of an independent population satisfy

$$\kappa_k[S_1 + \cdots + S_N] = \kappa_k[S_1] + \cdots + \kappa_k[S_N] \quad \text{for } k = 1, 2, \ldots.$$

If, however, the S_i are not independent, then $\psi_{S_1+\cdots+S_N} \neq \psi_{S_1} + \cdots + \psi_{S_N}$. As a consequence, also $\kappa_k[S_1 + \cdots + S_N] \neq \kappa_k[S_1] + \cdots + \kappa_k[S_N]$, and the connected cumulants measure the degree to which equality is violated. For $k = 3$, for instance, we have

$$\begin{aligned}\kappa_3[S_1 + \cdots + S_N] = \sum_{i=1}^{N} \kappa_3[S_i] + \sum_{i \neq j} \mathrm{Cov}\bigl[S_i^2, S_j\bigr] \\ + \sum_{i \neq j,\, j \neq k,\, i \neq k} \kappa_{1,1,1}[S_i, S_j, S_k].\end{aligned}$$

Thus, $\kappa_{1,1,1}$ measures the extent to which triplet correlations influence the linearity of the third cumulant.

The above paragraphs provide only a very abstract interpretation of cumulant correlations. For a more concrete intuition, the next section generalizes the common component interpretation of the covariance (12.5) to higher-order connected cumulants. Before, however, we will fix some notation and discuss the central "interaction property" of connected cumulants.

Definition 1 Let $\mathbf{S} = (S_1, \ldots, S_N)$ be an N-dimensional random variable, e.g., the spike counts of N parallel spike trains, let $M = \{m_1, \ldots, m_k\}$ be a subset of $\{1, \ldots, N\}$ with $k = |M|$ elements, and denote by $\boldsymbol{\sigma}(M) \in \{0, 1\}^N$ the binary indicator vector of the set M whose ith component is 1 if $i \in M$ and 0 otherwise.

Then we measure kth-order correlations among $\{S_i\}_{i\in M}$ by the connected cumulant $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$. We say that $\mathbf{S}$ has correlations of order k if and only if at least one kth-order connected cumulant of $\mathbf{S}$ is nonzero.

As discussed above, cumulants can be expressed in terms of sums of (mixed) moments of the S_i. For instance, $\kappa_{\boldsymbol{\sigma}(\{1\})}[\mathbf{S}] = \kappa_{1,0,0,\ldots,0}[\mathbf{S}] = \mathrm{E}[S_1]$ and $\kappa_{\boldsymbol{\sigma}(\{1,2\})}[\mathbf{S}] = \kappa_{1,1,0,\ldots,0}[\mathbf{S}] = \mathrm{Cov}[S_1, S_2] = \mathrm{E}[S_1S_2] - \mathrm{E}[S_1]\mathrm{E}[S_2]$. Furthermore, $\kappa_{2,0,\ldots,0}[\mathbf{S}] = \kappa_2[S_1] = \mathrm{Var}[S_1] = \mathrm{E}[S_1^2] - \mathrm{E}[S_1]^2$ and $\kappa_{3,0,\ldots,0}[\mathbf{S}] = \kappa_3[S_1] = \mathrm{E}[(S_1 - \mathrm{E}[S_1])^3]$. Similar expressions relating higher cumulants to moments exist but are increasingly complicated for higher orders (Stuart and Ord 1987).

The theoretical foundation for cumulants as measures for higher-order correlations is their "interaction property" (see Appendix A, Theorem 2, and Streitberg 1990). In the simplest case, where the order of measured correlation equals the size of the population, i.e., $k = N$, it states that a population whose distribution factorizes into independent components does not have Nth-order correlations. Even though this property appears to be a fundamental requirement, not all candidate measures for higher-order correlation fulfill it. The Sarmanov–Lancaster parameters (Lancaster 1958; Lancaster and Adams 1986; Bahadur 1961; Sarmanov 1962) that recently found their way into neuroscience (Johnson and Goodman 2008; Roudi et al. 2009), for instance, do not fulfill this requirement for orders > 3 (Streitberg 1990). Note that the reverse of the interaction property does not hold in general, i.e., a distribution that does not contain higher-order correlations does not necessarily factorize (e.g., the population in the right panel of Fig. 12.1).

12.2.3 The Additive Common Component Model

Besides the "corrective" and "linearity" interpretations presented in the previous section, higher-order cumulant correlations also inherit the common component interpretation from the covariance. The underlying additive model of (12.4) in the case $N > 2$ assigns independent "component variables" Y_M to all subgroups $M \subset \{1, \ldots, N\}$. The ith individual variable S_i is then defined as the sum of those Y_M with $i \in M$,

$$S_i = \sum_{M \ni i} Y_M \quad (i = 1, \ldots, N). \tag{12.8}$$

Before we proceed, we wish to stress that the Y_M are typically not observable in experimental situations but are targets of statistical analysis. In Fig. 12.1, for instance, we regarded the y_M as inputs to neurons. Estimating higher-order correlations can then be interpreted as the estimation of overlaps in input pools from output spike counts S_i.

Now recall that for $N = 2$, the common component of S_1 and S_2 was Y_{12} (12.4). For $N > 2$ and $M \subset \{1, \ldots, N\}$, the common component of the subgroup $\{S_i\}_{i\in M}$ is the superposition of all those component variables Y_B that affect all members of

the group $\{S_i\}_{i\in M}$. Given (12.8), these are the Y_B whose index sets B contain M. Thus, the total common component of the group $\{S_i\}_{i\in M}$ is the sum $\sum_{B\supset M} Y_B$. The following theorem states that the connected cumulants of Definition 1 respect this additive construction (see Sect. 12.7.3 for a proof).

Theorem 1 *Let* $\mathbf{S}$ *be an* N*-dimensional random variable that obeys the representation of* (12.8)*. Let* $M \subset \{1,\ldots,N\}$ *be a subset with* $k = |M|$ *elements. Then the* k*th-order correlation of the set* $\{S_i\}_{i\in M}$*, i.e. the connected cumulant* $\kappa_{\sigma(M)}[\mathbf{S}]$*, is the sum of the* k*th cumulants of all* Y_B *with* $M \subset B$*,*

$$\kappa_{\sigma(M)}[\mathbf{S}] = \sum_{B\supset M} \kappa_k[Y_B]. \tag{12.9}$$

Theorem 1 requires a few remarks. First of all, it shows that connected cumulants directly estimate the strength of common components in observed signals, like, e.g., overlaps in input pools of neurons (however, see Tetzlaff et al. 2008). In particular, the kth-order correlation of the subgroup $\{S_i\}_{i\in M}$ is the strength of its common component, as measured by the sum of the cumulants $\sum_{B\supset M} \kappa_k[Y_B]$. Note that (12.5) is a special case of the above theorem with $N = 2$ and the identities $\kappa_{1,1}[\mathbf{S}] = \mathrm{Cov}[S_1, S_2]$ and $\kappa_2[Y_{12}] = \mathrm{Var}[Y_{12}]$.

Second, if the higher cumulants of the Y_B vanish, then also $\kappa_{\sigma(M)}[\mathbf{S}] = 0$, even though the variables $\{S_i\}_{i\in M}$ do have common components. This holds, for instance, if the Y_B are Gaussian variables, because $\kappa_k[Y_B] = 0$ for $k > 2$ for Gaussian Y_B. As a consequence, the additive *Gaussian* model does not have higher-order correlations. However, the Gaussian distribution is the only distribution with vanishing cumulants, and any other distribution has $\kappa_k[Y_M] \neq 0$ for all $k > 2$.

Third, for pairwise correlations, i.e., $|M| = 2$, Theorem 1 implies that $\kappa_{\sigma(M)}[\mathbf{S}] = \sum_{M\subset B} \mathrm{Var}[Y_B] \geq 0$, because the variance is nonnegative. The additive model can thus only model positive pairwise correlations. Note that this is not a limitation of cumulant correlations, but of the additive model only.

Fourth, a component variable Y_M generates correlations among all S_i with $i \in M$. As a consequence, correlations of order k in the additive model automatically generate correlations of orders $< k$ (Johnson and Goodman 2007).

Fifth, the construction of (12.8) imposes constraints on the variables S_i. As the S_i are sums of the independent random variables Y_M, the distribution of S_i is the convolution of the distributions of the Y_M. In order to control the statistical properties of the S_i, we thus have to know how convolutions of distributions of the Y_M look like. For practical purposes, one typically assumes that the Y_M are either Gaussian (De la Rocha et al. 2007; Shea-Brown et al. 2008; Tetzlaff et al. 2008) or Poisson (see next section and, e.g., Holgate 1964; Kawamura 1979; Karlis and Meligkotsidou 2005; Johnson and Goodman 2007; Ehm et al. 2007; Brette 2009; Staude et al. 2009, 2010), as in this case also the S_i are Gaussian or Poisson, respectively.

Finally, if $Y_B = 0$ for all groups B above a given size $|B| > \xi_0$, Theorem 1 implies that $\kappa_{\sigma(M)}[\mathbf{S}] = 0$ for all $|M| > \xi_0$. Thus, the presence of correlations beyond a given order ξ_0 requires the presence of a common component of more than ξ_0 of

the S_i. This is precisely the intuition of higher-order correlations which was used in Fig. 12.1.

We wish to emphasize that although the additive model provides a strong intuition, the alternative interpretations presented in Sect. 12.2.2 render cumulant correlations meaningful even without assuming an underlying additive model (see Sect. 12.5).

12.3 Correlated Poisson Processes

In order to interpret cumulant correlations for parallel *spike trains* $s_i(t)$, we simply assume that their counting variables S_i can be decomposed according to (12.8). Then, the component variables Y_M of Sect. 12.2.3 can be interpreted as the counting variables of independent point processes $y_M(t)$, obtained by discretization with bin size h. For the individual spike trains $s_i(t)$, we then have (see Fig. 12.2A)

$$s_i(t) = \sum_{i \in M} y_M(t). \tag{12.10}$$

Assuming the $y_M(t)$ to be Poisson processes with "component rates" ν_M, the $s_i(t)$ are also Poisson processes. For the associated counting variables Y_M, the Poisson property implies that cumulants of all orders are identical to the mean of the distribution (Papoulis and Pillai 2002), i.e., $\kappa_k[Y_M] = \nu_M h$ for all subgroups $M \subset \{1, \ldots, N\}$ and orders $k \in \mathbb{N}$. Hence, Theorem 1 implies that

$$\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = \sum_{B \supset M} \nu_B h. \tag{12.11}$$

The left-hand side of (12.11) is a measure for the strength of the correlation among the counting variables $\{S_i\}_{i \in M}$. In the "corrective" interpretation with binary S_i, for instance, $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ should be the probability for "excess coincidences" of the entire group $\{S_i\}_{i \in M}$. The right-hand side of (12.11), on the other hand, sums the expected spike counts of all components $y_B(t)$ with $M \subset B$. As the spikes of those components occur with perfect temporal precision in all $\{s_i(t)\}_{i \in M}$, (12.11) states that the probability for an excess coincidence in a given group is the sum of the rates of the perfectly synchronized spikes.

Due to the perfect temporal precision of common spikes, spike trains generated in the above model have correlations at zero lag, i.e., the cross-correlation function $\mathrm{E}[s_1(t)s_2(t-\tau)]$ has a delta-peak at $\tau = 0$. To model broader and/or nonzero-lag correlations as in Fig. 12.1B and E, a common remedy is to jitter the spikes of the common components before superposition. The cross-correlation function of s_1 and s_2 is then determined by the details of the jitter. If, for instance, a jitter is added independently to every spike of y_M and for every spike train s_i, a uniform jitter of width J_c results in a triangular correlation function of base length $2J_c$ (Staude et al. 2008), while a Gaussian jitter of standard deviation σ yields a Gaussian correlation function of width $\sqrt{2}\sigma$ (Brette 2009). For the present study, however, it is crucial to keep in mind that correlations are described strictly on the basis of the

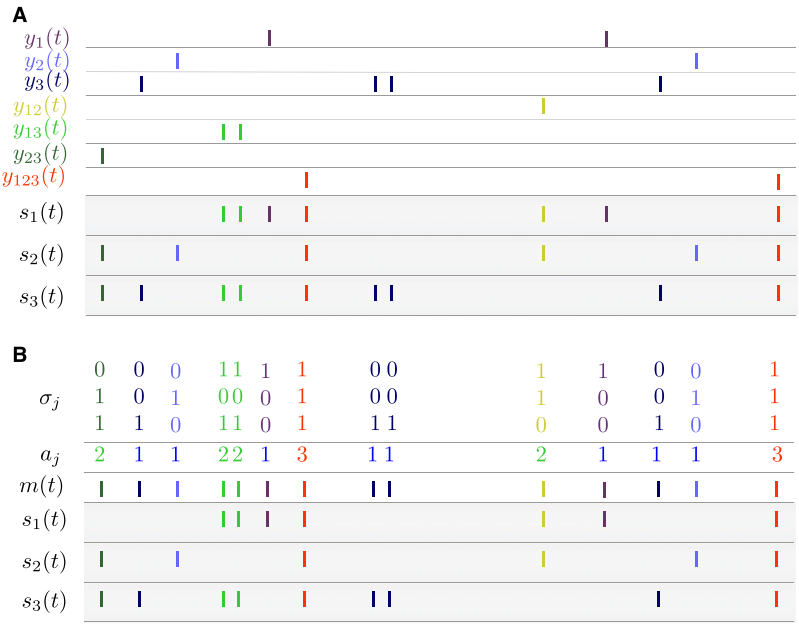

Fig. 12.2 Two parameterizations of correlated Poisson processes. In the component parameterization (**A**), individual spike trains $s_i(t)$ (*gray boxes*) are constructed as superposition of independent component processes $y_M(t)$ ($M \subset \{1,2,3\}$; compare (12.10)). Correlation is induced by components $y_M(t)$ with $|M| \geq 2$. In the marked point process parameterization (**B**), the carrier process $m(t)$ determines the spike-event times t_j irrespective of the neuron IDs. The amplitudes a_j determine the number of neurons, while the marks $\boldsymbol{\sigma}_j$ determine the precise neuron IDs that fire at time t_j. Here, correlation is induced by events with amplitudes $a_j \geq 2$. To model temporally imprecise correlation, jitters can be added to the common spikes, i.e., the events of $y_M(t)$ with $|M| \geq 2$ (**A**) or the events of $m(t)$ with $a_j \geq 2$ (**B**) (Staude et al. 2008; Brette 2009). Note that the different parameterizations generate identical raster displays in **A** and **B**

counting variables S_i, not via cross-correlation functions between the spike trains $s_i(t)$ in continuous time. Given a large enough bin size h, most potentially imprecise coincidences among s_1 and s_2 fall into the same bin of the counting variables S_1 and S_2. As a consequence, the effect of jittering on the correlation measures, i.e., the cumulants of the S_i, is negligible. We therefore neglect potential imprecision in the remainder of this contribution.

To study the correlation structure of the additive Poisson model in more detail, it is instructive to write the population of spike trains $\mathbf{s}(t) = (s_1(t), \ldots, s_N(t))$ as a marked Poisson process (Fig. 12.2B; Staude et al. 2009, 2010). To this end, we first collapse the event times $\{t_i^M\}$ of all the independent component processes $\{y_M(t)\}_M$ into a single "carrier process" $m(t) = \sum_j \delta(t - t_j)$ by letting $\{t_j\} = \bigcup_M \{t_i^M\}$. Because the y_M are independent Poisson processes, also the carrier process $m(t)$ is a Poisson process, and it has a "carrier rate" of $\nu = \sum_M \nu_M$. Second, we keep track of the neuron IDs that fire at time t_j by associating to each "carrier event" an N-

dimensional binary pattern variable $\sigma_j \in \{0,1\}^N$ according to the rule $\sigma_j = \boldsymbol{\sigma}(M)$ for $t_j = t_i^M$. Then, the population of spike trains $\mathbf{s}(t)$ has the simple representation

$$\mathbf{s}(t) = \sum_j \delta(t - t_j) \cdot \sigma_j . \tag{12.12}$$

The independence of the y_M implies that the so-called "marks" σ_j associated to subsequent carrier spikes t_j can be regarded as i.i.d. random variables with common "mark distribution" P_σ. It should be evident that constructing individual spike trains $s_i(t)$ by summing the appropriate components y_M (via (12.30)) yields a (statistically) identical population as generating the population $\mathbf{s}(t)$ as a marked Poisson process (via (12.12)) if the mark distribution fulfills

$$P_\sigma(M) := \Pr\{\sigma_j = \boldsymbol{\sigma}(M)\} = \frac{\nu_M}{\nu}. \tag{12.13}$$

To simplify the correlation structure, let us for now ignore the precise neuron IDs that fire at time t_j and keep track only of the *number of neurons* $a_j = \sum_{k=1}^N (\sigma_j)_k$ (Fig. 12.2B, above the carrier $m(t)$). We call the $\{a_j\}$ the *amplitudes* of the events $\{t_j\}$ and denote their common *amplitude distribution* by

$$f_A(\xi) = \Pr\{a_j = \xi\} = \sum_{|M|=\xi} P_\sigma(M).$$

The amplitude distribution f_A encodes the orders of correlation that are present in a population of spike trains, irrespective of the specific neuron IDs that realize these correlations, and independent of their firing rates (see Fig. 12.3 for examples). If f_A vanishes above a given $\xi_0 \in \{1,\ldots,N\}$, for instance, all components y_B with $|B| > \xi_0$ vanish. With the last remark of Theorem 1, the corresponding population has correlations of maximal order ξ_0. Thus, a model whose amplitude distribution vanishes for $\xi > \xi_0$ does not have correlations beyond order ξ_0.

12.3.1 Firing Rate and Pairwise Correlations

The marked Poisson parameterization with the carrier rate ν and the amplitude distribution f_A is very intuitive in terms of the correlation structure (see Fig. 12.3). To parameterize neuronal populations, however, it is more customary to prescribe parameters that are readily accessible in empirical data, such as the firing rates λ_i and/or the pairwise correlation coefficients c_{ij} (12.3). To relate the latter to the former parameters, consider the compound signal or "population spike count" (Staude et al. 2009)

$$Z := \sum_{i=1}^N S_i . \tag{12.14}$$

Using the independence of the component variables Y_M, the cumulants of the population spike count Z can be expressed in terms of the moments of the amplitude distribution as (Sect. 12.7.4)

$$\kappa_k[Z] = \nu h \sum_{\xi=1}^{N} \xi^k \cdot f_A(\xi) = \nu h \mathrm{E}\big[A^k\big]. \tag{12.15}$$

The population-average firing rate λ is then given by

$$\lambda = \frac{1}{N}\sum_{i=1}^{N} \frac{\mathrm{E}[S_i]}{h} = \frac{\mathrm{E}[Z]}{Nh} = \frac{\kappa_1[Z]}{Nh} \overset{(12.15)}{=} \nu \frac{\mathrm{E}[A]}{N}. \tag{12.16}$$

To derive the population-average pairwise correlation coefficient $c = \langle c_{ij} \rangle$, we assume for simplicity that the population is homogeneous, so that all spike trains have identical firing rate, all pairwise covariances are identical, all triplet correlations are identical, and so forth. In this case, we obtain (e.g., Staude et al. 2009)

$$c = \frac{\kappa_2[Z] - \kappa_1[Z]}{\kappa_1[Z](N-1)} = \frac{\frac{\kappa_2[Z]}{\kappa_1[Z]} - 1}{N-1} \overset{(12.15)}{=} \frac{\frac{\mathrm{E}[A^2]}{\mathrm{E}[A]} - 1}{N-1}. \tag{12.17}$$

Note that pairwise correlations c are determined by the moments ratio $\frac{\mathrm{E}[A^2]}{\mathrm{E}[A]}$ of the amplitude distribution, but do not depend on higher moments of A. As a consequence, keeping this ratio fixed but changing higher moments of A generates populations with identical pairwise correlations but different higher-order correlations (see next section and Fig. 12.3C–F for examples). Note also that c is not affected by the carrier rate ν, illustrating a separation of correlation structure and intensity in the marked point process parameterization.

Finally, we wish to point out that the correlated Poisson processes as presented here impose explicit constraints between correlations of different orders (Johnson and Goodman 2007; Staude et al. 2009). For instance, allowing only correlations below a certain order ξ_0 imposes an upper bound on the pairwise correlation coefficient. In fact, if $f_A(\xi) = 0$ for all $\xi > \xi_0$, then

$$\frac{\mathrm{E}[A^2]}{\mathrm{E}[A]} = \frac{\sum_{\xi=1}^{\xi_0} \xi^2 f_A(\xi)}{\sum_{\xi=1}^{\xi_0} \xi f_A(\xi)} \leq \frac{\sum_{\xi=1}^{\xi_0} \xi_0 \xi f_A(\xi)}{\sum_{\xi=1}^{\xi_0} \xi f_A(\xi)} = \xi_0.$$

With (12.17), we thus have

$$c \leq \frac{\xi_0 - 1}{N-1}, \tag{12.18}$$

where equality holds for the amplitude distribution that has all its mass concentrated at ξ_0.

12.3.2 Examples

In the marked Poisson parameterization of the previous section, the generation of a population of spike trains proceeds in two steps. First, we realize the Poissonian carrier processes $m(t)$ and draw for each of its events t_j a corresponding amplitude a_j from the amplitude distribution. In a second step, we assign the spike at t_j to

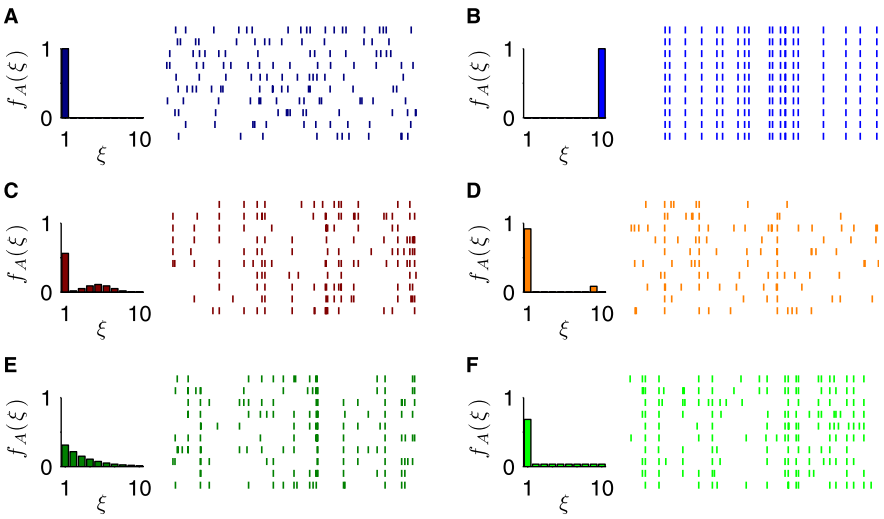

Fig. 12.3 Examples of amplitude distributions f_A and corresponding raster plots with homogeneous populations of $N = 10$ neurons. **A**. All mass is centered at $\xi = 1$, and the corresponding population consists of independent spike trains. **B**. All mass is centered at $\xi = 10$, the corresponding population is fully synchronized. In **C**–**F**, the parameters that determine f_A are set to generate identical pairwise correlation coefficient $c = 0.4$ in the populations: **(C)** MIP-like model with $\epsilon = 0.5$, $\eta = 0.55$; **(D)** SIP-like model with $\xi_{syn} = 9$, $\eta = 0.91$; **(E)** Exponential amplitude distribution $f_A(\xi;\tau) = e^{-\tau\xi} / \sum_{k=1}^{N} e^{-\tau k}$ with $\tau = 0.36$; **(F)** Uniform amplitude distribution with additional background, $f_A(\xi;\eta) = \eta\delta_{1,\xi} + \frac{1-\eta}{N}$ with $\eta = 0.65$. Note the strong differences in the higher-order structure of **C**–**F** despite the identical pairwise correlations

a_j individual process, where the precise process IDs have to be determined by a separate "assignment distribution". In the simplest case, one could assume uniform assignment, where the a_j neuron IDs that receive the spike at t_j are drawn randomly from $\{1, \ldots, N\}$, resulting in a homogeneous population. We here ignore the assignment distribution and focus on the amplitude distribution only. Then, the model has N parameters in total: the carrier rate ν and the $N - 1$ parameters of the amplitude distribution (N probabilities minus the normalization condition $\sum f_A(\xi) = 1$). A further reduction of parameters is achieved by describing the $(N-1)$-dimensional amplitude distribution with fewer parameters. Let us present two examples.

12.3.2.1 SIP-Like Models

In our first example, the amplitude distribution has two isolated peaks: a "background peak" at $\xi = 1$ that determines the number of independent spikes and a "network peak" at $\xi = \xi_{syn}$ that determines the maximal order of correlation in the population (Fig. 12.3D). Parameterizing the relative number of background and network spikes by $\eta \in [0, 1]$, the amplitude distribution can be written as

$$f_A^{\mathrm{SIP}}(\xi) = \eta\delta_{1,\xi} + (1-\eta)\delta_{\xi_{syn},\xi}, \tag{12.19}$$

where $\delta_{i,\xi} = 1$ if $i = \xi$ and zero otherwise. The parallel spike trains in Fig. 12.1 were generated by this model (C: $\xi_{syn} = 3$; F: $\xi_{syn} = 2$). This model can be interpreted as a population of independent spike trains, with additional coincident spikes injected into random subsets of ξ_{syn} processes. The Single Interaction Process (SIP) presented by Kuhn et al. (2003) is a special case of this model with $\xi_{syn} = N$.

12.3.2.2 MIP-Like Models

As a slight modification, the network peak at ξ_{syn} can be substituted by a distribution that allows for some spread around ξ_{syn}. This is achieved, for instance, by letting

$$f_A^{\mathrm{MIP}}(\xi) = \eta\delta_{1,\xi} + (1-\eta)B_{N,\epsilon}(\xi), \tag{12.20}$$

where $B_{N,\epsilon}(\xi) = \binom{N}{\xi}\epsilon^{\xi}(1-\epsilon)^{1-\xi}$ is the binomial distribution with population size N and probability ϵ (Fig. 12.3C). For $\eta = 0$, this amplitude distribution is a binomial distribution with parameters N and ϵ, and thus corresponds to a model where each event of the carrier process is copied into a spike train with fixed copy probability ϵ. Thus, this second model corresponds to the Multiple Interaction Process (MIP) presented by Kuhn et al. (2003), with additional independent background added.

Concrete expressions relating physiological parameters (λ and c) to the model parameters (ν, η, and ξ_{syn} or ϵ) are obtained by computing the first two moments of the respective amplitude distributions and inserting into (12.16) and (12.17). The resulting expressions are summarized in Table 12.1, together with the constraining relations among the parameters. Note that both model classes have three parameters and hence allow one to describe different models with identical rates and pairwise correlations. In SIP, for instance, fixing λ and c leaves ξ_{syn} as a free parameter to determine the order of correlation that is used to realize the prescribed pairwise correlations.

It is easy to add more parameters to this model. One could substitute the $B_{N,\epsilon}$-part in the second model by $B_{\xi_{syn},\epsilon}$, or use other positive, discrete distributions with finite support (see Fig. 12.3E, F). For such cases, expressions relating model parameters to average firing rates and pairwise correlations can become more complicated but can be solved by straightforward computation or using computer algebra systems, yielding a plethora of models with fixed firing rates and pairwise correlations but different higher-order properties.

12.4 Data Analysis

To use the cumulant correlations presented here for data analysis, they have to be estimated from a given data sample. The corresponding estimators, the so-called l-statistics, or polykays, have been extensively studied (Stuart and Ord 1987; Di Nardo et al. 2008). However, the estimation of higher-order cumulant correlations suffers from the same limitations as other higher-order correlation measures.

Table 12.1 Relationships and constraints between model parameters ν and η and the population average firing rate λ and pairwise correlation coefficient c for homogeneous populations of N neurons with either SIP-like amplitude distributions with fixed value of ξ_{syn} (*top*) or MIP-like amplitude distributions for fixed value of ϵ (*bottom*)

SIP-like amplitude distributions		
Prescribed parameters	λ, c and N, ξ_{syn}	ν, η and N, ξ_{syn}
Retrieved parameters	$\nu = \frac{N\lambda(\xi_{syn}-c(N-1))}{\xi_{syn}}$	$\lambda = \frac{\nu(\eta+\xi_{syn}(1-\eta))}{N}$
	$\eta = \frac{\xi_{syn}(\xi_{syn}-1-c(N-1))}{(\xi_{syn}-1)(\xi_{syn}-c(N-1))}$	$c = \frac{\xi_{syn}(\xi_{syn}-1)(1-\eta)}{(N-1)(\eta+\xi_{syn}(1-\eta))}$
Constraints	$0 \leq c \leq \frac{\xi_{syn}-1}{N-1} \leq 1$	$\nu > 0$, $\eta \in [0, 1]$

MIP-like amplitude distributions		
Prescribed parameters	λ, c and N, ϵ	ν, η and N, ϵ
Retrieved parameters	$\nu = \frac{\lambda(N\epsilon(\epsilon-c)+c)}{\epsilon^2}$	$\lambda = \frac{\nu(\eta+N\epsilon(1-\eta))}{N}$
	$\eta = \frac{N\epsilon(\epsilon-c)}{N\epsilon(\epsilon-c)+c}$	$c = \frac{N\epsilon^2(1-\eta)}{N\epsilon(1-\eta)+\eta}$
Constraints	$0 \leq c \leq \epsilon \leq 1$	$\nu > 0$, $\eta \in [0, 1]$

Namely, the number of parameters grows exponentially with the size of the neuronal population. As a consequence, exorbitant sample sizes are required for the reliable estimation of correlations of high orders (Martignon et al. 1995). In this section, we outline two recent approaches that avoid the need for such large sample sizes. Both approaches assume the additive Poisson model presented in the previous section to underlie the data, and base their inference on the superimposed activity of the discretely sampled spike trains, i.e., the population spike count Z (12.14).

12.4.1 CuBIC

Rather than directly estimating particular correlation parameters, the cumulant based inference of higher-order correlations (CuBIC) presented by Staude et al. (2009) aims at a lower bound $\hat{\xi}$ for the maximal order of correlation in a given data set. This lower bound is inferred by exploiting constraining relations between correlations of different orders within the framework of correlated Poisson processes. For instance, as shown in (12.18), if the order of correlation is at most ξ_0, the correlation coefficient in a homogeneous population cannot exceed the value $\frac{\xi_0-1}{N-1}$. Reversing this statement, a homogeneous data set with correlation coefficient $c > \frac{\xi_0-1}{N-1}$ must

have correlations of order $> \xi_0$. CuBIC exploits this observation and generalizes it in two ways. First, it estimates summed correlations in a population by the sample cumulants of the population spike count Z, thereby superseding the homogeneity constraint. And second, it allows one to infer the presence of higher-order correlations not only from measured pairwise correlations, but also from measured correlations of higher order. In total, CuBIC is a collection of statistical hypothesis tests $H_0^{m,\xi}$, indexed by the *estimated order of correlation* m and the *maximal order of correlation* allowed in the null hypothesis, ξ. The rejection of $H_0^{m,\xi}$ implies that the combination of the first mth-order correlations in the data requires correlations of at least order $\xi + 1$. Subsequent tests with different values for m and ξ finally yield the lower bound $\hat{\xi}$ for the order of correlation in a given data set.

The power of CuBIC is that the lower bound $\hat{\xi}$ can exceed the estimated order of correlation m. For instance, in artificial data sets of $N = 100$ parallel spike trains with correlations up to order 30, simulated for $T = 100$ seconds, the lower bound obtained by CuBIC with $m = 3$ can be expected in the range of ~ 25 (Staude et al. 2009), even if the effect of the higher-order correlations on pairwise correlations is extremely small ($c \sim 0.01$). Furthermore, using a nonstationary version of the additive Poisson model, time-varying firing rates can be incorporated into the formulation of the null hypothesis, allowing CuBIC to analyze also nonstationary data (Staude et al. 2010).

12.4.2 De-Poissonization

Instead of only aiming for a lower bound, the empirical de-Poissonization developed by Ehm et al. (2007) directly estimates the compounded component rates $\nu_k := \sum_{|M|=k} \nu_M = \nu \cdot f_M(k)$ from the population spike count Z. The underlying observation is that the logarithm of the characteristic function $\gamma(u) = \mathrm{E}[\exp(iuZ)]$ can be written as

$$\log \gamma(u) = h \sum_{k=1}^{N} \nu_k \left(e^{iku} - 1\right). \tag{12.21}$$

Noting that the right-hand side of (12.21) is essentially a Fourier series with coefficients ν_k obtained by Fourier-inversion,

$$\nu_k = \frac{1}{2\pi} \int_{-\infty}^{\infty} h^{-1} \log \gamma(u) e^{-iku}\, du. \tag{12.22}$$

The study of Ehm et al. (2007) is concerned with asymptotic properties of the estimators $\hat{\nu}_k$ that are defined by replacing the true characteristic function in (12.22) by their estimate, the so-called empirical characteristic function

$$\hat{\gamma}(u) = \frac{1}{L} \sum_{k=1}^{L} e^{iuZ_l},$$

where $\{Z_1, \ldots, Z_L\}$ is the available data sample. Besides the direct estimation of the compound rates ν_k, Ehm et al. (2007) derive asymptotic properties of linear functionals in the $\hat{\nu}_k$, i.e., expressions of the form $\sum_k c_k \hat{\nu}_k$. As an application, "tail sums" $\hat{\rho}_m = \sum_{k=m}^{\infty} \hat{\nu}_k$ are discussed as measures of correlation beyond a given order m. Similar to the hypothesis tests of CuBIC, a rejection of the hypothesis "$\rho_m = 0$" provides evidence for existing correlations of orders $> m$.

The major difference between these two approaches is that CuBIC exploits only the first few cumulants of the population spike count, while the empirical de-Poissonization integrates the entire empirical characteristic function to estimate the component rates ν_k (12.22). As a consequence, this latter method can be expected to have higher test power than CuBIC, but also to be less robust against empirical variations. For instance, while the analytical characteristic function γ does not have zeros in the unit disk (when considered as a polynomial in $\mathbb{C}$), its empirical counterpart $\hat{\gamma}$ can have zeros. If this is the case, the complex logarithm of $\hat{\gamma}$ can end up in a different branch, which systematically biases the estimators computed via (12.22). Furthermore, the validity of the asymptotics of the variances of $\hat{\nu}_k$ depends on expressions related to the average population spike count per bin, so that large populations, high firing rates, and/or large bin sizes might impede the reliability of the empirical de-Poissonization. Although both of these problems are discussed, and remedies for the first issue have been proposed, additional simulations are required to outline the scope of empirical de-Poissonization for particular applications.

Note that the sensitivity for higher-order correlations of both CuBIC and de-Poissonization is bought at the price of neglecting the neuron IDs that realize this correlation. This issue has to be addressed in additional steps.

12.5 Cumulants vs. Exponential Interactions

In this contribution, we presented connected cumulants as intuitive measures of higher-order correlations in populations of spike trains. In the neuroscience literature, however, attempts to elucidate the role of higher-order correlations rely almost exclusively on the higher-order parameters of the exponential, log-linear family (Martignon et al. 1995, 2000; Shlens et al. 2006; Schneidman et al. 2006; Montani et al. 2009; Shimazaki et al. 2009 and Chap. 11). In this latter framework, the bin size h is chosen such that the S_i are binary, in which case the probability to observe the pattern $\mathbf{S} = (S_1, \ldots, S_N)$ is given as

$$P_{\mathbf{S}} = \exp\left[\sum_{i=1}^{N} \theta_i S_i + \sum_{i<j} \theta_{ij} S_i S_j + \sum_{i<j<k} \theta_{ijk} S_i S_j S_k + \cdots \right.$$
$$\left. + \theta_N S_1 \cdots S_N - \psi \right]$$

$$= \prod_{i=1}^{N} \exp[\theta_i S_i] \times \prod_{i<j} \exp[\theta_{ij} S_i S_j] \times \prod_{i<j<k} \exp[\theta_{ijk} S_i S_j S_k] \times \ldots$$
$$\times \exp[\theta_N S_1 \cdots S_N] \times \frac{1}{\exp \psi}, \tag{12.23}$$

where ψ is a normalization factor, and the $\{\theta_M\}_{M \subset \{1,\ldots,N\}}$ are the "interaction parameters". To interpret the parameters θ_M, suppose that $\theta_M = 0$ for all $|M| > 1$. In this case, $P_{\mathbf{S}}$ is the product of the $\exp[\theta_i S_i]$s and does not explicitly depend on products of the S_i. Hence, the population is independent. If, however, $\theta_{M'} \neq 0$ for some $|M'| > 1$, the probabilities for patterns where $S_i = 1$ for $i \in M'$ are altered with respect to the independent case. In this sense, the parameter $\theta_{M'}$ determines the interactions of the neurons with $i \in M'$ (see Chap. 11 for more details).

Importantly, connected cumulants and the exponential interactions do not measure the same kind of dependence. While higher-order cumulant correlations indicate additive common components as described in Sect. 12.2.3, the exponential parameters directly change the probabilities of certain patterns multiplicatively. Abstractly, the former parameters therefore measure additive interactions, while the latter capture multiplicative interactions. Although mathematical differences between these concepts of dependence are being investigated (Darroch and Speed 1983; Streitberg 1990, 1999; Ip et al. 2004), little is known about their respective advantages and disadvantages for the analysis of parallel *spike trains*.

To explore potential differences between the two frameworks, we computed the respective correlation parameters from two different multivariate pattern distributions, $P_{\mathbf{S}}^{\text{MaxEnt}}$ and $P_{\mathbf{S}}^{\text{SIP}}$ (Fig. 12.4). Both distributions model homogeneous populations with binary spike counts of $N = 7$ spike trains. Furthermore, the populations are identical with respect to their single-process properties (Bernoulli processes with spike probability $p = \Pr\{S_i = 1\} = 0.001$) and pairwise correlations (correlation coefficient $c = 0.01$). The only difference between the models lies in the way higher-order correlations are arranged. Specifically, the difference lies in the framework in which higher-order correlations are set to zero.

In the first model (blue lines in Fig. 12.4), we maximized the entropy $H = -\sum_M P_{\mathbf{S}}(\boldsymbol{\sigma}(M)) \log P_{\mathbf{S}}(\boldsymbol{\sigma}(M))$ of the pattern distribution $P_{\mathbf{S}}$ with the given spike probability and pairwise correlations (Jaynes 1957b, 1957a; Bohte et al. 2000; Shlens et al. 2006; Schneidman et al. 2006; Montani et al. 2009). The resulting model $P_{\mathbf{S}}^{\text{MaxEnt}}$ is known to have no higher-order interactions in the exponential framework, i.e., $\theta_M = 0$ for all $|M| > 2$.

The second model (red lines in Fig. 12.4) is a discretized SIP-like model with $\xi_{syn} = 2$ (see Sect. 12.3.2.1), firing rate $\lambda = 1$ Hz, and a bin size of $h = 1$ ms. Note that the resulting pattern distribution $P_{\mathbf{S}}^{\text{SIP}}$ does not denote the spike trains $s_i(t)$ in continuous time, but really the population of discretized spike counts S_i. To achieve a pairwise correlation coefficient of $c = 0.01$, the model parameters ν and η were fixed according to Table 12.1. As the amplitude distribution vanishes for $\xi > 2$ in this model, it has no higher-order cumulant-correlations (see the paragraph before Sect. 12.3.1). Because the S_i are the discretized spike counts of Poisson processes, $P_{\mathbf{S}}^{\text{SIP}}$ assigns nonzero probabilities also to patterns with $S_i > 1$. However, for $\lambda h =$

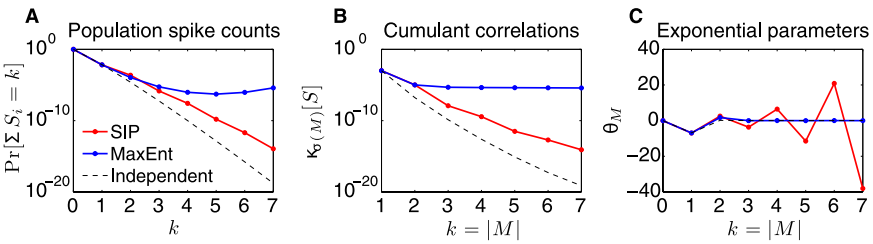

Fig. 12.4 Cumulant correlations vs. exponential interactions. Shown are the distribution of the population spike counts $Z = \sum_{i=1}^{N} S_i$ (**A**; note the logarithmic y-scale), cumulant correlations $\kappa_{\sigma(M)}[\mathbf{S}]$ (**B**), and the interaction parameters θ_M of the exponential family (**C**) of two homogeneous populations of $N = 7$ neurons (as the populations are homogeneous, we show only one parameter for every order $k = |M|$). Both populations have identical single process properties (binary Bernoulli variables with firing probability $p = 0.001$) and pairwise correlations ($c_{ij} = 0.01$). However, while $P_{\mathbf{S}}^{\mathrm{MaxEnt}}$ (*blue lines*) determines the higher-order correlations by maximizing the entropy under the given constraints and thus has $\theta_M = 0$ for $|M| > 2$, $P_{\mathbf{S}}^{\mathrm{SIP}}$ (*red lines*) is a discretized additive Poisson model (firing rate $\lambda = 1$ Hz and bin size of $h = 1$ ms) with amplitude distribution $f_A(\xi) = 0$ for $\xi > 2$ and thus has $\kappa_{\sigma(M)}[\mathbf{S}] = 0$ for $|M| > 2$. The *dashed line* shows the results for a population of independent Bernoulli processes with the same spike probability. Note the dramatic differences in higher-order correlations between the two populations

0.001, the probability for $S_i > 1$ on the order of 10^{-7}. We therefore neglect this effect and regard $P_{\mathbf{S}}^{\mathrm{SIP}}$ as the distribution of binary patterns $\mathbf{S} \in \{0, 1\}^N$.

To avoid estimation bias, we computed the exponential parameters and the cumulant correlations for both models directly from the available multivariate distributions $P_{\mathbf{S}}^{\mathrm{MaxEnt}}$ and $P_{\mathbf{S}}^{\mathrm{SIP}}$. The details of the respective computations are given in Appendix B.

As predicted by the parameterization, both models have identical cumulant correlations for orders $k = |M| = 1$ and $k = 2$ (Fig. 12.4B). For orders $k > 2$, however, the exponential parameters and cumulant correlations differ strongly between the two models.

In $P_{\mathbf{S}}^{\mathrm{SIP}}$, cumulant correlations of orders $k > 2$ are zero by construction, and the computation of $\kappa_{\sigma(M)}[\mathbf{S}]$ for $|M| > 2$ supports this with high precision (red line in Fig. 12.4B; deviations from 0 are due to numerical issues, as is illustrated by the fact that also independent Bernoulli processes have nonzero cumulant correlations to accuracy of 10^{-10}, see dashed lines). In contrast, the lack of exponential interactions in $P_{\mathbf{S}}^{\mathrm{MaxEnt}}$ produces positive higher cumulant correlations of all orders $k = 1, \ldots, 7$ (blue line in Fig. 12.4B). An intuitive explanation might be that the distribution with the highest entropy is the uniform distribution. Thus, conceptually, maximizing the entropy under constraints pushes the distribution as close to the uniform distribution as possible. As a consequence, $P_{\mathbf{S}}^{\mathrm{MaxEnt}}$ has a much higher probability for patterns with large spike counts than $P_{\mathbf{S}}^{\mathrm{SIP}}$ (Fig. 12.4A; see also Bohte et al. 2000). Interpreting the patterns with large spike count in the framework of the additive Poisson model assigns such "network events" (Schneidman et al. 2006) to injected coincidences into many neurons, i.e., nonzero-component process

$y_M(t)$ with $|M| \gg 2$. This, in turn, implies positive higher cumulant correlations (Fig. 12.4C).

The lack of cumulant correlations in $P_{\mathbf{S}}^{\mathrm{SIP}}$, on the other hand, generates strongly nonzero exponential parameters of all orders (red lines in Fig. 12.4C). The sign of these parameters alternates with the order k, generating negative correlations of odd orders and positive correlations of even orders. An interesting consequence is that the sign of the highest correlation parameter θ_N changes with N so that $\theta_N < 0$ for odd N, while $\theta_N > 0$ for even population sizes N.

In summary, the presence of cumulant correlations does not provide evidence for nonvanishing exponential parameters, and vice versa. Furthermore, the complicated behavior of the exponential parameters for $P_{\mathbf{S}}^{\mathrm{SIP}}$ makes it unlikely that a simple relationship between the two frameworks exists. More work is required to understand the relationship between θ-parameters, synchronous firing and cumulant correlations (Bohte et al. 2000; Ince et al. 2009a; Montani et al. 2009).

12.6 Conclusions

This chapter presented higher-order cumulant correlations as generalizations of the count-covariance to more than two spike trains. An "additive common component model" was presented, whose implementation in terms of correlated Poisson processes provides a very intuitive relationship between cumulant correlations and above-chance coincident firing ("excess synchrony") in neuronal populations. This makes cumulants particularly attractive when analyzing parallel spike trains with respect to synchronous activity. Importantly, cumulant correlations differ drastically from the higher-order parameters of the exponential family, as we discussed in the last section. While cumulant-correlations might be easier to interpret, the exponential parameters possess preferable statistical properties, especially with the orthogonal decomposition of correlation of different orders (see Chap. 11). We conclude that the analysis technique used for higher-order correlation measurements should be carefully chosen with respect to the scientific question in mind.

Appendix A: Cumulants

This appendix provides a mathematically rigorous definition of cumulants and lists their most important properties. We begin by reviewing characteristic functions and cumulants of a single random variable (Sect. 12.7.1), before we define the cumulant correlations as measured by the connected cumulants of vector-valued variables in Sect. 12.7.2.

12.7.1 Univariate Random Variables

12.7.1.1 Moments

Given the probability density f_S of S, the characteristic function of S is defined as

$$\gamma_S(u) := \mathrm{E}\big[\exp(iuS)\big] = \int_{-\infty}^{\infty} e^{ius} f_S(s)\, ds. \tag{12.24}$$

Inspection of the last expression of (12.24) reveals that the characteristic function of S is the Fourier transform $\mathcal{F}[f_S](u)$ of its density f_S. Using the power series expansion of the exponential function $e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$ and the linearity of the expectation operator, $\mathrm{E}[aS_1 + bS_2] = a\mathrm{E}[S_1] + b\mathrm{E}[S_2]$, we obtain a series expansion of the characteristic function in terms of the raw moments of S

$$\gamma_S(u) = \int_{-\infty}^{\infty} \sum_{k=0}^{\infty} \frac{(ius)^k}{k!} f_S(s)\, ds = \sum_{k=0}^{\infty} i^k \frac{u^k}{k!} \mathrm{E}\big[S^k\big]. \tag{12.25}$$

Thus, the raw moments $\mathrm{E}[S^k]$ can be regarded as the coefficients in a series-approximation of the distribution of S in Fourier space. Differentiation of $\gamma_S(u)$ at $u = 0$, gives back the moments

$$\mathrm{E}\big[S^k\big] = \frac{1}{i^k} \frac{\partial^k \gamma_S(u)}{\partial u^k}\bigg|_{u=0}.$$

Unfortunately, some distributions, like the log-normal distribution, have infinite moments. To avoid notational complications, however, we here assume that all moments are finite.

12.7.1.2 Cumulants

While moments are the coefficients of the series-expansion of the characteristic function, cumulants are the coefficients for the log-characteristic function. That is,

$$\psi_S(u) := \log\big(\gamma_S(u)\big) = \sum_{k=1}^{\infty} i^k \frac{u^k}{k!} \kappa_k[S], \tag{12.26}$$

and reversely

$$\kappa_k[S] = \frac{1}{i^k} \frac{\partial^k \psi_S(u)}{\partial u^k}\bigg|_{u=0}.$$

Expressions for the cumulants in terms of the moments are obtained by comparing the coefficients with equal powers of u in (12.25) and (12.26). For $k = 1, 2, 3$, we obtain

$$\begin{aligned}
\kappa_1[S] &= \mathrm{E}[S], \\
\kappa_2[S] &= \mathrm{E}\big[S^2\big] - \mathrm{E}[S]^2, \\
\kappa_3[S] &= \mathrm{E}\big[S^3\big] - 3\mathrm{E}\big[S^2\big]\mathrm{E}[S] + 2\mathrm{E}[S]^3.
\end{aligned} \tag{12.27}$$

Furthermore, the second and third cumulants correspond to the corresponding central moments, i.e.,

$$\kappa_2[S] = \mathrm{E}\big[\big(S - \mathrm{E}[S]\big)^2\big] = \mathrm{Var}[S],$$
$$\kappa_3[S] = \mathrm{E}\big[\big(S - \mathrm{E}[S]\big)^3\big].$$

For higher k, the expressions become increasingly complex (Stuart and Ord 1987), but efficient algorithms for their computation are available (Di Nardo et al. 2008).

12.7.2 Multivariate Random Variables and Correlation

Mixed cumulants arise if the random variable under consideration is vector valued, i.e., $\mathbf{S} = (S_1, \ldots, S_N)$. In this case, the log-characteristic function is a function of the N-dimensional variable $\mathbf{u} = (u_1, \ldots, u_N)$ and is defined as

$$\psi_{\mathbf{S}}(\mathbf{u}) := \log \mathrm{E}\big[\exp(i\mathbf{u} \cdot \mathbf{S})\big] = \sum_{k=1}^{\infty} i^k \sum_{|\mathbf{j}|=k} \frac{u_1^{j_1} \cdots u_N^{j_N}}{j_1! \cdots j_N!} \kappa_{\mathbf{j}}[\mathbf{S}], \tag{12.28}$$

where $\mathbf{u} \cdot \mathbf{S} = \sum_{k=1}^{N} u_k S_k$ is the standard Euclidean scalar product, and $|\mathbf{j}| = \sum_{k=1}^{N} j_k$ is the order of the multiindex $\mathbf{j} = (j_1, \ldots, j_N) \in \{0, \ldots, N\}^N$. As in the univariate case, the cumulant that corresponds to a specific multiindex $\mathbf{j}$ is obtained from $\psi_{\mathbf{S}}$ via differentiation

$$\kappa_{\mathbf{j}}[\mathbf{S}] = \frac{1}{i^{|\mathbf{j}|}} \frac{\partial^{|\mathbf{j}|} \psi_{\mathbf{S}}(\mathbf{u})}{\partial u_1^{j_1} \cdots \partial u_N^{j_N}} \bigg|_{\mathbf{u}=\vec{0}}, \tag{12.29}$$

where $\vec{0} = (0, \ldots, 0)$.

Note that cumulants $\kappa_{\mathbf{j}}[\mathbf{S}]$ are defined for arbitrary multiindices $\mathbf{j} \in \{0, \ldots, N\}^N$, while the connected cumulant used to define correlations in the main text (Definition 1) refer only to binary multiindices $\boldsymbol{\sigma}(M) \in \{0, 1\}^N$, where the set $M \subset \{1, \ldots, N\}$ determines the IDs of components among which correlation is measured (compare Gardiner 2003). For instance, $\kappa_{\boldsymbol{\sigma}(\{1,2\})}[S_1, S_2] = \kappa_{1,1}[S_1, S_2] = \mathrm{Cov}[S_1, S_2]$, while $\kappa_{\{1,2\}}[S_1, S_2] = \mathrm{Cov}[S_1, S_2^2]$. Just like the covariance measures *linear* correlation between pairs, higher connected cumulants therefore capture only linear higher-order correlation.

Let us summarize the general properties of cumulants (Streitberg 1990).

Theorem 2 *For all $M = (m_1, \ldots, m_k) \subset \{1, \ldots, N\}$, the cumulant $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ fulfills*

1. *Symmetry*: $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = \kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}^\pi]$ *for all permutations π of $\{1, \ldots, N\}$ that leave the set M invariant, where $\mathbf{S}^\pi = (S_{\pi(1)}, \ldots, S_{\pi(N)})$.*
2. *Multilinearity*: $\kappa_{\boldsymbol{\sigma}(M)}[\alpha S_1, \ldots, S_N] = \alpha \kappa_{\boldsymbol{\sigma}(M)}[S_1, \ldots, S_N]$ *for* $\alpha \in \mathbb{R}$, *and* $\kappa_{\boldsymbol{\sigma}(M)}[S_1 + S_1', \ldots, S_N] = \kappa_{\boldsymbol{\sigma}(M)}[S_1, \ldots, S_N] + \kappa_{\boldsymbol{\sigma}(M)}[S_1', \ldots, S_N]$.
3. *Moment property*: $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = \kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}']$ *if* $\mathbf{S}$ *and* $\mathbf{S}'$ *have identical mixed moments (terms of the form* $\mathrm{E}[S_1^{k_1} \ldots S_N^{k_N}]$*) up to order $|M|$.*
4. *Normalization*: *If expressing* $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ *in terms of mixed moments as in* (12.7), *the coefficient in front of* $\mathrm{E}[S_{m_1} \ldots S_{m_k}]$ *is equal to* 1.

5. *Interaction property*: *If the population* $\mathbf{S}$ *can be decomposed into two* (*or more*) *independent subgroups, none of which contains* M, *then* $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = 0$. *That is, assume that here exists* $B \subset \{1, \ldots, N\}$ *with* $M \cap B \notin \{M, \emptyset\}$ *such that* $f_{\mathbf{S}} = f_{\mathbf{S}|B} f_{\mathbf{S}|B'}$, *where* $B' = \{1, \ldots, N\} \setminus B$ *is the complement of* B, *and* $f_{\mathbf{S}|B}$ *is the multivariate distribution of the variables* S_i *with* $i \in B$. *Then* $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = 0$.

Properties 1–4 are direct consequences of the definition of the cumulants (12.28), while the interaction property results from the Fourier-and-convolution argument of Sect. 12.2.2. Interestingly, Theorem 2 can also be used to define cumulants, as cumulants are the only dependence measures that fulfill properties 1–5 (Streitberg 1990).

12.7.3 Proof of Theorem 1

Let $\{Y_B\}_{B \subset \{1,\ldots,N\}}$ be a family of independent random variables, and define the S_i by

$$S_i = \sum_{B \ni i} Y_B. \tag{12.30}$$

for $i = 1, \ldots, N$. Then we can write the population $\mathbf{S} = (S_1, \ldots, S_N)$ as

$$\mathbf{S} = \sum_{B \subset \{1,\ldots,N\}} Y_B \cdot \boldsymbol{\sigma}(B),$$

where $\boldsymbol{\sigma}(B)$ is a binary column-vector whose ith component is 1 if $i \in B$ and zero otherwise.

Let M be a subset of $\{1, \ldots, N\}$. The independence of the $\{Y_B\}$ provides

$$\begin{aligned} \kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] &= \kappa_{\boldsymbol{\sigma}(M)}\Big[\sum_{B \subset \{1,\ldots,N\}} Y_B \cdot \boldsymbol{\sigma}(B)\Big] \\ &\overset{\{Y_B\} \text{ independent}}{=} \sum_{B \subset \{1,\ldots,N\}} \kappa_{\boldsymbol{\sigma}(M)}\big[Y_B \cdot \boldsymbol{\sigma}(B)\big]. \end{aligned}$$

Now recall that $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ computes correlations among the variables $\{S_i\}_{i \in M}$ and, as such, is a sum where each summand is a product of (expectations of) all S_i with $i \in M$. If $S_i = 0$ for some $i \in M$, we therefore have $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = 0$. Let $B \subset \{1, \ldots N\}$. If $M \nsubseteq B$, there is $j \in M$ with $j \notin B$. In this case the jth component of the vector $Y_B \cdot \boldsymbol{\sigma}(B)$ equals zero. Hence $\kappa_{\boldsymbol{\sigma}(M)}[Y_B \cdot \boldsymbol{\sigma}(B)] = 0$ by the above argument. If $M \subset B$, we have $\kappa_{\boldsymbol{\sigma}(M)}[Y_B \cdot \boldsymbol{\sigma}(B)] = \kappa_{\boldsymbol{\sigma}(M)}[Y_B \cdot \boldsymbol{\sigma}(M)] = \kappa_{|M|}[Y_B]$, because $\kappa_{\boldsymbol{\sigma}(M)}$ ignores all components j with $j \notin M$, and all nonzero components of $Y_B \cdot \boldsymbol{\sigma}(M)$ are Y_B. Hence

$$\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = \sum_{B \supset M} \kappa_{|M|}[Y_B],$$

which proves Theorem 1.

12.7.4 Cumulants of the Population Spike Count in the Additive Poisson Model

Let $Z = \sum_{i=1}^{N} S_i$ be the population spike count of a population of correlated Poisson processes with carrier rate ν and amplitude distribution f_A. In Z, a given component Y_M occurs exactly $|M|$ times; hence,

$$\begin{aligned} Z &= \sum_{i=1}^{N} \sum_{M \ni i} Y_M \\ &= \sum_{M \subset \{1,\ldots,N\}} |M| Y_M . \end{aligned}$$

Using the independence of the components $\{Y_M\}$, the multilinearity of the cumulants, and the identities $\sum_{|M|=\xi} \nu_M = \nu_\xi = \nu f_A(\xi)$, the cumulants of Z can thus be computed as

$$\begin{aligned} \kappa_k[Z] &= \sum_{M \subset \{1,\ldots,N\}} \kappa_k\big[|M| Y_M\big] \\ &= \sum_{M \subset \{1,\ldots,N\}} |M|^k \kappa_k[Y_M] \\ &= \sum_{M \subset \{1,\ldots,N\}} |M|^k \nu_M h \\ &= h \sum_{\xi=1}^{N} \xi^k \underbrace{\sum_{|M|=\xi} \nu_M}_{=\nu f_A(\xi)} \\ &= \nu h \sum_{\xi=1}^{N} \xi^k f_A(\xi) \\ &= \nu h \mathrm{E}\big[A^k\big]. \end{aligned}$$

Appendix B: Computing Correlation Parameters in Practice

To compute the correlation parameters $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ and θ_M, we first derive the multivariate binary pattern distributions

$$P_{\mathbf{S}}(\mathbf{s}) = \Pr\{S_1 = s_1, \ldots, S_N = s_N\}$$

for the two models of Sect. 12.5, where $\mathbf{s} = (s_1, \ldots, s_N) \in \{0, 1\}^N$. For the MaxEnt model, the multivariate distribution $P_{\mathbf{S}}^{\mathrm{MaxEnt}}$ is a direct result of the maximization of the entropy under the given constraints (spike probability $p = 0.001$, pairwise correlation $c = 0.01$; see Ince et al. 2009b).

For the SIP model, we use the equations of Table 12.1 with $N = 7$, $\xi = 2$, $c = 0.01$, $\lambda = 1$ Hz, and $h = 1$ ms to obtain the amplitude distribution f_A and carrier rate ν. To derive the multivariate pattern distribution $P_{\mathbf{S}}^{\mathrm{SIP}}$, we compute the distribution of the population spike count $Z = \sum_{i=1}^{N} S_i$. In the marked point process framework, Z is the number of spikes of the carrier process $m(t)$ per bin, with each carrier spike being multiplied by its amplitude. As the carrier process is Poisson and amplitudes of subsequent carrier spikes are drawn independently, we have

$$\begin{aligned} P_Z(n) &= \sum_{k=1}^{\infty} \Pr\big\{m(t) \text{ has } k \text{ spikes in the bin}\big\} \\ &\quad \times \Pr\{\text{those } k \text{ spikes are copied into } n \text{ spike trains}\} \\ &= \sum_{k=0}^{\infty} e^{-\nu h} \frac{(\nu h)^k}{k!} \cdot \underbrace{f_A * \cdots * f_A}_{k \text{ times}}(n). \end{aligned} \tag{12.31}$$

In practice, (12.31) has two issues, especially for the specific case of Sect. 12.5. The first issue is that a binned SIP model does not produce strictly binary variables S_i, because the discretization may produce spike counts $S_i > 1$. For a rate of $\lambda = 1$ Hz and bin size of $h = 1$ ms as investigated in Sect. 12.5, however, the probability for this to happen is $\Pr\{S_i > 1\} = 1 - \Pr\{S_i \leq 1\} = 1 - (e^{-0.001} + 0.001 e^{-0.001}) \sim 5 \cdot 10^{-7}$. We therefore ignore this effect and assume that (12.31) is the true distribution even if we set all $S_i > 1$ to 1.

The second issue concerns the infinite sum over k, the number of spikes in the carrier process in (12.31). In practice, an upper bound k_{max} has to be chosen. For the parameters of Sect. 12.5 and $k_{max} = 100$, the result of (12.31) has a total error of $1 - \sum_{n=0}^{N} P_Z(n) \sim 10^{-16}$, illustrating that P_Z is accurately approximated with $k_{max} = 100$.

Finally, as the population under consideration is homogeneous, the probability for the patterns $\mathbf{s}$ with given spike count $n = \sum_{i=1}^{N} s_i$ can be computed form P_Z by (compare Ince et al. 2009a)

$$P_{\mathbf{S}}^{\mathrm{SIP}}(\mathbf{s}) = \frac{1}{\binom{N}{n}} P_Z(n).$$

With both multivariate pattern distributions available, connected cumulants are computed by first constructing the multivariate log-characteristic function (12.28)

$$\begin{aligned} \psi_{\mathbf{S}}(\mathbf{u}) &= \log \mathrm{E}\big[e^{i\mathbf{s}\cdot\mathbf{u}}\big] \\ &= \log \sum_{\mathbf{s}\in\{0,1\}^N} e^{i\mathbf{s}\cdot\mathbf{u}} P_{\mathbf{S}}(\mathbf{s}). \end{aligned}$$

The kth-order connected cumulant for the subset $M = (m_1, \ldots, m_k) \subset \{1, \ldots, N\}$ is then given by (12.29), i.e.,

$$\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}] = \frac{1}{i^k} \frac{\partial^k \psi_{\mathbf{S}}(\mathbf{u})}{\partial u_{m_1} \cdots \partial u_{m_k}}\bigg|_{\mathbf{u}=\vec{0}}.$$

Although this construction is straightforward, computing $\kappa_{\boldsymbol{\sigma}(M)}[\mathbf{S}]$ in practice can become tedious. The cumulants of the examples of Sect. 12.5 were computed with the help of Mathematica (Wolfram Research, Inc 2008).

The computation of the θ_Ms from $P_{\mathbf{S}}(\mathbf{s})$ is explained in detail in Chap. 11, and a Python implementation is available (Ince et al. 2009b).

References

Bahadur R (1961) A representation of the joint distribution of responses to n dichotomous items. In: Solomon H (ed) Studies in item analysis and prediction. Stanford University Press, Stanford, pp 158–168

Bair W, Zohary E, Newsome W (2001) Correlated firing in Macaque visual area MT: time scales and relationship to behavior. J Neurosci 21:1676–1697

Bell A, Sejnowski T (1996) Learning the higher-order structure of a natural sound. Network Comput Neural Syst 7:261–266

Blaschke T, Wiskott L (2004) CuBICA: independent component analysis by simultaneous third- and fourth-order cumulant diagonalization. IEEE Trans Signal Process 52:1250–1256

Bohte SM, Spekreijse H, Roelfsema PR (2000) The effects of pair-wise and higher-order correlations on the firing rate of a postsynaptic neuron. Neural Comput 12:153–179

Brette R (2009) Generation of correlated spike trains. Neural Comput 21:188–215

Brillinger D (1996) Uses of cumulants in wavelet analysis. J Nonparam Statist 6:93–114

Darroch J, Speed T (1983) Additive and multiplicative models and interactions. Ann Statist 11:724–738

De la Rocha J, Doiron B, Shea-Brown E, Kresimir J, Reyes A (2007) Correlation between neural spike trains increases with firing rate. Nature 448:802–807

Di Nardo E, Guarino G, Senato D (2008) A unifying framework for k-statistics, polykays and their multivariate generalizations. Bernoulli 14:440–468

Ehm W, Staude B, Rotter S (2007) Decomposition of neuronal assembly activity via empirical de-Poissonization. Electron J Statist 1:473–495

Fujisawa S, Amarasingham A, Harrison M, Buzsaki G (2008) Behavior-dependent short-term assembly dynamics in the medial prefrontal cortex. Nature Neurosci 11:823–833

Gardiner CW (2003) Handbook of stochastic methods for physics, chemistry and the natural sciences. Springer Series in Synergetics, vol 13, 3rd edn. Springer, Berlin

Gerstein GL, Bedenbaugh P, Aertsen A (1989) Neuronal assemblies. IEEE Trans Biomed Eng 36:4–14

Gray CM, Singer W (1989) Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex. Proc Natl Acad Sci USA 86:1698–1702

Hebb DO (1949) The organization of behavior: a neuropsychological theory. Wiley, New York

Holgate P (1964) Estimation for the bivariate Poisson distribution. Biometrika 51:241–245

Ince RAA, Montani F, Arabzadeh E, Diamond ME, Panzeri S (2009a) On the presence of high-order interactions among somatosensory neurons and their effect on information transmission. In: Proceedings of the international workshop on statistical mechanical informatics, pp 1–11

Ince RAA, Petersen RS, Swan DC, Panzeri S (2009b) Python for information theoretic analysis of neural data. Frontiers Neuroinform 3:Article 4

Ip E, Wang Y, Yeh Y (2004) Structural decompositions of multivariate distributions with applications in moment and cumulant. J Multivariate Anal 89:119–134

Jaynes E (1957a) Information theory and statistical mechanics. Phys Rev 106:620–630

Jaynes E (1957b) Information theory and statistical mechanics. II. Phys Rev 108:171–190

Johnson D, Goodman I (2007) Jointly Poisson processes. arXiv:0911.2524

Johnson D, Goodman I (2008) Inferring the capacity of the vector Poisson channel with a Bernoulli model. Network Comput Neural Syst 19:13–33

Karlis D, Meligkotsidou L (2005) Multivariate Poisson regression with covariance structure. Stat Comput 15:255–265
Kawamura K (1979) The structure of multivariate Poisson distribution. Kodai Math J 2:337–345
Kohn A, Smith MA (2005) Stimulus dependence of neuronal correlations in primary visual cortex of the Macaque. J Neurosci 25:3661–3673
Kohn A, Zandvakili A, Smith MA (2009) Correlations and brain states: from electrophysiology to functional imaging. Curr Opin Neurobiol 19:1–5
Kuhn A, Aertsen A, Rotter S (2003) Higher-order statistics of input ensembles and the response of simple model neurons. Neural Comput 1:67–101
Lancaster B, Adams PR (1986) Calcium-dependent current generating the afterhyperpolarization of hippocampal neurons. J Neurophysiol 55:1268–1282
Lancaster H (1958) The structure of bivariate distributions. Ann Math Statist 29:719–736
Lestienne R (2001) Spike timing, synchronization and information processing on the sensory side of the central nervous system. Progr Neurobiol 65:545–591
Martignon L, von Hasseln H, Grün S, Aertsen A, Palm G (1995) Detecting higher-order interactions among the spiking events in a group of neurons. Biol Cybern 73:69–81
Martignon L, Deco G, Laskey K, Diamond M, Freiwald W, Vaadia E (2000) Neural coding: higher-order temporal patterns in the neurostatistics of cell assemblies. Neural Comput 12:2621–2653
Mattner L (2004) Cumulants are universal homomorphisms into Hausdorff groups. Probab Theory Related Fields 130:151–166
Montani F, Ince RAA, Senatore R, Arabzadeh E, Diamond ME, Panzeri S (2009) The impact of high-order interactions on the rate of synchronous discharge and information transmission in somatosensory cortex. Philos Trans R Soc Lond Ser A Math Phys Eng Sci 367:3297–3310
Nakahara H, Amari S (2002) Information-geometric measure for neural spikes. Neural Comput 10:2269–2316
Papoulis A, Pillai SU (2002) Probability, random variables, and stochastic processes, 4th edn. McGraw-Hill, Boston
Riehle A, Grün S, Diesmann M, Aertsen A (1997) Spike synchronization and rate modulation differentially involved in motor cortical function. Science 278:1950–1953
Roudi Y, Nirenberg S, Latham PE (2009) Pairwise maximum entropy models for studying large biological systems: when they can work and when they can't. PLoS Comput Biol 5:e1000380
Salinas E, Sejnowski TJ (2001) Correlated neuronal activity and the flow of neural information. Nat Rev Neurosci 2:539–550
Sarmanov O (1962) Maximum correlation coefficient (nonsymmetric case). In: Selected translations in mathematical statistics and probability, vol 4. Am Math Soc, Providence, pp 271–275
Schneidman E, Berry MJ, Segev R, Bialek W (2006) Weak pairwise correlations imply strongly correlated network states in a neural population. Nature 440:1007–1012
Shea-Brown E, Josic K, de la Rocha J, Doiron B (2008) Correlation and synchrony transfer in integrate-and-fire neurons: basic properties and consequences for coding. Phys Rev Lett 100:108102
Shimazaki H, Amari S, Brown EN, Grün S (2009) State-space analysis on time-varying correlations in parallel spike sequences. In: Proc IEEE international conference on acoustics speech and signal processing ICASSP, pp 3501–3504
Shlens J, Field GD, Gauthier JL, Grivich MI, Petrusca D, Sher A, Litke AM, Chichilnisky EJ (2006) The structure of multi-neuron firing patterns in primate retina. J Neurosci 26:8254–8266
Staude B, Rotter S, Grün S (2008) Can spike coordination be differentiated from rate covariation?. Neural Comput 20:1973–1999
Staude B, Rotter S, Grün S (2009) CuBIC: cumulant based inference of higher-order correlations. J Comp Neurosci. doi: 10.1007/s10827-009-0195-x
Staude B, Grün S, Rotter S (2010) Higher-order correlations in non-stationary parallel spike trains: statistical modeling and inference. Frontiers Computat Neurosci 4:16. doi:10.3389/fncom.2010.00016
Stratonovich RL (1967) Topics in the theory of random noise. Gordon & Breach Science, New York

Streitberg B (1990) Lancaster interactions revisited. Ann Statist 18:1878–1885
Streitberg B (1999) Exploring interactions in high-dimensional tables: a bootstrap alternative to log-linear models. Ann Statist 27:405–413
Stuart A, Ord JK (1987) Kendall's advanced theory of statistics, 5th edn. Griffin and Co, London
Tetzlaff T, Rotter S, Stark E, Abeles M, Aertsen A, Diesmann M (2008) Dependence of neuronal correlations on filter characteristics and marginal spike-train statistics. Neural Comput 20:2133–2184
Vaadia E, Haalman I, Abeles M, Bergman H, Prut Y, Slovin H, Aertsen A (1995) Dynamics of neuronal interactions in monkey cortex in relation to behavioural events. Nature 373:515–518
Wolfram Research, Inc (2008) Mathematica edition: version 7.0. Wolfram Research, Inc, Champaign
Womelsdorf T, Fries P (2007) The role of neuronal synchronization in selective attention. Curr Opin Neurobiol 17:154–160
Zhou DL, Zeng B, Xu Z, You L (2006) Multiparty correlation measure based on the cumulant. Phys Rev A 74:052110

Part IV
Population-Based Approaches

Chapter 13
Information Theory and Systems Neuroscience

**Don H. Johnson, Ilan N. Goodman,
and Christopher J. Rozell**

Abstract Information theory reveals the performance limits of communication and signal processing systems, the brain being an interesting example. However, applying this powerful theory to neural signals has many pitfalls. The problem areas are discussed and we describe how to resolve the issues. In addition, we describe modern information theoretic results pertinent to neuroscience.

13.1 Introduction

Shannon's classic work (Shannon 1948) on information theory determined the ultimate fidelity limits that communication and signal processing systems can achieve. Because of obvious similarities, neuroscientists have long thought that the tools of information theory, so successful in characterizing communication systems, should enable a deeper understanding of how neural systems process information. Indeed, in systems neuroscience, many of the same issues Shannon addressed have always been research issues. How is information encoded? What is the fidelity of information represented by neural signals? Despite the similarities between the terminology used by sensory neuroscientists and communication engineers—information, information encoding and decoding, communication channels—the issues and goals of the two communities are very different. Communication engineers want to *design* systems; neuroscientists want to *analyze* an existing system. A designer wants to know what he or she has to do to meet a performance specification. Shannon's classic theory not only provides a framework for design but also shows how well the system can perform. However, Shannon's results are notoriously vague about how

D.H. Johnson (✉)
Department of Electrical and Computer Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA
e-mail: dhj@rice.edu
url: http://www.ece.rice.edu/~dhj

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_13, © Springer Science+Business Media, LLC 2010

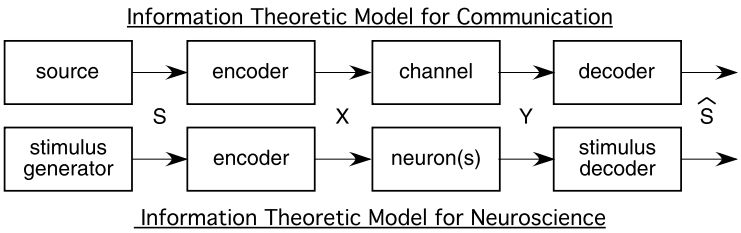

Fig. 13.1 The classic information theory model ubiquitously applied to communications and control problems is shown as well as its translation to a neuroscience context

to achieve these limits. To analyze an existing system, his work suggests what quantities could be used to assess effectiveness, but not how they can be used in this role or how they can be measured. This problem is not a question of finding good measurement techniques; rather, a much deeper problem arises. The key information theoretic quantities—capacity and the rate-distortion function—are solutions of mathematical optimization problems. His theory is silent on how to judge a given system's performance relative to these milestones. Despite this quandary, information theory encompasses many issues of interest and can provide insight into neural processing. Furthermore, post-Shannon developments do provide some tools for analyzing neural systems.

Figure 13.1 shows the fundamental model underlying classic information theory. An information source produces an information-bearing signal S that, from a neuroscience perspective, is either the stimulus in sensory systems or an intended motion in motor systems. To simplify the presentation, we use terminology from sensory systems, but information theoretic results developed here apply to motor systems as well. The signal X encodes the stimulus in such a way that preserves the information contained in the stimulus. In neuroscience, the encoder represents neural coding: how the stimulus is represented in the firing pattern of one or several neurons. For simplicity in the following discussion, X and the (population) instantaneous rate λ are one and the same. The encoded signal passes through the channel, which disturbs the signal in such a way that the channel's input X cannot be precisely determined from its output Y. In neuroscience, the channel represents a neuron or a population of neurons, wherein "channel disturbances" arise from the stochastic behavior of neural responses. Here, Y represents spike train(s), sometimes represented in neuroscience papers as the response R (Borst and Theunissen 1999). The decoder, which represents the final processing stage, produces $\widehat{S}$, an estimate of the stimulus. Thus, the standard model of information theory does, in fact, encompass many common situations of interest to neuroscience. In systems neuroscience, all elements of the standard information model shown in Fig. 13.1 occur naturally but are ill characterized to varying degrees.

This chapter has two goals: tutorial and prospective. The fundamental results of information theory are reviewed, but from a neuroscience, not a digital communication, perspective. We go on to demonstrate new results that have both practical and theoretical applications in systems neuroscience.

13.2 The Encoder

The first stage in the basic model (Fig. 13.1) is the encoder which, in a well-designed system, presumably represents an effective (and efficient) way of converting the stimulus into a signal that can be transmitted through the channel. A fundamental assumption of information theory is that all stimulus signals are stochastic and described by their probability distribution $p_S(s)$. Any encoder will be limited by the fundamental complexity of the stimulus under consideration, an effect characterized in Shannon's first important result through the notion of *entropy*.

13.2.1 Entropy

Let S be a discrete-valued stimulus that assumes one of M values, $S \in \{s_1, \ldots, s_M\}$, and let $P(s_m)$ denote the probability with which each component of the stimulus occurs. The entropy of the stimulus, $H(S)$, is defined according to

$$\boxed{H(S) \stackrel{\Delta}{=} -\sum_m P(s_m) \log_2 P(s_m).} \tag{13.1}$$

By using base-two logarithms, we implicitly express entropy in units of bits. Entropy must be nonnegative: it equals zero when one stimulus has unit probability and achieves its upper bound—$\log_2 M$—when all stimuli are equally likely. Thus, entropy generally characterizes stimulus uncertainty: the greater the entropy, the greater the uncertainty (the more uniform the stimulus probabilities). Note that discrete-valued stimulus sets usually consist of a predetermined selection of continuous-valued signals (such as a series of sinusoidal gratings at several distinct spatial frequencies, orientations and contrasts). Regardless of what comprises the stimulus set, its entropy depends *only* on stimulus probabilities and not on stimulus complexity.

Shannon showed that entropy provides an operational bound on the minimum length (number of bits) of any binary code needed to accurately represent the stimulus.

Source Coding Theorem *The smallest number of bits B required to represent a discrete-valued stimulus S without incurring decoding errors satisfies*

$$H(S) \leq B < H(S) + 1. \tag{13.2}$$

The lower bound in the Source Coding Theorem defines a fundamental limit: if fewer bits than entropy are used, at least one stimulus having a nonzero probability cannot be correctly decoded, even if communicated via a noiseless channel. The upper bound means that no more than one bit beyond entropy need be used to have a decodable source coding algorithm.

Unfortunately, extending the definition of entropy to the case of continuous-valued stimuli (a piece of music, for example) proves problematic, so much so that it

has a different name and symbol to emphasize its different properties. The so-called *differential entropy* is defined for continuous-valued random variables analogously to the definition of entropy given above:

$$h(S) \stackrel{\Delta}{=} -\int p_S(s) \log_2 p_S(s)\, ds. \tag{13.3}$$

While the entropy of a discrete-valued random variable is always nonnegative, differential entropy can assume any real value, positive or negative. For example, the differential entropy of a Gaussian random variable (having mean m and variance σ^2) equals $\frac{1}{2}\log_2 2\pi e\sigma^2$. Depending on whether $2\pi e\sigma^2$ is greater than or less than one, the differential entropy can be positive or negative, even zero if the variance precisely equals $1/2\pi e$. Moreover, when the stimulus is continuous-valued, the Source Coding Theorem *cannot* apply: the number of bits required to encode a continuous-valued quantity without error must be infinite, regardless of the underlying probability distribution. Consequently, the differential entropy has decidedly lesser importance in information theory than the entropy of a set of stimuli.

13.2.2 Entropy and Neuroscience

At first glance, entropy would thus seem to be a useful quantity in neuroscience, providing experimenters a simple means to characterize stimulus complexity. However, "large" entropy does not mean that the component stimuli $\{s_m\}$ are themselves complicated. Rather a larger entropy means the encoder requires more bits to describe the stimulus *indices*. Because the probabilities assigned to the stimulus set are typically determined by the experimenter, entropy calculations do not quantify inherent stimulus complexity in a typical laboratory setting. Therefore, these issues make it difficult to apply entropy directly to characterize stimulus complexity in most realistic laboratory settings where stimuli have probabilities chosen by the experimenter.

Whether or not entropy provides a meaningful measure of stimulus content, using entropy to characterize neural spike trains poses an even more basic dilemma (McFadden 1965). Spike times are continuous variables, so the Source Coding Theorem certainly does not apply to timing codes. But even applying it to rate codes, wherein the number of spikes is all that matters, is problematic. Consider a Poisson process (Chap. 1) having a constant rate λ over the interval $[0, T)$. The probability distribution of this process can be written two ways, both of which express the fact that in a rate code, event times do not matter.

- The probability distribution of the number of events that occur, $N_{\{0\le t<T\}}$, is given by

$$p_{N_{\{0\le t<T\}}}(n) = \frac{(\lambda T)^n e^{-\lambda T}}{n!}. \tag{13.4}$$

The event count is a discrete-valued random variable, and its entropy is nonnegative. Using (13.1) to calculate the entropy yields

$$H(N_{\{0\le t<T\}}) = \frac{\lambda T(1-\ln \lambda T) - \mathsf{E}[\ln N!]}{\ln 2}.$$

The expected value $\mathsf{E}[\ln N!]$ has no closed-form expression but is easily computed.

- The joint probability distribution of the number of events and the event times equals

$$p_{\mathbf{w},N_{\{0\le t<T\}}}(\mathbf{w}, n) = \lambda^n e^{-\lambda T}. \tag{13.5}$$

Here, $\mathbf{w} = w_1, \ldots, w_{N_{\{0\le t<T\}}}$, $0 \le w_1 \le w_2 \le \cdots \le w_{N_{\{0\le t<T\}}} \le T$, is the vector of event times. The fact that the joint distribution does not depend on event times indicates that they do not matter. Because this joint distribution contains both discrete- and continuous-valued random variables, we can only describe it using differential entropy. Evaluating the entropy according to (13.3) yields

$$h(\mathbf{w}, N_{\{0\le t<T\}}) = \frac{\lambda T(1-\ln \lambda)}{\ln 2},$$

an expression significantly different from the first.

The underlying reasons for this discrepancy are the differences between differential entropy and the entropy of discrete-valued variables. Because these two models describe the same stochastic process but cannot be described by the same entropy function, the entropy cannot be defined unambiguously for Poisson processes and, for that matter, any point process. Thus, entropy stands out from other common manipulations of probability distributions: regardless of which model is used, we obtain the same results for the expected number of events, the variance, the maximum likelihood estimate of λ, and the accompanying Cramér–Rao bound.

Measuring entropy from spike train recordings presents a similar problem. The standard approach to spike train analysis is to chop time into a sequence of abutting bins of duration Δt. Assuming the bin duration is small, either zero or one spike occurs in each bin, which means that the spike train has been converted into a Bernoulli (binary-valued) discrete-time random process. Under a Poisson model for the spike train, the probability of a spike in a bin equals $\lambda \Delta t$. Since a Bernoulli random variable is discrete valued, its entropy can be computed according to (13.1). Moreover, since the event occurrence in each bin is statistically independent under the Poisson model, the entropy of the spike train can be found by simply adding the entropies of the individual bins. Accordingly, for small values of $\lambda \Delta t$, we obtain

$$H(N_{\{0\le t<T\}}) \approx \frac{\lambda T(1-\ln \lambda) - \lambda T \ln \Delta t}{\ln 2}. \tag{13.6}$$

This result most closely resembles the differential entropy produced by the second model, but it is always positive since it is the entropy of a discrete-valued variable. However, the dangling $\ln \Delta t$ term means that the result depends on binwidth and diverges in the small binwidth limit, resulting in an infinite entropy asymptotically. Note that this binwidth term cannot be removed by normalizing.

With three different answers for the entropy of a simple point process, we must question the utility of using entropy alone to characterize neural responses. The only valid information-theoretic interpretation of entropy occurs for discrete-valued signals. Even when the stimulus is encoded via a rate code into spike counts (certainly a discrete-valued quantity), entropy cannot be uniquely defined. Looking beyond rate codes, the timing of neural responses can be translated in a discrete framework by binning, but only by making strong assumptions about the way information is encoded: assuming a fixed temporal resolution *and* a fixed internal clock defining response bin boundaries. We have already seen that even when these assumptions can be overcome, entropy calculations are suspect.

More fundamentally, when analyzing information flow, no rule governs how the entropy of a signal changes as it passes through a system: examples show that entropy can increase, decrease, or stay the same.[1] Thus, when entropy changes do occur, they cannot be interpreted to mean that information has been extracted or inserted.

One viable application of entropy to spike train analysis is testing for statistical independence of simultaneous recordings. If $\mathbf{Y} = [Y_1, \ldots, Y_N]$ represents recordings from N neurons, the joint entropy can be no larger than the sum of the component entropies, the result that obtains when the recordings are statistically independent:

$$\max_{p_{\mathbf{Y}}(\mathbf{y})} H(\mathbf{Y}) = \sum_n H(Y_n).$$

This result holds for differential entropy as well, and, regardless of how the spike trains are digitized, this property summarizes statistical dependence. Another way of stating this property is that statistical dependence reduces entropy. This result is quite powerful: the joint entropy equals the sum of the individual entropies if and only if the underlying quantities are statistically independent.

13.3 The Channel

Much more interesting to neuroscience is the way information theory characterizes the channel, especially its ability to convey information. The channel's input–output relation is defined by the conditional probability distribution $p_{Y|X}(y|x)$. Shannon's characterization of a channel begins by defining what is known today as *mutual information*.

[1] Consider a Gaussian process passing through an amplifier. Since the amplifier's gain affects the variance, the expression for a Gaussian random variable's entropy given earlier shows that it can increase or decrease.

13.3.1 Mutual Information

The mutual information between two jointly defined random quantities X and Y has similar forms for both discrete- and continuous-valued versions, exemplified here in integral form:

$$\boxed{I(X;Y) = \iint p_{X,Y}(x,y) \log_2 \frac{p_{X,Y}(x,y)}{p_X(x)p_Y(y)}\, dx\, dy.} \tag{13.7}$$

The discrete version can be obtained by replacing the integration by summation over the values taken by the channel's input and output. Again, mutual information has units of bits because of the base-two logarithm. In both discrete and continuous versions, mutual information is nonnegative. This expression can be simplified by making explicit the channel's input–output relationship, which can be rewritten as the difference of two entropies:

$$I(X;Y) = \iint p_{Y|X}(y|x)p_X(x) \log_2 \frac{p_{Y|X}(y|x)}{p_Y(y)}\, dx\, dy = h(Y) - h(Y|X). \tag{13.8}$$

Here, the so-called *conditional entropy* $H(Y|X)$ is defined to be

$$h(Y|X) \stackrel{\Delta}{=} -\int p_X(x) \left(\int p_{Y|X}(y|x) \log_2 p_{Y|X}(y|x)\, dy \right) dx.$$

Two important properties of mutual information are that $I(X;Y) = 0$ only when X and Y are statistically independent and that it achieves its maximal value when $X = Y$: $I(X;X) = H(X)$ in the discrete-valued case and $I(X;X) = +\infty$ in the continuous-valued case. Mutual information completely characterizes the degree of similarity between the statistical properties of two random variables, making it a far more powerful measure of statistical dependence than the correlation coefficient.

The difficulties in reconciling entropy between discrete- and continuous-valued cases do not carry over to mutual information. To illustrate, perhaps the simplest model for a rate code has a constant rate λ serving as the input to a Poisson process generator. To calculate the mutual information, we assume that the event rate is a random variable having the probability distribution $p_\lambda(\lambda)$. We can interpret both mathematical descriptions for the neuron's input–output relationship ((13.4) and (13.5)) as being $p_{N_{\{0\le t<T\}}|\lambda}(n|\lambda)$ and $p_{\mathbf{w},N_{\{0\le t<T\}}|\lambda}(\mathbf{w},n|\lambda)$, respectively, with λ now a random variable. In contrast to entropy, the same but difficult to evaluate result emerges from using either description:

$$I(\lambda; N_{\{0\le t<T\}}) = \sum_{n=0}^{\infty} \int \frac{(\lambda T)^n e^{\lambda T}}{n!} p_\lambda(\lambda) \log_2 \frac{\lambda^n e^{-\lambda T}}{\int \alpha^n e^{-\alpha T} p_\lambda(\alpha)\, d\alpha}\, d\lambda.$$

This result can also be derived from the Bernoulli approximation used to derive (13.6) as the inherent binwidth-related error that plagues entropy estimates does not carry over to mutual information: the offending binwidth-related term in (13.6) cancels because mutual information equals the difference of entropies (13.8). When the rate is allowed to vary with time as a stochastic process, the expression for mutual information is much more complicated (Brémaud 1981).

More importantly, writing mutual information as the difference of entropies (13.8) shows that it depends upon *both* the channel's input–output relationship and the probability distribution of the input. The dependence on the input is disguised: the probability distribution of the output $p_Y(y)$ equals $\int p_{Y|X}(y|x)p_X(x)\,dx$, showing that $H(Y)$ depends on both the channel's characteristics and the input's probability distribution. Loosely speaking, mutual information indicates how much information is shared between the input and output. Consequently, mutual information does not summarize the behavior of the channel. The mutual information between the stimulus and a measured response depends on neural processing, the stimuli, and the stimulus probabilities. Especially when the stimulus set is chosen to be nonnaturalistic stimuli (e.g., gratings) with probabilities determined by the experimenter, mutual information does not constitute an objective measure of information processing that is relevant to the organism's ecological behavior.

13.3.2 Capacity and Reliable Communication

Shannon sought a fundamental quantity that characterized the channel for any input. One way to eliminate the dependence of mutual information on the input's probability distribution is to maximize or minimize mutual information over all input probability distributions. Maximizing in this way yields the *channel capacity*, a fundamental quantity that characterizes *any* channel:

$$\boxed{C = \max_{p_X(\cdot)\in\mathcal{C}} \frac{1}{T} I(X;Y).} \tag{13.9}$$

Mutual information usually increases in proportion to the length of the observation interval T; hence the division by T, which results in capacity having units of bits/s. The maximization is usually restricted to probability distributions for the channel input that has characteristics defined by the constraint class $\mathcal{C}$. For example, in communication systems, the input power might be constrained to be less than some value. In neuroscience, we might want to constrain the spike rate to lie within some range.

In his landmark 1948 paper, Shannon showed that the channel capacity captures everything required to determine when a digital communications channel conveys an encoded source sequence at a rate of R bits/s can *reliably* communicate information.

Noisy Channel Coding Theorem *If the data rate R produced by a discrete-valued source and its encoder is less than capacity, there exists a channel coding scheme so that* all *errors incurred in the transmission of information can be corrected. Furthermore, the converse is true: if $R > C$, no scheme can prevent errors from occurring.*

The converse makes capacity a fundamental quantity: capacity uniquely defines a sharp boundary between reliable (error-free) and unreliable digital communication.

In conjunction with the Source Coding Theorem, Shannon's result completely determines (theoretically) when a discrete source can be communicated over a digital channel without error. Shannon's proof and the others that followed are not constructive: what channel coding scheme results in reliable communication over an unreliable channel is not known to this day. Systems that come closest to the capacity limit must produce a channel-input probability distribution that approaches the mutual information maximizing distribution (Shamai and Verdú 1997).

13.3.3 Capacity and Neuroscience

As we shall see, to capture how well any neural code could express a discrete stimulus set, we will need to know the capacities of a single neuron and a population. To calculate capacity for situations relevant to neuroscience, we need to detail the neural channel: how are encoded stimuli converted into a discharge pattern? The most frequently used model is the point process (Johnson 1996), wherein the occurrence of a spike depends on the stimulus and on when previous events from the same or other neurons occurred. Considering first a single neuron, the point process's intensity has the general form of $\mu(t; \mathcal{H}_t)$, where $\mathcal{H}_t$ denotes the neuron's discharge pattern history (what occurred before time t). The intensity summarizes how discharge rate at any moment depends on previous events and on the stimulus. The simplest point process is the Poisson process, wherein the intensity does not depend on previous events, which allows us to express the intensity as an instantaneous rate function: $\mu(t; \mathcal{H}_t) = \lambda(t)$.

Despite the fact that most ecological stimuli are naturally thought of as a continuous distribution, laboratory experiments are often run (out of necessity) using a discrete approximation with a finite set test of stimuli. While this artificial experimental setting does fall into the digital communications framework described above, trying to find the capacity for a simple case illustrates the problems induced by using such a randomly presented discrete-stimulus set. Again consider the constant-rate Poisson channel model expressed by (13.4), where the rate λ will assume one of M values. We can maximize mutual information to determine what rates should be chosen within the range $[\lambda_{\min}, \lambda_{\max}]$ and what the corresponding stimulus probabilities should be for effective coding (achieve capacity). Stein (1967) solved this problem, finding that equally likely stimuli are not optimal and that the rates should lie on an approximately quadratic curve. However, this result depends on the number of stimuli in a very nonlinear manner. Once the number of stimuli exceeds a threshold number, the number of "extra" stimuli should be assigned zero probability, which means presenting them anyway lowers mutual information from its maximal value (Johnson 2002). Thus, just as with entropy considerations, imposing the discrete-stimulus model on an information theoretic analysis leads to difficulties. The results thus obtained cannot be considered an approximation to the more interesting and realistic case with a continuous stimulus distribution.

Kabanov (1978) derived the capacity of the single point process channel, and Johnson and Goodman (2008) extended his result to the multiple Poisson process case. Neither of these derivations restricted the probability distribution of the channel input to have any particular form (as was done in Stein's calculation). Kabanov's result imposes minimal and maximal constraints on the instantaneous rates: $\mathcal{C} = \{\mu(t;\mathcal{H}_t)\colon\ \lambda_{\min} \le \mu(t;\mathcal{H}_t) \le \lambda_{\max}\}$. If the maximal rate were not constrained, the capacity would be infinite. Note that this maximal rate equals the peak rate a given neuron can produce, even though it may be only capable of doing so transiently. Kabanov found that $C^{(1)}$, the capacity of the single neuron channel, to be attained by a Poisson process and was related to the constraints according to

$$C^{(1)} = \frac{\lambda_{\min}}{\ln 2}\left[\frac{1}{e}\left(\frac{\lambda_{\max}}{\lambda_{\min}}\right)^{\frac{\lambda_{\max}}{\lambda_{\max}-\lambda_{\min}}} - \ln\left(\frac{\lambda_{\max}}{\lambda_{\min}}\right)^{\frac{\lambda_{\max}}{\lambda_{\max}-\lambda_{\min}}}\right].$$

Typically, a neuron's rate of discharge can dip to zero, making $\lambda_{\min} = 0$. In this case, the expression for capacity simplifies greatly and we shall frequently use this result in subsequent expressions:

$$\boxed{C^{(1)} = \frac{\lambda_{\max}}{e \ln 2}.} \tag{13.10}$$

Frey (1991) showed that when the maximal rate varies with time, as it would to describe a neuron's response to a suddenly applied stimulus, Kabanov's capacity result easily generalizes:

$$C^{(1)} = \frac{1}{e \ln 2}\frac{1}{T}\int_0^T \lambda_{\max}(t)\,dt.$$

The input that achieves capacity is known as a random telegraph wave, a square wave that randomly switches between the minimal and maximal rates in such a way that the probability of the input equaling $\lambda_{\max}$ at any random time is $1/e$. Kabanov further showed that no non-Poisson process satisfying the constraints can have a larger capacity. Our recent work details his result, showing that more realistic point process models that embrace refractory effects and other dependencies on previous events have a strictly smaller capacity neatly given by the expression

$$C^{(1)} = \max_{\mu(t;\mathcal{H}_t)\in\mathcal{C}} \frac{\mathsf{E}\left[\mu(t;\mathcal{H}_t)\right]}{e \ln 2},$$

where $\mathsf{E}\left[\mu(t;\mathcal{H}_t)\right]$ equals the expected value of intensity with respect to all possible histories. For a Poisson process that has no history effects, this quantity equals $\lambda_{\max}$. For more realistic models of single-neuron discharge characteristics, the average rate is strictly less than $\lambda_{\max}$, thereby resulting in a smaller capacity. For example, if absolute refractory effects occur (refractory interval Δ), the capacity under a maximal rate constraint of $\lambda_{\max}$ is given by

$$C^{(1)} = \frac{1}{1+\lambda_{\max}\Delta}\cdot\frac{\lambda_{\max}}{e \ln 2}. \tag{13.11}$$

Capacity results are quite limited for neural populations. In contrast with Kabanov's general single point process result, multineuron results are confined to

jointly Poisson process descriptions of a restricted type (Johnson and Goodman 2008). Two issues arise in an information theoretic analysis of a population's capacity. The first is the input innervation: do the neurons comprising the population have individual innervation, do they share a single input, or do they have a combination of the two? Secondly, are there lateral connections among the population's members? Interestingly, when no lateral connections exist, input innervation (at least for the two extremes) does *not* affect the capacity result. For a population of N neurons, each of which is modeled as a Poisson process having a maximal firing rate of $\lambda_{\max}$, the population capacity $C^{(N)}$ simply equals $NC^{(1)}$. Thus, the capacity is simply proportional to the population size, which makes a population far more capable of encoding a stimulus to a high degree of fidelity than a single neuron. When lateral connections amongst population members do exist, capacity depends on the degree of connection-induced correlation and on the input innervation. If a common input serves the population, increased coupling between neurons results in a capacity smaller than $NC^{(1)}$. If each neuron has its own input, a different result emerges: capacity increases with increased coupling, attaining values that can double it from the uncoupled baseline.

This result puts on a sound theoretical basis the empirical finding that populations can express information better than can a population of independently functioning neurons (Latham and Nirenberg 2005; Brenner et al. 2000; Schneidman et al. 2003; Reich et al. 2001): from an information theoretic perspective, the whole can be greater than the sum of its parts. "Synergy" means that $C^{(N)} > NC^{(1)}$; "redundancy" means that $C^{(N)} < NC^{(1)}$. For the Poisson population model, synergy/redundancy depend on both innervation and lateral connections. Only extremes of innervation were considered by Johnson and Goodman (2008); whether synergy can be sustained with partial sharing of inputs was not determined.

But what good befalls knowing the capacity? The capacity achieving distribution in both the single Poisson process and a Poisson population (equivalent to the neuron's discharge rate) is the random telegraph wave described above. Theory suggests that effective communications systems should have channel-input probability distributions mimicking the capacity achieving one (Shamai and Verdú 1997). We know of no firing rate resembling a square wave ever being reported, which means that neural systems viewed from an information-theoretic paradigm are operating inefficiently to some degree. When the input probability distribution does not correspond to its capacity achieving form, information theory is silent about the resulting fidelity losses. But, more fundamentally, neurons are not likely constrained to be digital devices with binary codewords and synchronized signalling with a fixed internal clock to represent a discrete set of stimuli. This makes the relevance of the Noisy Channel Coding Theorem problematic. Shannon had similar issues with analog communication schemes prevalent in his day, like AM and FM radio. Furthermore, the digital communication framework discussed above only addresses communicating information from point to point and does not encompass the information processing that underlies the fundamental task of neural systems. Is channel capacity relevant to neuroscience? In response to his own questions about generalizing the communication paradigm, Shannon developed his all encompassing result: rate-distortion theory.

13.4 Entire System Analysis

13.4.1 Rate-Distortion Theory

The Source Coding Theorem and Noisy Channel Coding Theorem together provide fundamental limits on reliable communication for discrete-valued sources. Rate-distortion theory embraces all situations—discrete- and continuous-valued sources and channels—and provides a unifying theory of entire communication systems. This theory begins by introducing a *distortion measure* $d(S, \widehat{S})$ that expresses how the fidelity of the communication (or signal processing) system is to be assessed. As in Fig. 13.1, S represents the stimulus, and $\widehat{S}$ its "estimate". Presumably the distortion increases as the discrepancy increases between the stimulus and its reconstructed value. Interestingly, Shannon's framework allows *any* reasonable distortion measure ($d(S, \widehat{S}) \geq 0$, equaling zero when $\widehat{S} = S$). It can be chosen according to whatever criteria evaluate what effective communication means in any particular scenario. A common distortion measure used in communications and signal processing is the squared-error measure: $d(S, \widehat{S}) = (\widehat{S} - S)^2$. More relevant to sensory neuroscience would be a perceptual error measure, such as one that reflects Weber's Law. Importantly, the distortion measure could also incorporate some desired stimulus processing, in which $\widehat{S}$ is an estimate of some feature of S. Rate-distortion theory also applies to motor systems, where S represents the intended motion, and $\widehat{S}$ the actual motion. In this case, the distortion measure might include a penalty for too slow a motion as well as path and target errors.

Next, define the average distortion $\overline{D}$ as the expected value of the distortion measure with respect to the joint distribution of the stimulus and its estimate:

$$\overline{D} \stackrel{\Delta}{=} \mathsf{E}\big[d(S, \widehat{S})\big] = \iint d(s, \hat{s}) p_{S,\widehat{S}}(s, \hat{s})\, ds\, d\hat{s} = \iint d(s, \hat{s}) p_{\widehat{S}|S}(\hat{s}|s) p_S(s)\, ds\, d\hat{s}.$$

The conditional distribution $p_{\widehat{S}|S}(\hat{s}|s)$ depends on virtually everything in a neural coding scenario: how the stimulus is encoded, the neuron's spiking characteristics, and how the decoder works. Shannon proceeded by defining the *rate-distortion function* $\mathcal{R}(D)$ to be the minimal mutual information between the stimulus and its estimate over *all* possible channels, encoders and decoders that yield an average distortion smaller than D:

$$\boxed{\mathcal{R}(D) \stackrel{\Delta}{=} \frac{1}{T} \min_{p_{\widehat{S}|S}(\cdot|\cdot):\overline{D} \leq D} I(S; \widehat{S}).} \tag{13.12}$$

Note that the minimization is calculated over *all* possible relationships between a stimulus and its estimate, not just the one under study. Rate-distortion functions are notoriously difficult to calculate, with only a few results known. If the stimulus source is a bandlimited Gaussian random process having power P and bandwidth W, and the distortion measure is squared error, the rate-distortion function equals (Cover and Thomas 2006)

$$\mathcal{R}(D) = \begin{cases} W \log_2 \frac{P}{D}, & D \leq P, \\ 0, & D > P. \end{cases} \tag{13.13}$$

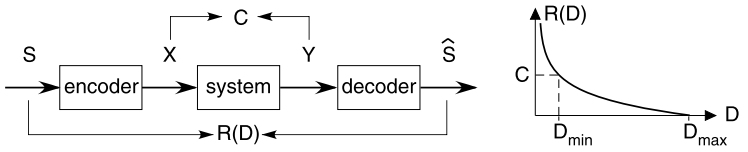

Fig. 13.2 The key quantities in Shannon's classic information theory—capacity and the rate-distortion function—are defined in the context of the standard model shown in Fig. 13.1. Capacity (C) summarizes the channel that presumably introduces disturbances into the communication process. The rate-distortion function $\mathcal{R}(D)$ depends solely on stimulus characteristics. Shannon's Rate-Distortion Theorem relates these two quantities, showing that the smallest possible distortion $D_{\min}$ is determined by $C = \mathcal{R}(D_{\min})$

This result, shown in Fig. 13.2, illustrates the properties *all* rate-distortion functions satisfy.

- $\mathcal{R}(D)$ is a strictly decreasing and convex function.
- $\mathcal{R}(D)$ equals zero at some distortion, equaling zero for any larger values. This critical value is known as the maximal distortion $D_{\max}$ and corresponds to the decoder's best guess as to what the stimulus might be with no data. For example, in the Gaussian case, mean-squared distortion is minimized by guessing $\widehat{S} = \mathsf{E}[S]$. Consequently, no combination of encoder, channel, and decoder should yield a distortion larger than $D_{\max}$.

Shannon's crowning result, the *Rate-Distortion Theorem*, unifies all of his results.

Rate-Distortion Theorem *The distortion at which the rate-distortion function equals the channel capacity defines the smallest possible distortion* $D_{\min}$ any *encoder and decoder can obtain* (*see Fig.* 13.2): $\mathcal{R}(D_{\min}) = C$.

The rate-distortion function depends only on stimulus characteristics and on the desired level of distortion. Capacity summarizes the properties of the channel and determines via the rate-distortion function the smallest possible distortion that any encoder and decoder can achieve for a given source. Shannon's proof of this result was not constructive, providing no guidance on how to find the encoder/decoder pair that produces the smallest possible distortion. However, the value of $D_{\min}$ defined by the source and the channel determines how well a given system can perform, thereby serving as a benchmark.

When the source is discrete valued, the value of the rate-distortion function is indeed the data rate, the number of bits transmitted per second that can result in a specified or greater distortion. Thus, the Source Coding Theorem and the Channel Coding Theorem are actually special cases of the Rate Distortion Theorem: for discrete sources, the zero-distortion rate equals the entropy of the source and is only achievable if it does not exceed the capacity of the channel.

However, rate-distortion theory also applies when *no* digital scheme is involved. We can interpret $\mathcal{R}(D)$ as the virtual data rate in such examples, which would suggest that some equivalent digital scheme exists. Whether this equivalent system can

be found or not is irrelevant. More importantly, "bits/s" is the fundamental unit of exchange in *any* communication or signal processing system, whether it communicates discrete- or continuous-valued signals. Information-theoretic rate (not to be confused with spike rate) serves as an intermediary. What we really want to know is the distortion a given channel imposes. Because the rate-distortion function is always a decreasing function, increasing capacity *always* allows the possibility of a smaller distortion. In our Gaussian stimulus example (13.13), the smallest possible distortion decreases exponentially with capacity,

$$\boxed{D_{\min} = D_{\max} 2^{-C/W}.} \tag{13.14}$$

Note that this result applies no matter what channel intervenes between the encoder and decoder. It could be a radio channel, cable television, or a group of neurons. This generality of information theory makes it fundamental to the study of communication, signal processing, and control systems, be they natural or man-made.

13.4.2 Rate-Distortion Theory and Neuroscience

To explore how to use rate-distortion theory in a neuroscience application, consider encoding a Gaussian stimulus in the discharge rate of a single neuron and use mean-squared error as the average distortion measure. We can determine the effectiveness of *any* single-neuron encoding/decoding scheme for Gaussian stimuli by simply plugging in the capacity formula: $D_{\min} = D_{\max} \exp\{-\lambda_{\max}/eW\}$. Thus, the maximal rate needs to be several times the bandwidth to obtain significant distortion reductions from the performance $D_{\max}$ provided by the intelligent but data-blind decoder. Even for signals with a modest bandwidth (visual signals having a bandwidth of about 30 Hz, for example), maximal firing rates would need to be well over 100 Hz for a single neuron to represent temporal stimulus changes accurately. In the auditory system, the situation is much worse. Auditory-nerve fibers having a center frequency of 1 kHz have a bandwidth of about 500 Hz. Thus, a single neuron would need to be capable of a maximal discharge rate of several thousand spikes/s. In this case, refractory effects would come into effect, and, as (13.11) shows, higher rates may not suffice: refractory effects limit capacity to be no more than $1/(\Delta t \cdot e \ln 2)$. We arrive at our first result directly applicable to neuroscience: *only with population coding can measured perceptual distortions be achieved*. Using $C^{(N)} = N\lambda_{\max}/(e \ln 2)$ for a population's capacity, the best possible distortion decreases exponentially in the population size: $D_{\min} = D_{\max} \exp\{-N\lambda_{\max}/eW\}$. Now, almost without regard to its input innervation pattern or to the stimulus bandwidth, a sufficiently large population has the capability to render a stimulus accurately. When synergy occurs, the exponent increases, which means that smaller distortions can occur with the same size population.

13.5 Post-Shannon Information Theory

Many developments have occurred in information theory since the publication of Shannon's defining results. For example, Gastpar et al. (2003) found that if the distortion measure $d(S, \widehat{S})$ and the capacity constraint class $\mathcal{C}$ are well matched to source and channel characteristics, then the rate-distortion limit can be achieved. For example, squared-error distortion was shown to be well matched to the additive Gaussian noise channel commonly used in studies of communication systems. A distortion measure well matched to the Poisson channel has not been found.

On another front, the importance of the Kullback–Leibler distance to systems neuroscience has become evident, especially in light of a post-Shannon fundamental result known as the Data Processing Theorem.

13.5.1 Kullback–Leibler Distance

The Kullback–Leibler distance between two probability distributions is defined by

$$\mathcal{D}_X(\boldsymbol{\alpha}_1 \| \boldsymbol{\alpha}_0) = \int p_X(x; \boldsymbol{\alpha}_1) \log_2 \frac{p_X(x; \boldsymbol{\alpha}_1)}{p_X(x; \boldsymbol{\alpha}_0)} \, dx. \tag{13.15}$$

The semicolons in the probability functions denote that $\boldsymbol{\alpha}_0$ and $\boldsymbol{\alpha}_1$ are parameters of the distribution, not random variables. While mutual information defines how similar two random variables are, the Kullback–Leibler distance characterizes disparity: how much one random variable's probability distribution changes when its parameters change from $\boldsymbol{\alpha}_0$ to $\boldsymbol{\alpha}_1$. Only when the probability distributions are unchanged does this quantity equal zero; in all other cases, the Kullback–Leibler distance is positive. This distance can be infinite, expressing that the two distributions are maximally different. Note that calling this quantity a "distance" is a misnomer: it is not a symmetric function of the two probability distributions, a technical requirement for any quantity to be called a distance. Consequently, we do not use "distance" in the strict sense here.

The Kullback–Leibler distance has emerged as *the* information-theoretic quantity to assess how two probability distributions differ. Furthermore, it can be related to how well the parameters can be estimated and how well the parameter change can be discerned with an optimal detector (Sinanović and Johnson 2007). The bigger the Kullback–Leibler distance for some parameter change, the smaller the error in estimating the parameter, and the easier a parameter change is to detect. We interpret the parameter $\boldsymbol{\alpha}$ to reflect stimulus parameters, with $\boldsymbol{\alpha}_0$ and $\boldsymbol{\alpha}_1$ denoting two different stimulus conditions. Unfortunately, a relation between the Kullback–Leibler distance ratio and mutual information has not been discovered and may not exist, making it difficult to relate Kullback–Leibler analysis to channel coding.

13.5.2 Data Processing Theorem

Of particular relevance to neuroscience is how stimulus fidelity—the ability to extract information—changes when a system acts on an information-bearing signal.

Data Processing Theorem *The Kullback–Leibler distance between two inputs* $X(\boldsymbol{\alpha}_0)$ *and* $X(\boldsymbol{\alpha}_1)$ *must exceed that between the corresponding outputs* $Y(\boldsymbol{\alpha}_0)$ *and* $Y(\boldsymbol{\alpha}_1)$:

$$\mathcal{D}_X(\boldsymbol{\alpha}_1 \| \boldsymbol{\alpha}_0) \geq \mathcal{D}_Y(\boldsymbol{\alpha}_1 \| \boldsymbol{\alpha}_0) \quad or \quad \gamma_{X,Y}(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1) \stackrel{\Delta}{=} \frac{\mathcal{D}_Y(\boldsymbol{\alpha}_1 \| \boldsymbol{\alpha}_0)}{\mathcal{D}_X(\boldsymbol{\alpha}_1 \| \boldsymbol{\alpha}_0)} \leq 1.$$

Successive stages of processing only maintains or, more likely, decreases the fidelity to which information can be extracted. The closer the ratio $\gamma_{X,Y}(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1)$ is to one, the more the system has preserved in its output the fidelity of the parameter encoding inherent in the input. Note that as opposed to classic information theory, which seeks the ultimate limits of fidelity, the Data Processing Theorem can be used to assess whether existing systems are optimal or not. The Kullback–Leibler distance has been used to measure the fidelity of neural coding (Johnson et al. 2001; Rozell et al. 2004). A system theory based on Kullback–Leibler distance analysis complements classic information theoretic considerations (Sinanović and Johnson 2007). In particular, for a population innervated with a common input, the ratio $\gamma_{X,Y}(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1)$ has been shown to increase monotonically with population size N regardless of what comprises the population (Johnson 2004).

13.6 Measuring Information Theoretic Quantities

How to measure entropy, mutual information and the Kullback–Leibler distance from recordings has been largely overlooked by the information theory community. The best algorithms have been developed by neuroscientists seeking to quantify neural information processing. We do not include capacity or the rate-distortion function in the list because they result from maximizing/minimizing mutual information. Capacity would seem to be possible to estimate since it depends only on $p_{Y|X}(y|x)$, the stimulus-response characteristic of the neural system in question. To perform the required maximization, this quantity would be needed for all conceivable stimuli, which creates a daunting empirical task. If the stimulus-response characteristics were available, the Blahut–Arimoto algorithm (Cover and Thomas 2006) could be used to compute capacity. Also, as noted earlier, the capacity can be calculated analytically from a few simple parameters (e.g., maximum firing rate) when the responses are well modeled by simple point process encoders.

13.6.1 Entropy and Mutual Information

Aside from questions pertaining to analytically calculating or interpreting entropy and mutual information in neuroscience, there are many interesting and difficult statistical issues that must be considered when trying to estimate these quantities empirically from limited neural data. The primary difficulty arises because the quantities of interest depend critically on the joint distribution $p_{X,Y}(x, y)$ over very high-dimensional stimulus and responses spaces with limited observations of $p_{Y|X}(y|x)$ and highly restricted choices of $p_X(x)$. A number of methods have been devised by the neuroscience community to estimate mutual information or entropy (useful in calculating mutual information via (13.8)), generally customized to try and deal with the particular strengths and weakness of different experimental scenarios (repeated trials vs. many single trials, for example) and types of signals (binned spike times vs. continuous spike times or nonspike waveforms). Victor (2006) provides a thorough review of the many different estimation methods.

One of the first mutual information estimation methods used in analyzing neural data is the "reconstruction method". In this approach, the best linear predictor of the stimulus given the spike response is found, and the mutual information is calculated from the stimulus to the reconstruction (Bialek et al. 1991). Exploiting the Data Processing Theorem expressed in terms of mutual information (Cover and Thomas 2006), using *any* system in this calculation will give a lower bound on the true mutual information. While this technique is not guaranteed to give a tight bound, it is sometimes possible to derive upper bounds independently based on physiological constraints that can reduce the uncertainty in estimating the true mutual information.

The most commonly used mutual information estimation method is the "direct method". This algorithm bins the response of a single neuron into discrete time slices to indicate whether a spike in each bin. The method then forms "words"—a sequence of bits—from the binary values both over time and across jointly recorded neurons. The probability of a word occurring is estimated with a simple histogram, and the quantity of interest is calculated by "plugging in" the estimated distributions into the desired formulas (Strong et al. 1998). Many of these estimators have significant bias and variance; several lines of work have attempted to reduce these effects. Paninski (2003) provides a nice discussion of estimator quality.

Several researchers have taken a different tack by decomposing mutual information into quantities that express how much rate, timing, and interneuronal correlations contribute to the overall value (Panzeri et al. 1999; Panzeri and Schultz 2001) (see also Chap. 14). This approach provides an information-theoretic insight into various components of a population response.

13.6.2 Kullback–Leibler Distance

The Kullback–Leibler distance and the input–output ratio of distances was developed expressly for empirical studies. Theoretical results (Johnson et al. 2001;

Sinanović and Johnson 2007) relate measured distance values to the asymptotic performance of optimal processing systems that use the signal in question as its input. Many of the statistical techniques developed for measuring entropy and mutual information apply to the Kullback–Leibler distance as well. In particular, an algorithm very similar to the direct method described above has been successfully applied to estimating Kullback–Leibler distances when paired with statistical resampling techniques to reduce bias (Johnson et al. 2001; Rozell et al. 2004).

13.7 Conclusions

Shannon's information theory is one of the greatest technical achievements of the twentieth century. In fact, once the power of Shannon's results became evident, the title of his work changed from "*A* mathematical theory of communication" to "*The* mathematical theory…". His theory defines the ultimate fidelity limits that communication and information processing systems can attain under a wide variety of situations. A good reference for information theory is the book by Cover and Thomas (2006).

However, information theory provides few answers when it comes to analyzing a preexisting system, whose components and constraints are only vaguely known. For neuroscience applications, even the simple act of applying a stochastic stimulus presents a major hurdle. Furthermore, the point-process nature of the signals complicates the use of information theoretic viewpoints such as rate-distortion theory where only a very few situations (Gaussian and Bernoulli statistics) can be characterized analytically. Finally, we have followed in the footsteps of many other authors in ignoring the existence of feedback. Feedback is a notoriously difficult concept to handle with the traditional tools of information theory, but it would be foolish to discount the role of feedback connections when they are so prominent anatomically even in the most peripheral sensory pathways. Some recent work has sparked interested on feedback in the information theory community (Massey 1990; Venkataramanan and Pradhan 2005), however none of it has yet been translated to a neural setting.

Despite these obstacles, information theory has certainly provided insights into neural processing that other approaches have not provided. For information theory to yield significantly more important gains in our understanding of neural processing, the difficulties surrounding empirical studies of information theoretic properties need to be removed, and the structures it can embrace expanded. In particular, the grand challenge of finding tractable approaches to calculating rate-distortion functions and understanding the role of feedback in reasonably complex systems needs to be advanced.

References

Bialek W, Rieke F, de Ruyter van Steveninck RR, Warland D (1991) Reading a neural code. Science 252:1852–1856

Borst A, Theunissen FE (1999) Information theory and neural coding. Nature Neurosci 2:947–957
Brémaud P (1981) Point processes and queues. Springer, New York
Brenner N, Strong SP, Koberle R, Bialek W, de Ruyter van Steveninck RR (2000) Synergy in a neural code. Neural Comput 12:1531–1552
Cover TM, Thomas JA (2006) Elements of information theory, 2nd edn. Wiley, New York
Frey MR (1991) Information capacity of the Poisson channel. IEEE Trans Inform Theory 37:244–256
Gastpar M, Rimoldi B, Vetterli M (2003) To code, or not to code: lossy source-channel communication revisited. IEEE Trans Inform Theory 49:1147–1158
Johnson DH (1996) Point process models of single-neuron discharges. J Comp Neurosci 3:275–299
Johnson DH (2002) Four top reasons mutual information does not quantify neural information processing. In: Computational neuroscience '02, Chicago
Johnson DH (2004) Neural population structures and consequences for neural coding. J Comp Neurosci 16:69–80
Johnson DH, Goodman IN (2008) Inferring the capacity of the vector Poisson channel with a Bernoulli model. Network Comput Neural Syst 19:13–33
Johnson DH, Gruner CM, Baggerly K, Seshagiri C (2001) Information-theoretic analysis of neural coding. J Comp Neurosci 10:47–69
Kabanov YM (1978) The capacity of a channel of the Poisson type. Theory Probab Appl 23:143–147
Latham PE, Nirenberg S (2005) Synergy redundancy, and independence in population codes, revisited. J Neurosci 25:5195–5206
Massey JL (1990) Causality feedback and directed information. In: Proc. 1990 intl. sym. on information theory and its applications, Waikiki
McFadden JA (1965) The entropy of a point process. SIAM J Appl Math 13:988–994
Paninski L (2003) Estimation of entropy and mutual information. Neural Comput 15:1191–1253
Panzeri S, Schultz SR (2001) A unified approach to the study of temporal, correlational, and rate coding. Neural Comput 13:1311–1349
Panzeri S, Schultz SR, Treves A, Rolls ET (1999) Correlations and the encoding of information in the nervous system. Proc R Soc Lond 266:1001–1012
Reich DS, Mechler F, Victor JD (2001) Independent and redundant information in nearby cortical neurons. Science 294:2566–2568
Rozell CJ, Johnson DH, Glantz RM (2004) Measuring information transfer in crayfish sustaining fiber spike generators: methods and analysis. Biol Cybernet 90:89–97
Schneidman E, Bialek W, Berry MJ II (2003) Synergy, redundancy, and independence in population codes. J Neurosci 23:11539–11553
Shamai S, Verdú S (1997) Empirical distribution for good codes. IEEE Trans Inform Theory 43:836–846
Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27:379–423, 623–656
Sinanović S, Johnson DH (2007) Toward a theory of information processing. Signal Process 87:1326–1344
Stein RB (1967) The information capacity of nerve cells using a frequency code. Biophys J 7:67–82
Strong SP, Koberle R, de Ruyter van Steveninck RR, Bialek W (1998) Entropy and information in neural spike trains. Phys Rev Lett 80:197–200
Venkataramanan R, Pradhan SS (2005) Directed information for communication problems with common side information and delayed feedback/feedforward. In: Proc. Allerton conference on communication, control and computing
Victor JD (2006) Approaches to information-theoretic analysis of neural activity. Biol Theory 1:302–316

Chapter 14
Population Coding

Stefano Panzeri, Fernando Montani, Giuseppe Notaro, Cesare Magri, and Rasmus S. Peterson

Abstract Population coding is the quantitative study of which algorithms or representations are used by the brain to combine together and evaluate the messages carried by different neurons. Here, we review an information-theory-based approach to population coding. We discuss how to quantify the information carried by a neural population and how to quantify the contribution of individual members of the population, or the interaction between them, to the overall information encoded by the considered group of neurons. We present examples of applications of this formalism to simultaneous recordings of multiple spike trains.

14.1 Introduction

Brains support highly reliable sensory perception: in most conditions, animals need to be presented a sensory stimulus only once in order to perceive it correctly, and this perception is highly robust even to the presence of sensory noise. Yet, responses of individual neurons in the central nervous system of mammals are often highly variable: repeated presentations ("trials") of the same stimulus elicit each time a different single-neuron response. As a result, single-neuron messages are ambiguous. Neurophysiologists, in order to describe the responses of single neurons to stimuli, often reduce the effect of this variability by averaging responses over repeated trials. However, the brain itself must resort to a different strategy than trial averaging, because it can process information and take decisions based on single events. It is widely believed that the brain makes sense of the noisy responses of individual neurons by evaluating the activity of large neural populations.

Population coding is the quantitative study of which algorithms or representations are used by the brain to combine together and evaluate the messages carried

S. Panzeri (✉)
Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy
e-mail: stefano.panzeri@iit.it
url: http://www.iit.it

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_14, © Springer Science+Business Media, LLC 2010

by different neurons. In other words, population coding studies how other neurons within a population resolve the ambiguity of the messages carried within a single trial by each individual neuron (Quian Quiroga and Panzeri 2009), for example, by coordinating their relative firing time to tag particularly salient events (Singer and Gray 1995; Engel and Singer 2001) or by having each neuron represent a particular stimulus feature (Quian Quiroga et al. 2005; Reich et al. 2001; Barlow et al. 1964).

In this chapter, we will consider an "information-based" approach (Quian Quiroga and Panzeri 2009) to population coding, i.e. an approach based on information theory (see Chap. 13). We will discuss how to quantify the information carried by a neural population and how to quantify the contribution of individual members of the population, or the interaction between them, to the overall encoding of information by the considered group of neurons.

The chapter is structured as follows. First, we establish the basic definitions and notation. Second, we consider a breakdown of the population information which permits to quantify the impact of correlations across neurons to population coding. Third, we discuss a link between information-based and decoding-based population analysis. Fourth, we consider how to address whether pooling together the responses of different neurons is a viable population coding mechanism.

14.2 Definitions of the Experimental Quantities

Consider an experiment in which the animal is presented with a stimulus s selected with probability $P(s)$ from a stimulus set S, and the consequent response of a population of C neurons is recorded and quantified in a certain poststimulus time window. We assume that the neural population response is quantified as a discrete, multidimensional array $\mathbf{r} = r_1, \ldots, r_C$ of dimension C, where r_c is the response of neuron c on a given trial in the response window. For example, the experimenter may be interested in a spike count code. In this case, r_c would be the number of spikes emitted by neuron c during the trial in the response window. Or else, the experimenter may wish to investigate a spike timing code. In this case, the response window can be divided into L bins of width Δt (the assumed time precision of the code), and the response r_c over the individual neuron is an L-dimensional array containing the number of spikes fired in each time bin (Strong et al. 1998). We denote by $\mathbf{R}$ the set of possible values taken by the response array.

The reason why it is convenient to quantify the neural response as a discrete variable is that it makes it easier to estimate the probability of different neural responses to stimuli, which are necessary for information calculation (see below). Discrete response representations are very natural for spike train analysis, due to the all-or-none nature of spikes (for example, the spike count is a discrete variable). If the response is measured as an analog signal (such as, for example, a Local Field Potential), the signal can also be naturally discretized into bins, although in principle with analog signals it is possible to circumvent the discretization in cases when there are suitable analytical models for the probability distribution (e.g. Gaussian distribution).

Having measured the responses to the presented stimuli, we can quantify how well it allows us to discriminate among the different stimuli by using Shannon's mutual information (see Chap. 13):

$$I(S;\mathbf{R}) = \sum_{s\in S}\sum_{\mathbf{r}\in\mathbf{R}} P(s)P(\mathbf{r}|s)\log_2\frac{P(\mathbf{r}|s)}{P(\mathbf{r})}, \tag{14.1}$$

where $P(\mathbf{r}|s)$ is the probability of observing response $\mathbf{r}$ when stimulus s is presented, $P(s)$ is the probability of presentation of stimulus s, and $P(\mathbf{r})$ is the probability of observing $\mathbf{r}$ in response to any stimulus. For the purpose of illustration, a simple algorithmic implementation of Eq. (14.1) is reported in Fig. 14.1.

Mutual information quantifies how much of the information capacity provided by stimulus-evoked differences in neural activity is robust to the presence of trial-by-trial response variability. Alternatively, it quantifies the reduction of uncertainty about the stimulus that can be gained from observation of a single trial of the neural response. We refer to Chap. 13 for a complete introduction to information theory. In this chapter, we will concentrate on how to adapt information theory to the study of population codes.

14.3 Quantifying the Role of Correlated Firing in Population Coding

Simultaneous recordings of the activity of individual neurons placed within local networks in the central nervous system show that a large fraction of pairs of neurons are correlated. The probability of observing near-simultaneous spikes from two different neurons is often significantly higher than the product of the probability of observing the individual spikes from each neuron (Li 1959; Perkel et al. 1967; Mastronarde 1983). The ubiquitous presence of correlations among the activity of different neurons has raised the question of what is the impact of correlation upon neural population coding of sensory stimuli (see Averbeck et al. 2006; Salinas and Sejnowski 2001 for recent reviews). Although the potential role of correlations in neural population codes is still unclear and robustly debated (Shadlen and Movshon 1999; von der Malsburg 1999), theoretical studies have suggested that correlations can profoundly affect the information transmitted by neural populations (Abbott and Dayan 1999; Averbeck et al. 2006). It is therefore of great interest to quantify, from experimental simultaneous recordings of population of neurons, the impact of correlations on the information carried by a population of neurons. This section describes algorithms and quantities that have been designed for this purpose.

14.3.1 Signal vs Noise Correlations

Before we consider the impact of correlations on coding, let us first introduce precisely what correlations are. Suppose that we record from two somatosensory neurons in the rat cerebral cortex which receive both a common connection from thalamic neurons selective for a particular whisker (let us call it the "preferred" whisker),

```
% A simple Matlab routine that computes the information between a  stimulus set
% and the spike count response of a single neuron

% The discrete stimulus set is made of Ns elements numbered from 1 to Ns
% The responses are spike counts from a single neurons in eachtrial
% totNt denotes the total number of experimental trials
% We assume that the data consist of two vectors, as follows:
% s_vec(i) is the stimulus presented in the i-th trial
% r_vec(i) is the spike count in the i-th trial

% Initializing arrays:
totNt=size(s_vec,2); % total number of experimental trials
Ns=max(s_vec);  % number of different stimuli
maxcount=max(r_vec);  % maximum spike count
Nr=maxcount+1; %number of different responses
Nt  = zeros(Ns,1);  % number of trials per stimulus
Pr  = zeros(Nr,1);  % unconditional response probability P(r)
Prs = zeros(Nr,Ns); % stimulus conditional response probability P(r|s)
Ps  = zeros(Ns,1);  %  probability of stimulus presentation P(s)
% Estimating probabilities:
for k=1:totNt
    s = s_vec(k);
    r = r_vec(k);
    Pr(r+1)    = Pr(r+1)    + 1;
    Prs(r+1,s) = Prs(r+1,s) + 1;
    Nt(s)    = Nt(s)    + 1;
end
for r=1:Nr,
    Pr(r) = Pr(r) / totNt;
    for s=1:Ns
        Prs(r,s) = Prs(r,s) / Nt(s);
    end
end
for s=1:Ns
    Ps(s)  = Nt(s) / totNt;
end
%Estimating information
Information = 0;
for s=1:Ns
    for r=1:Nr
        if Prs(r,s)>0
            Information = Information + Ps(s) * Prs(r,s) * log2(Prs(r,s) / Pr(r));
        end
    end
end
```

Fig. 14.1 A simple Matlab algorithm to compute information between a set of stimuli and the spike count of a single neuron. The algorithm was written for the purpose of illustration only; it is aimed to aid readers with limited experience of programming; it was deliberately kept simple and transparent, and was not optimized

and also share a source of stimulus-independent variation: for example, a contribution from a neuromodulatory pathway unrelated to the processing of the whisker stimulation. Suppose that there is also a direct fast excitatory connection between these two cortical neurons. During periods of whisker stimulations, these two example neurons tend fire nearly simultaneously. However, this joint firing has several causes. One is the common stimulus selectivity: the two neurons tend to fire together whenever the preferred whisker is stimulated. However, they could fire together also

because of the direct excitatory interaction between them, or because of the common source of neuromodulatory noise. Ideally, we would need to be able to separate all of these types of contribution to correlations, but in practice this is very difficult. However, it is at least possible to separate the correlations entirely attributable to common or related stimulus preferences ("signal correlations") from the correlations which are not entirely attributable to stimulus preference ("noise correlations"), as follows. In the context of neural coding, the importance of separating noise from signal is, as revealed by theoretical studies (see below), that signal and noise correlations have a radically different impact on the sensory information carried by neural populations. In particular, signal correlations always reduce the information, whereas noise correlations can decrease it, increase it, or leave it unchanged, depending on certain conditions (Abbott and Dayan 1999; Oram et al. 1998; Panzeri et al. 1999; Pola et al. 2003; Schneidman et al. 2003).

In the population coding literature, correlations manifested as covariations in the trial-by-trial fluctuations of correlations at fixed stimulus are traditionally called "noise correlations" (Gawne and Richmond 1993; Averbeck et al. 2006). Because these noise covariations are measured at fixed stimulus, they ignore all effects attributable to shared stimulation. Although we will stick with the well-established "noise" terminology, we point out that the name is potentially misleading: noise correlations can reflect interesting neural effects. For example, they may reflect the presence of a direct connection between two neurons. Mathematically speaking, we say that there are noise correlations if the simultaneous joint response probability $P(\mathbf{r}|s)$ at fixed stimulus is different from the "conditionally independent" response probability in which responses to a given stimulus are statistically independent:

$$P_{\text{ind}}(\mathbf{r}|s) = \prod_{c=1}^{C} P(r_c|s). \tag{14.2}$$

The conditional probability $P_{\text{ind}}(\mathbf{r}|s)$ can be computed by taking the product of the marginal response probabilities of single neurons, as in (14.2), or alternatively by the empirical "shuffling" procedure described in the following. One generates a new set of shuffled responses to stimulus s by randomly permuting, for each neuron, the order of trials collected in response to the stimulus s considered, and then joining together the shuffled responses into a shuffled population response vector $\mathbf{r}_{sh}$. This shuffling operation leaves each single-neuron marginal probability $P(r_c|s)$ unchanged, while destroying any within-trial correlation at fixed stimulus between different neurons. The distribution of $\mathbf{r}_{sh}$ to given stimulus s approximates $P_{\text{ind}}(\mathbf{r}|s)$.

A natural definition for the normalized noise correlation strength of population response $\mathbf{r}$ is the following (Pola et al. 2003):

$$\gamma(\mathbf{r}|s) = \begin{cases} \frac{P(\mathbf{r}|s)}{P_{\text{ind}}(\mathbf{r}|s)} - 1 & \text{if } P_{\text{ind}}(\mathbf{r}|s) \neq 0, \\ 0 & \text{if } P_{\text{ind}}(\mathbf{r}|s) = 0. \end{cases}$$

The noise correlation coefficient $\gamma(\mathbf{r}|s)$ quantifies how much the probability that neurons emit a response $\mathbf{r}$ is higher than that expected in the uncorrelated case, normalized to the probability of response $\mathbf{r}$ expected in the uncorrelated case. Positive

values of this coefficient mean that the individual cell responses happen together during the same trial more frequently than if there was no cross-cell correlation. This correlation coefficient goes beyond second-order (pairwise) correlations, and it takes into account all possible interaction orders between all neurons in the population (Pola et al. 2003).

Correlation that is entirely attributable to common or related stimulus preferences is known as signal correlation (Averbeck et al. 2006; Gawne and Richmond 1993). We also need to introduce a coefficient that quantifies how similar the stimulus modulation of responses of individual cells is. This parameter is important for describing population coding: if all cells have similar stimulus selectivity, it is likely that the population code is redundant. To quantify the similarities of individual cell responses across stimuli, we hence introduce a signal similarity coefficient. In a way analogous to $\gamma(\mathbf{r}|s)$, the signal similarity coefficient is defined as follows:

$$\nu(\mathbf{r}) = \begin{cases} \frac{P_{\text{ind}}(\mathbf{r})}{\prod_c P(r_c)} - 1 & \text{if } \prod_c P(r_c) \neq 0, \\ 0 & \text{if } \prod_c P(r_c) = 0, \end{cases}$$

where

$$P_{\text{ind}}(\mathbf{r}) = \left\langle P_{\text{ind}}(\mathbf{r}|s) \right\rangle_s; \qquad P(r_c) = \left\langle P(r_c|s) \right\rangle_s, \tag{14.3}$$

and for any function of the stimulus $f(s)$, we use the bracket notation to denote its probability-weighted average over the stimulus distribution: $\langle f(s) \rangle \equiv \sum_s P(s) f(s)$. The signal similarity coefficient $\nu(\mathbf{r})$ is different from zero if signals coming from individual neurons are either positively correlated (i.e. similar) or negatively correlated.

14.3.2 The Information Breakdown

Shannon's mutual information, as expressed in (14.1), quantifies the overall information transmitted by the neuronal population activity. However, it tells us little about the specific contribution of correlations. In addition, it does not tell us directly whether correlations make the code redundant or synergistic. In the following we describe a formalism that aims at addressing this limitation of mutual information. This formalism is called the "information breakdown" (Pola et al. 2003) and takes the total mutual information $I(S; \mathbf{R})$ and decomposes it into a number of components, each reflecting a different way into which signal and noise correlation contribute to information transmission. The information breakdown formalism was introduced first in Panzeri et al. (1999), Panzeri and Schultz (2001) in the low spike rate limit and then generalized to arbitrary spike rates in Pola et al. (2003). This is not the only formalism proposed to quantify the contribution of correlation to information (see e.g. Nirenberg et al. 2001; Schneidman et al. 2003; Nakahara and Amari 2002). We decided to focus on the information breakdown formalism partly because it was developed by one of the authors of this chapter,

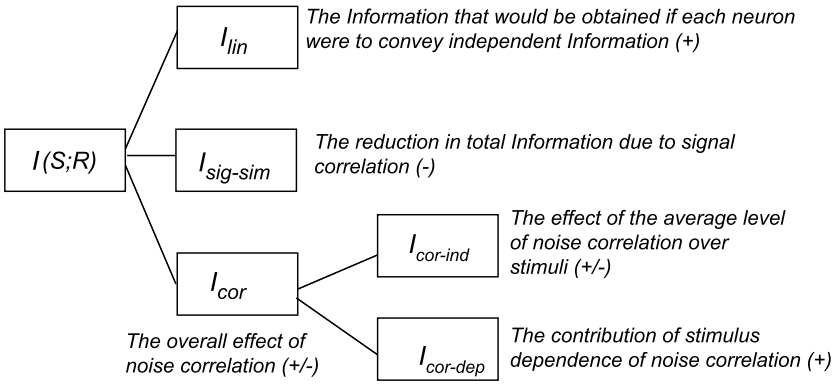

Fig. 14.2 Information component breakdown. The population mutual information $I(S;\mathbf{R})$ can be broken down into a linear component, the reduction of information due to signal correlation, and the contribution of noise correlation (I_{cor}). The noise correlation term can be further broken down to separate out the effect of the average level of noise correlation over all stimuli, and the stimulus-dependence of noise correlation—the latter term captures any effects due to coding by explicit modulation of noise correlation. Modified from Montani et al. (2007)

and partly because it naturally includes most of the quantities proposed by other investigators (Nirenberg et al. 2001; Schneidman et al. 2003).

The information breakdown writes the total mutual information into a sum of components that are related to different ways in which correlations contribute to population coding (Panzeri and Schultz 2001; Pola et al. 2003), as follows:

$$I(S;\mathbf{R}) = I_{\text{lin}} + I_{\text{sig-sim}} + I_{\text{cor}} = I_{\text{lin}} + I_{\text{sig-sim}} + I_{\text{cor-ind}} + I_{\text{cor-dep}}. \quad (14.4)$$

The meaning and mathematical expression of each of the components is summarized in Fig. 14.2 and is described in the following.

14.3.2.1 The Linear Term

The first term of the information breakdown, I_{lin}, gives the total amount of information which would be conveyed if all the cells were sharing neither noise nor signal. In this case, the total information transmitted by the population would just be a linear sum of the information provided by each cell:

$$I_{\text{lin}} = \sum_c I_c, \quad (14.5)$$

where I_c is the information about the stimuli carried by the responses of cell c:

$$I_c = \sum_{s\in S} P(s) \sum_{r_c} P(r_c|s) \frac{\log_2 P(r_c|s)}{P(r_c)}. \quad (14.6)$$

The difference between I_{lin} and $I(S;\mathbf{R})$ is called redundancy. Positive values of redundancy indicate that the population information is less than the sum of the

information provided by each individual neuron. Negative values of redundancy denote the presence of synergistic interaction between neurons, which make the total population information more than the sum of that provided by each individual neuron.

14.3.2.2 The Signal Similarity Term

The amount of redundancy specifically due to signal correlation (i.e. similarity of the stimulus modulation of responses of individual cells) is quantified by the second term of the information breakdown, the signal similarity term $I_{\text{sig-sim}}$, which is defined as follows (Pola et al. 2003):

$$I_{\text{sig-sim}} = \frac{1}{\ln 2} \sum_{\mathbf{r}} \left(\prod_{c} P(r_c) \right) \left\{ \nu(\mathbf{r}) + \left(1 + \nu(\mathbf{r})\right) \ln \frac{1}{1 + \nu(\mathbf{r})} \right\}. \tag{14.7}$$

It can be shown (Pola et al. 2003) that $I_{\text{sig-sim}} \leq 0$ and that $I_{\text{sig-sim}} = 0$ if and only if there are no signal correlations at all (i.e. $\nu(\mathbf{r}) = 0$). This matches intuition: if there are signal correlations but no noise correlations, one would intuitively expect that $I(S; \mathbf{R})$ to be less than I_{lin}. $I_{\text{sig-sim}}$ can be equivalently rewritten as $I_{\text{sig-sim}} = I_{\text{ind}}(S; \mathbf{R}) - I_{\text{lin}}$, where $I_{\text{ind}}(S, \mathbf{R})$ is the information that would be conveyed if there was no noise correlation at all, but the single-neuron marginal response probabilities $P(r_c|s)$ were the same:

$$I_{\text{ind}}(S; \mathbf{R}) = \sum_{s \in S} \sum_{\mathbf{r} \in \mathbf{R}} P(s) P_{\text{ind}}(\mathbf{r}|s) \log_2 \frac{P_{\text{ind}}(\mathbf{r}|s)}{P_{\text{ind}}(\mathbf{r})}. \tag{14.8}$$

It should be noted that the negative of $I_{\text{sig-sim}}$ equals the quantity ΔI_{signal} defined by Schneidman et al. (2003).

14.3.2.3 The Terms Quantifying the Impact of Noise Correlation

The next two terms in the information breakdown, given in what follows, are the correlational terms. They are the only terms that depend on the noise correlation strength $\gamma(\mathbf{r}|s)$. They can be nonzero only if $\gamma(\mathbf{r}|s)$ is different from zero for some response or stimulus. Hence they express any further effects that noise cell correlations might have beyond that accounted for by individual cell properties. These two "correlational" components reflect two different ways in which correlations may contribute to coding (Pola et al. 2003).

The component $I_{\text{cor-ind}}$ reflects the contribution of stimulus-independent correlations and is defined as

$$I_{\text{cor-ind}} = \sum_{\mathbf{r}} \left\langle P_{\text{ind}}(\mathbf{r}|s) \gamma(\mathbf{r}|s) \right\rangle_s \log_2 \frac{1}{1 + \nu(\mathbf{r})}. \tag{14.9}$$

The first multiplicative factor reflects the effect of noise correlation, but these correlations are averaged across stimuli (weighted proportional to the probability of

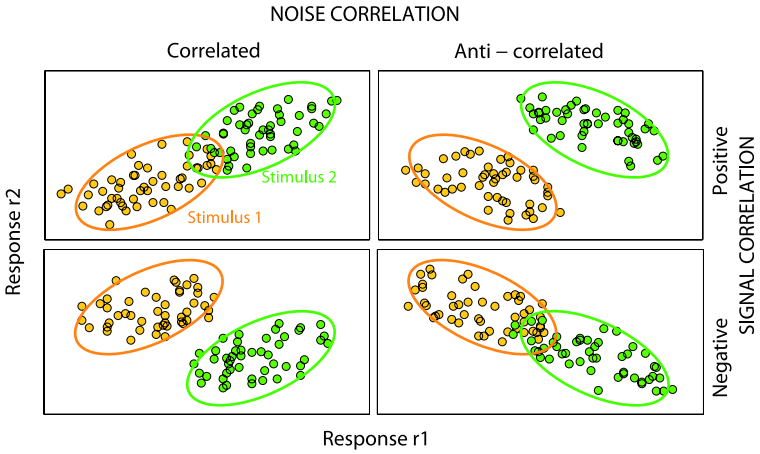

Fig. 14.3 *The effect of stimulus-independent correlations on information encoding. Each panel* sketches joint distributions of responses of two hypothetical cells to two different stimuli (data for stimulus one and two are plotted in *orange* and *green* color, respectively). The *dots* represent a hypothetical scatter plot from single-trial responses to the given stimulus, and each *ellipse* denotes 95% confidence limits. In the *upper panel*, there is positive signal similarity (i.e. individual cell responses to each stimulus are positive correlated), whereas in the *lower panels* there is negative signal similarity. Positive noise correlations correspond to *ellipses* aligned along the diagonal. The more the *ellipses* are elongated, the stronger the noise correlation. The sign of noise correlations between the joint responses differs across columns of this figure (noise correlation is positive in the *left column* and negative in the *right column*). In this figure, noise correlations are stimulus independent equally strong across stimuli (all the *ellipses* within a panel have the same elongation). In general, if noise correlation and signal similarities have opposite signs, the effect of stimulus-independent correlations increases the information about stimuli, because the joint response probabilities to each stimulus become more separated. If instead cross-cell correlation and signal similarities have the same sign, stimuli are less discriminable than in the other case. This intuitively explains the mathematical properties of $I_{\text{cor-ind}}$

each response). The logarithmic term depends instead on signal similarity. In general, if noise and signal correlations have opposite signs, $I_{\text{cor-ind}}$ is positive. In this case, stimulus-independent noise correlations increase stimulus discriminability compared to what it would be if noise correlations were zero (Oram et al. 1998; Pola et al. 2003). If, instead, noise and signal correlations have the same sign, $I_{\text{cor-ind}}$ is negative, and stimuli are less discriminable than the zero noise correlation case. In the absence of signal correlation, $I_{\text{cor-ind}}$ is zero, whatever the strength of noise correlation. For intuitive illustration of these effects, see Fig. 14.3 and (Oram et al. 1998; Petersen et al. 2002a).

The final term of the information breakdown, $I_{\text{cor-dep}}$, is associated with stimulus modulation of correlations

$$I_{\text{cor-dep}} = \sum_{\mathbf{r}} \left\langle P_{\text{ind}}(\mathbf{r}|s)\big(1+\gamma(\mathbf{r}|s)\big)\log_2 \frac{\langle P_{\text{ind}}(\mathbf{r}|s')\rangle_{s'}(1+\gamma(\mathbf{r}|s))}{\langle P_{\text{ind}}(\mathbf{r}|s')(1+\gamma(\mathbf{r}|s'))\rangle_{s'}} \right\rangle_s . \quad (14.10)$$

This term is nonnegative. It is associated with stimulus-dependent correlations because it equals zero if and only if the correlation coefficient $\gamma(\mathbf{r}|s)$ does not

depend on the stimulus s for every response. If a neuronal population carries information by emitting patterns of correlated spikes that 'tag' each stimulus, $I_{\text{cor-dep}}$ is greater than zero. The expression reported above for $I_{\text{cor-dep}}$ was first introduced in Nirenberg et al. (2001) (they gave to this term the name ΔI); Pola et al. (2003) then introduced this equation into the information breakdown. Nirenberg, Latham and colleagues have shown that $I_{\text{cor-dep}}$ is an upper bound to the information lost by a downstream system interpreting the neural responses without taking into account the presence of correlations (Nirenberg et al. 2001; Latham and Nirenberg 2005).

The sum of the two correlational terms $I_{\text{cor-ind}}$ and $I_{\text{cor-dep}}$ is called I_{cor}, and it quantifies the total impact of noise correlation in information encoding. I_{cor}, originally introduced by Hatsopoulos et al. (1998), equals the difference between the information $I(S;\mathbf{R})$ in the presence of noise correlations and the information $I_{\text{ind}}(S,\mathbf{R})$ in the absence of noise correlation. I_{cor} quantifies whether the presence of noise correlations increases or decreases the information available in the neural response, compared to the case where such correlations are absent but the marginal stimulus-conditional response probabilities of all the individual neurons are the same.

14.3.3 Examples of Application of the Information Breakdown to Neural Data

To illustrate the type of insights that could be gained by studying neural population responses using these methods, we briefly review results obtained using the information breakdown in the somatosensory and in the visual pathways.

14.3.3.1 Role of Correlated Firing in Coding Whisker Stimuli

We will begin by describing a study of the population code used by neurons in the rat somatosensory cortex to encode the identity of the deflected whisker (Petersen et al. 2001, 2002b). The whisker representation of the rat somatosensory cortex is known to be organized into anatomically defined columns, arranged in one-to-one correspondence with the whiskers on the rat snout, and containing approximately ten thousand neurons with similar whisker selectivity. To gain insights into how such columnar organization may affect the population code, Petersen and colleagues (Petersen et al. 2001, 2002b) used the information breakdown formalism to compute the information about the location of a rapidly deflected whisker and studied the differences in the coding mechanisms employed by pairs of neurons located either in the same column or in different columns. When pairs of neurons were both located in the same column, they found that the stimulus-independent correlational component $I_{\text{cor-ind}}$ was negative and appreciable in size ($I_{\text{cor-ind}}$ was approximately 10% of the total information). The reason why this component was negative was

that neurons within the same column tended to have similar stimulus selectivity (i.e. positive signal correlation) and also positive noise correlations. When considering pairs of neurons located in different columns, $I_{\text{cor-ind}}$ became very close to zero. This is because neurons in different columns have different stimulus preferences (weak signal correlation) and also very weak noise correlations. This result suggests that cross-neuronal correlations limit the information encoding capacity of single cortical columns, but they influence less the information encoding capabilities obtained when putting together several columns.

The contribution of stimulus-dependent noise correlations $I_{\text{cor-dep}}$ was very small both for same-column and different-column pairs. Given that $I_{\text{cor-dep}}$ is an upper bound to the information lost by a downstream system interpreting the neural responses without taking into account the presence of correlations (Latham and Nirenberg 2005), these results suggest that noise correlations in somatosensory cortex can be ignored, at only a little information cost, during decoding.

14.3.3.2 Role of Correlations in Encoding Stimulus Contrast in Monkey Visual Cortex

The information breakdown has also been applied to the study of how groups of neurons in primary visual cortex (V1) encode contrast and direction of motion of visual stimuli. To address this question, Kohn and colleagues (Kohn and Smith 2005) recorded, from area V1 of anesthetized macaques, responses of pairs of neurons to drifting sinusoidal gratings of different directions and contrasts. Using cross-correlation measures, they showed that an orientation-sensitive component of the response correlation dominated at short time scales, and an orientation-insensitive component was prevalent at longer time scales. In a subsequent analysis of the same data, Montani and colleagues (Montani et al. 2007) used the information breakdown to gain more quantitative insights into the role of noise correlations in the encoding of contrast and orientation. Fig. 14.4 summarizes the results, averaged over the whole population, of the breakdown of information about direction carried by pairs of neurons (Montani et al. 2007). For this analysis, the neural response of each pair was quantified as the spike count carried over a post-stimulus window of fixed length. Different window lengths (in the range 2–80 ms) were used. The contribution of the information components was strongly dependent on the length of the response window: while the negative contribution of $I_{\text{cor-ind}}$ grew with the time window, the effects of the other components follow a 'U' or inverted 'U' curve (Fig. 14.4). For time windows shorter than 10 ms, the temporally precise noise correlations (Kohn and Smith 2005) led to a large contribution from correlational components (Fig. 14.4). At time windows longer than 10 ms the contribution from $I_{\text{cor-dep}}$ was significantly curtailed and was effectively canceled by the signal similarity term, $I_{\text{sig-sim}}$, and also by $I_{\text{cor-ind}}$, which provided a greater negative contribution as the time window increased (Fig. 14.4). Since the various components had different signs, their total contribution to information almost canceled out, and the total information $I(S; R)$ was roughly equal to the linear term I_{lin}. Thus, the linear

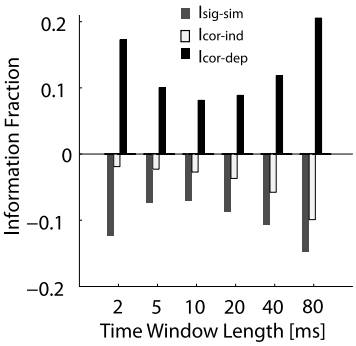

Fig. 14.4 The breakdown of the information about stimulus direction carried by pairs of V1 neurons. *Bar charts* report the fraction of the mutual information accounted for by the different components of the information breakdown plotted as functions of the length of the post-stimulus time window used to compute the spikes. Modified from Montani et al. (2007)

approximation was relatively accurate even despite the presence of substantial and very informative noise correlations. The additional information they provided was approximately offset by the redundancy arising from the similar tuning properties of the two cells. A potential function of this cancelation may be that of allowing cortical circuits to enjoy the stability provided by having similarly tuned neurons without suffering the penalty of redundancy (Montani et al. 2007). The coding of contrast was far more redundant than that of direction at all time windows considered (Montani et al. 2007).

Other studies have applied some or all of the information component analysis discussed above to simultaneously recorded pairs of neurons from visual cortex (Golledge et al. 2003; Montani et al. 2007; Rolls et al. 2003) or from the retina (Nirenberg et al. 2001). A result which is highly consistent across all such experiments is that $I_{\text{cor-dep}}$ is always relatively small, in most cases much less than 10% of the total mutual information. A caveat to all work done on pairs is that in principle, it may be possible for correlations to exert a weak effect at the level of neuron pairs yet a strong one at the level of larger populations. A key challenge is to be able to extend the analysis to much larger populations of cells and characterize how the contribution of correlation scales with population size.

14.4 Studying the Information Content Through Decoding Algorithms

When the neuronal population is large, the number of possible responses $\mathbf{r}$ is very high. As a consequence, it becomes extremely difficult to sample the response probabilities $P(\mathbf{r}|s)$ from the relatively small number of experimental trials that is typically possible to collect from an experimental session. The poor sampling of the response probabilities of large populations eventually leads to a severe systematic error ("bias") in the estimation of mutual information (Panzeri et al. 2007). Although techniques are available to estimate and subtract out the bias, they usually work well only for relatively small populations (Panzeri et al. 2007). This has restricted most information analyses of population codes to the study of pairs or of

triplets of simultaneously recorded neurons (see previous section). How best to extend the information approach to larger populations remains an open and debated question.

A promising approach to the information analysis of larger populations is the use of information theory coupled to decoding approaches (Quian Quiroga and Panzeri 2009). These procedures use a stimulus-decoding procedure (see Chap. 15) to predict the most likely stimulus elicited from a single-trial population response, and this makes it possible to compress the response space into the 'predicted stimulus' (Quian Quiroga and Panzeri 2009). If the number of stimuli is much smaller than the number of responses, stimulus-decoding is an effective and simple way to reduce the space of responses.

In more detail, this approach works as follows. In each trial, a stimulus s is presented and a stimulus s^P is predicted by a decoding algorithm, and the corresponding probability $P(s^P|s)$ (of predicting stimulus s^P when stimulus s is presented) is computed. The "decoded" information $I(S; S^P)$ is then quantified as follows:

$$I\left(S; S^P\right) = \sum_s \sum_{s^P} P(s) P\left(s^P|s\right) \log_2 \frac{P(s^P|s)}{P(s^P)}. \tag{14.11}$$

Information-theoretic inequalities ensure that $I(S; S^P) \leq I(S; \mathbf{R})$ (Quian Quiroga and Panzeri 2009). To validate decoding results, some trials can be used to optimize the decoder (the training set) and the rest to testing its performance, a procedure called cross-validation (Quian Quiroga and Panzeri 2009). It is important that trials belonging to the training set are not used to evaluate the decoding performance because this may lead to artificial high values due to overfitting. A common procedure is the "leave-one-out" validation, in which each trial is predicted based on the distribution of all the other trials. This has the advantage that both optimization and assessment testing are based on the largest possible number of trials (Quian Quiroga and Panzeri 2009).

It is interesting to consider how to evaluate the role of correlations using the decoding step. As discussed above, a simple but effective way to destroy noise correlations is the shuffling procedure described earlier. The shuffling procedure can be applied to the evaluation of population information through decoding in the following way. When trying to understand whether the presence of correlations decreases or increases information (a question addressed using I_{cor} within the information breakdown), the correct procedure is to shuffle both the training and the test dataset, as done e.g. in Quian Quiroga et al. (2007). When trying to evaluate whether any information is lost when interpreting correlated data with a decoding algorithm ignoring correlations (a question addressed using I_{cor} within the information breakdown), the appropriate procedure is to shuffle only the training data while leaving the test data untouched.

14.5 Pooling as a Strategy for Population Coding

The previous sections concentrated on one particular question about population codes: the role of correlations in the representation of sensory information. How-

ever, the use of stimulus-related correlations is only one of several possibilities that have been proposed for reducing the ambiguity of the noisy messages carried by individual neurons. It has been suggested that another strategy that neural populations may use to reduce the variability of single-neuron messages is that of "pooling" together the responses of the neural population, by summing the spikes from different neurons into a single, pooled population response (Darian-Smith et al. 1973; Zohary et al. 1994; Shadlen and Newsome 1998). If all neurons in the population are similarly tuned to the stimulus, this strategy is effective at reducing (by simple averaging) the trial to trial noise of each neuron. The noise reduction by pooling is particularly effective if the neurons are not correlated or are only very weakly correlated (Darian-Smith et al. 1973; Zohary et al. 1994). However, pooling poses also a problem: if different neurons in the population respond in a varied manner to the stimuli (for example, each neuron may fire only to a very specific and small subset of stimuli Quian Quiroga et al. 2005), then reading out the population information requires to keep track of which neuron fired each spike (a labeled-line code Reich et al. 2001), and pooling would destroy the information carried by the population. In practice, whether or not pooling is an effective scheme depends on the balance between the benefits of averaging away noise and of the disadvantages of wiping away the differences in the stimulus selectivity of each neuron.

One way to address quantitatively the question of how good pooling is as a candidate scheme for population coding is to compare the overall information available in the population response (14.1) to the information available in the pooled response $\sigma = \sum_{c=1}^{C} r_c$, which is defined as follows:

$$I_{\text{pooled}}(S, \mathbf{\Sigma}) = \sum_{s,\boldsymbol{\sigma}} P(s) P(\boldsymbol{\sigma}|s) \log_2 \left(\frac{P(\boldsymbol{\sigma}|s)}{P(\boldsymbol{\sigma})} \right). \tag{14.12}$$

This analysis has been applied to population recordings from visual, somatosensory, and auditory cortices (Reich et al. 2001; Panzeri et al. 2003; Kayser et al. 2009). In one such study, Panzeri et al. (2003) considered the encoding of stimulus location and analysed small populations of up to three neurons simultaneously recorded in the whisker representation of the rat somatosensory cortex. Panzeri et al. (2003) found that for populations of neurons located in the same column, pooling was effective at reducing noise while preserving the full information content of the population. For same-column neurons, the information in the pooled code was equal to the one in the full population responses. This also means that differences in the response properties of neurons located within the same column are not essential for representing salient stimulus features (Panzeri et al. 2003). Thus, in rat somatosensory cortex the columnar organization could act as a framework to facilitate pooling with minimal information loss. However, pooling neurons belonging to different columns led to large information losses (Panzeri et al. 2003). Thus, pooling was a viable scheme, but only on spatially restricted populations. Other coding schemes have to be used to bring together the information carried by distant populations.

14.6 Software Implementation

A software implementation of most of these algorithms is described in Magri et al. (2009) and can be found at http://www.ibtb.org/ or http://www.apst.spiketrain-analysis.org/.

14.7 Conclusions

We have described an information-theoretic approach to the quantitative study of neural population codes. An advantage of this approach is that it allows a detailed characterization of how different aspects of neuronal population activity may contribute to transmission of information on a single trial basis. A limitation of this approach is that accurate calculations of information are difficult with neuronal populations because of large number of trials needed to sample the population response probabilities (Panzeri et al. 2007). Recent progress on the statistical estimation techniques now permits the computation of the information carried by populations of up to five–ten neurons (Panzeri et al. 2007). This is useful to get some insights into the details of information processing in some local networks. A major and important challenge for computational neuroscientists is to find ways to further extend the feasibility of performing information computations with large populations (Quian Quiroga and Panzeri 2009). This step would be crucial to enable us to gain better insights into how information is processed by distributed networks and would be an important analytical complement to the experimental progress in recording technology, which already enables the observation of the activity of several tens of neurons (Csicsvari et al. 2003; Kerr and Denk 2008).

Acknowledgements We are grateful to R Quian Quiroga, M Maravall, MA Montemurro and SR Schultz for many useful discussions and interactions. This research was conducted under the framework of the BMI Project at the Department of Robotics, Brain and Cognitive Sciences of the Italian Institute of Technology. RSP was supported by the EPSRC CARMEN grant.

References

Abbott LF, Dayan P (1999) The effect of correlated variability on the accuracy of a population code. Neural Comput 11:91–101

Averbeck BB, Latham PE, Pouget A (2006) Neural correlations, population coding and computation. Nat Rev Neurosci 7(5):358–366

Barlow HB, Hill RM, Levick WR (1964) Retinal ganglion cells responding selectively to direction and speed of image motion in the rabbit. J Physiol (London) 173:377–407

Csicsvari J, Henze D, Jamieson B, Harris K, Sirota A, Bartho P, Wise K, Buzsaki G (2003) Massively parallel recording of unit and local field potentials with silicon-based electrodes. J Neurophysiol 90(2):1314–1323

Darian-Smith I, Johnson KO, Dykes R (1973) Cold fiber population innervating palmar and digital skin of the monkey: responses to cooling pulses. J Neurophysiol 36(2):325–346

Engel AK, Singer W (2001) Temporal binding and the neural correlates of sensory awareness. Trends Cogn Sci 5(1):16–25

Gawne T, Richmond BJ (1993) How independent are the messages carried by adjacent inferior temporal cortical neurons?. J Neurosci 13:2758–2771

Golledge HD, Panzeri S, Zheng F, Pola G, Scannell JW, Giannikopoulos DV, Mason RJ, Tovee MJ, Young MP (2003) Correlations, feature-binding and population coding in primary visual cortex. Neuroreport 14(7):1045–1050

Hatsopoulos NG, Ojakangas C, Paninski L, Donoghue JP (1998) Information about movement direction obtained from synchronous activity of motor cortical neurons. Proc Natl Acad Sci 95(26):15706–15711

Kayser C, Montemurro MA, Logothetis N, Panzeri S (2009) Spike-phase coding boosts and stabilizes information carried by spatial and temporal spike patterns. Neuron 61:597–608

Kerr JN, Denk W (2008) Imaging in vivo: watching the brain in action. Nat Rev Neurosci 9:195–205

Kohn A, Smith MA (2005) Stimulus dependence of neuronal correlation in primary visual cortex of the macaque. J Neurosci 25(14):3661–3673

Latham PE, Nirenberg S (2005) Synergy, redundancy, and independence in population codes, revisited. J Neurosci 25(21):5195–5206

Li C-L (1959) Synchronization of unit activity in the cerebral cortex. Science 129:783–784

Magri C, Whittingstall K, Singh V, Logothetis NK, Panzeri S (2009) A toolbox for the fast information analysis of multiple-site LFP, EEG and spike train recordings. BMC Neurosci 10:81

Mastronarde DN (1983) Correlated firing of cat retinal ganglion cells: spontaneously active input to x- and y-cells. J Neurophysiol 49:303–324

Montani F, Kohn A, Smith MA, Schultz SR (2007) The role of correlations in direction and contrast coding in the primary visual cortex. J Neurosci 27(9):2338–2348

Nakahara S, Amari S (2002) Information geometric measures for neural spikes. Neural Comput 14:2269–2316

Nirenberg S, Carcieri SM, Jacobs AL, Latham PE (2001) Retinal ganglion cells act largely as independent encoders. Nature 411:698–701

Oram MW, Foldiak P, Perrett DI, Sengpiel F (1998) The 'ideal homunculus': decoding neural population signals. Trends Neurosci 21(6):259–265

Panzeri S, Petroni F, Petersen RS, Diamond ME (2003) Decoding neuronal population activity in rat somatosensory cortex: role of columnar organization, cerebral cortex. Cereb Cortex 13(1):45–52

Panzeri S, Schultz SR (2001) A unified approach to the study of temporal, correlational, and rate coding. Neural Comput 13:1311–1349

Panzeri S, Schultz SR, Treves A, Rolls ET (1999) Correlations and the encoding of information in the nervous system. Proc R Soc Lond B Biol Sci 266:1001–1012

Panzeri S, Senatore R, Montemurro MA, Petersen RS (2007) Correcting for the sampling bias problem in spike train information measures. J Neurophysiol 98:1064–1072

Perkel DH, Gerstein GL, Moore GP (1967) Neuronal spikes trains and stochastic point processes II. Simultaneous spikes trains. Biophys J 7:419–440

Petersen RS, Panzeri S, Diamond ME (2001) Population coding of stimulus location in rat somatosensory cortex. Neuron 32:503–514

Petersen RS, Panzeri S, Diamond ME (2002a) Population coding in somatosensory cortex. Curr Opin Neurobiol 12:441–447

Petersen RS, Panzeri S, Diamond ME (2002b) The role of individual spikes and spike patterns in population coding of stimulus location in rat somatosensory cortex. Biosystems 67:187–193

Pola G, Thiele A, Hoffmann KP, Panzeri S (2003) An exact method to quantify the information transmitted by different mechanisms of correlational coding. Network 14:35–60

Quian Quiroga R, Panzeri S (2009) Extracting information from neural populations: information theory and decoding approaches. Nat Rev Neurosci 10:173–185

Quian Quiroga R, Reddy L, Kreiman G, Koch C, Fried I (2005) Invariant visual representation by single neurons in the human brain. Nature 435:1102–1107

Quian Quiroga R, Reddy L, Koch C, Fried I (2007) Decoding visual inputs from multiple neurons in the human temporal lobe. J Neurophysiol 98(4):1997–2007
Reich DS, Mechler F, Victor JD (2001) Independent and redundant information in nearby cortical neurons. Science 294:2566–2568
Rolls ET, Franco L, Aggelopoulos NC, Reece S (2003) An information theoretic approach to the contributions of the firing rates and the correlations between the firing of neurons. J Neurophysiol 89:2810–2822
Salinas E, Sejnowski TJ (2001) Correlated neuronal activity and the flow of neural information. Nat Rev Neurosci 2(8):539–550
Schneidman E, Bialek W, Berry MJI (2003) Synergy, redundancy, and independence in population codes. J Neurosci 23(37):11539–11553
Shadlen MN, Movshon JA (1999) Synchrony unbound: a critical evaluation of the temporal binding hypothesis. Neuron 24(67–77):111–125
Shadlen MN, Newsome WT (1998) The variable discharge of cortical neurons: implications for connectivity, computation and information coding. J Neurosci 18:3870–3896
Singer W, Gray CM (1995) Visual feature integration and the temporal correlation hypothesis. Annu Rev Neurosci 18:555–586
Strong S, Koberle R, de Ruyter van Steveninck R, Bialek W (1998) Entropy and information in neural spike trains. Phys Rev Lett 80:197–200
von der Malsburg C (1999) The what and why of binding: the modeler's perspective. Neuron 24(1):95–104
Zohary E, Shadlen M, Newsome W (1994) Correlated neuronal discharge rate and its implications for psychophysical performance. Nature 370:140–143

Chapter 15
Stochastic Models for Multivariate Neural Point Processes: Collective Dynamics and Neural Decoding

Wilson Truccolo

Abstract This chapter reviews a stochastic point process framework for the modeling, analysis and decoding of neuronal ensembles. The spiking probability of any neuron in an ensemble is computed recursively via a system of stochastic nonlinear equations with delays in discrete time. These equations are expressed in terms of conditional intensity functions, which capture the effects of the neuron's own spiking history (intrinsic dynamics), ensemble history (collective dynamics), and dependencies on stimuli and behavioral covariates. Four related approaches for the statistical modeling of conditional intensity functions are presented: generalized linear models (GLM), penalized splines, hierarchical Bayesian P-splines, and nonparametric function approximation. Decoding of neuronal ensemble spike trains is implemented via stochastic state-space models with point process observations. The framework is illustrated with examples of neural decoding of hand velocities and assessment of collective dynamics in primary motor cortex.

15.1 Introduction

Recent advances in microelectrode array recordings allow us to measure the simultaneous spiking activities of hundreds of neurons: in the hippocampus (Wilson and McNaughton 1993; Harris et al. 2003), retina and primary visual cortex (Pillow et al. 2008; Jermakowicz et al. 2009), and cortical sensorimotor areas in behaving nonhuman primates (Hatsopoulos et al. 1998; Nicolelis et al. 2003; Riehle et al. 1997), and humans (Hochberg et al. 2006; Truccolo et al. 2008a, 2008b; Truccolo et al. 2009). These data open the window to addressing how coordinated

W. Truccolo (✉)
Department of Neuroscience and Brown Institute for Brain Science, Brown University, S.E. Frank Hall, 185 Meeting Street, Office 583, Providence, RI 02912, USA
Department of Neurology, Massachusetts General Hospital, Boston, MA 02114, USA
e-mail: wilson_truccolo@brown.edu
url: http://donoghue.neuro.brown.edu/people/WTruccolo/WTruccoloIndex.html

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_15, © Springer Science+Business Media, LLC 2010

activity in neuronal ensembles relates to representation, computation, physiological states, and dynamics in the brain. These questions are central not only to theoretical neuroscience, but also to translational neuroscience, in particular the new emerging field of neuroengineering and brain-machine interfaces (Hochberg et al. 2006; Donoghue 2008; Truccolo et al. 2008a, 2008b; Nicolelis and Lebedev 2009). Until recently, most approaches to neural encoding, decoding, and collective dynamics of spike train data were not directly applied to the neural point process itself, i.e., sequences of spike times, but to smoothed versions of the spike trains or to spike counts within relatively large time bins. In this chapter we focus on stochastic models and decoding approaches that preserve the point process nature of neuronal spiking. We take advantage of the conditional intensity function and the general likelihood for discrete time point processes. Before we address the neural decoding and collective dynamics of neuronal ensembles, basic concepts and terminology are introduced, followed by a framework for the modeling of conditional intensity functions.

Consider an ensemble of C neurons whose recorded individual spike times are denoted by

$$0 < u_1^c < u_2^c < \cdots < u_{J_c}^c \leq T,$$

$c = 1, \ldots, C$. These spike times induce counting processes, $N^c(t)$, which express the number of spikes up to time t. We would like to characterize how these spiking activities relate to the neuron's own spiking history (intrinsic dynamics), the spiking history of other neurons in the recorded ensemble, and other covariates of interest. We will denote the spiking history of the entire ensemble up to (but not including) time t as $\mathcal{H}_t$. In addition, $x_{0:t}$ will denote measured covariates in the same time interval. These covariates could include discrete and continuous stimuli or behavioral variables, and also other quantities related to physiological variables and neural states (e.g., spectral power in different field potential frequency bands). The case where $x_{0:t}$ may also include hidden variables has been considered in Yu et al. (2006), Kulkarni and Paninski (2007), Brockwell et al. (2007), Lawhern et al. (2010). Our goal is to construct stochastic models where the spiking probability in a given small enough time interval $[t, t + \Delta)$ for each neuron in the ensemble is recursively computed based on spiking history and covariates. This probability, denoted by

$$\Pr\big(N^c(t + \Delta) - N^c(t) = 1 \big| \mathcal{H}_t, x_{0:t}\big),$$

will be fundamental to our approach to decoding and to our study of collective dynamics in neuronal ensembles. The time evolution of a neuronal ensemble will be represented by these probabilities, which will be computed recursively via a system of stochastic nonlinear equations with delays. The state-space approach for neural decoding in Sect. 15.3 will explicitly require the computation of these probabilities. These models will be time "causal" in $\mathcal{H}_t$, but not necessarily in x. For example, when modeling sensorimotor cortical neurons, we might want to examine the statistical relationship between spiking at time t and some covariate at multiple time offsets $t + \tau$ for negative and positive lags, such as hand kinematics preceding and

following spiking activity. Although this issue should be kept in mind, we will put it aside for notational convenience.

The conditional intensity function and the general likelihood function for point processes will be key in our goal of modeling the spiking probability. The conditional intensity function for a given univariate point process is defined as

$$\lambda(t|\mathcal{H}_t, x_{0:t}) \triangleq \lim_{\Delta \to 0+} \frac{P(N(t+\Delta) - N(t) = 1|\mathcal{H}_t, x_{0:t})}{\Delta}. \tag{15.1}$$

This function is a history-dependent generalization of the inhomogeneous Poisson process intensity function and completely specifies any point process that admits a conditional intensity function representation (Daley and Vere-Jones 2003). Based on this definition, the recursive spiking probability of a given neuron can then be computed as

$$\Pr\big(N(t+\Delta) - N(t) = 1 \big| \mathcal{H}_t, x_{0:t}\big) = \lambda(t|\mathcal{H}_t, x_{0:t})\Delta + o(\Delta). \tag{15.2}$$

A discrete-time version of the process can be obtained by partitioning the time interval $(0, T]$ into $k = 1, \ldots, K$ subintervals $(t_k - 1, t_k]$ of length $\Delta = T K^{-1}$ such that at most one spike is observed in

$$\Delta N(t_k) = N(t_k) - N(t_{k-1}). \tag{15.3}$$

Due to the refractory period in neurons, a typical choice in practical applications is such that $\Delta \leq 1$ ms. Let $\lambda_k = \lambda(t_k|\mathcal{H}_k, x_{1:k})$, $\Delta N_k = \Delta N(t_k)$, $\mathcal{H}_k = \{\Delta N^c_{1:k-1}\}^C_{c=1}$, and $x_k = x(t_k)$. The discrete-time point process likelihood for a sequence of J spikes of a particular neuron corresponds to

$$\mathcal{L}(N_{1:K}; \mathcal{H}_K, x_{1:K}) = \exp\left\{\sum_{k=1}^{K} \log(\lambda_k \Delta)\Delta N_k - \lambda_k \Delta\right\} + o\big(\Delta^J\big). \tag{15.4}$$

(See Truccolo et al. 2004 for a derivation.) This distribution belongs to the exponential family with natural or canonical parameter $\log(\lambda_k \Delta)$. The problem we are concerned with is therefore the statistical modeling or approximation of

$$\log(\lambda_k \Delta) = F(\mathcal{H}_k, x_{1:k}), \tag{15.5}$$

via methods that will use the discrete time likelihood in (15.4). We will focus on "direct" models of the conditional intensity, rather than on deriving it from modeled interspike time interval (ISI) densities (e.g., Chap. 1; Barbieri et al. 2001). In addition to (15.2), note that (15.4) uses a second form of the single-neuron conditional spiking probability

$$\Pr(\Delta N_k|\mathcal{H}_k, x_{1:k}) = \exp\big\{\Delta N_k \log(\lambda_k \Delta) - \lambda_k \Delta\big\} + o(\Delta), \tag{15.6}$$

which will also be employed in the next sections.

15.2 Estimation of Conditional Intensity Functions

We review parametric, semiparametric, and nonparametric approaches, all of which take advantage of the discrete-time point process likelihood function in order to model the conditional intensity function of each neuron in a recorded ensemble.

15.2.1 Generalized Linear Models

We start by considering the following class of models:

$$\log\left[\hat{\lambda}(t_k|z_k,\theta)\Delta\right] = \mu_0 + \sum_i \beta_i F_i(z_{i,k}) + \sum_{i,j} \beta_{i,j} F_{i,j}(z_{i,k}, z_{j,k}) + \cdots, \quad (15.7)$$

where $\hat{\lambda}(t_k|z_k,\theta)$ is the estimated conditional intensity function, $z_k = \{\mathcal{H}_k, x_{1:k}\}$ ($z_{i,k}$ corresponds to a specific covariate), $\theta = \{\mu_0, \beta_i, \beta_{ij}, \ldots\}$ are model parameters, and $\{F_i\}$ and $\{F_{i,j}\}$ are general functions. In particular, $\{F_{i,j}\}$ corresponds to functions that capture second-order interaction effects between covariates. Higher-order terms can also be included. Even though the conditional intensity function in the log domain is linear in the parameters, the covariates can enter the model in any nonlinear form. Under this formulation of the conditional intensity function, the log-likelihood in (15.4) has the same form as the log-likelihood function for a GLM under a Poisson distribution and log link function (McCullagh and Nelder 1989; Truccolo et al. 2004). For small enough Δ, (15.4) can also be expressed in terms of a log-likelihood function for a GLM under a Bernoulli probability and logistic link function (Truccolo et al. 2004; Brillinger 1988). The GLM framework offers a class of very flexible models. For example, one can implement neural system identification via Volterra kernels (Marmarelis 2004) by choosing appropriate functions (e.g., Laguerre basis functions) and time lags (temporal memory) for the covariates. Neural point process models in the GLM class have been explored in Brillinger (1988), Kass and Ventura (2001), Paninski (2004), Truccolo et al. (2004), Harris et al. (2003), Pillow et al. (2008). The linear–nonlinear-Poisson (LNP) cascade model (Paninski 2004) is a particular case in this class.

Generalized linear models can be readily fit via maximum likelihood. If the maximum likelihood (ML) exists, it is unique (Wedderburn 1976). Conditions for uniqueness of solutions for certain conditional intensity models in noncanonical form have been presented in Paninski (2004). However, even if the ML solution exists, there are potential problems when collinearity in the covariate (design) matrix or in the Fisher information matrix is present (Lesaffre and Marx 1993). In this case, solutions can become inaccurate, and the estimated variance of model's parameters, based on the observed Fisher information matrix, tends to be inflated. Some statistical packages will by default solve the optimization problem in the subspace spanned by a subset of ranked covariates and dimension specified by the rank of the covariate matrix. Although this approach is computationally convenient, highly explanatory

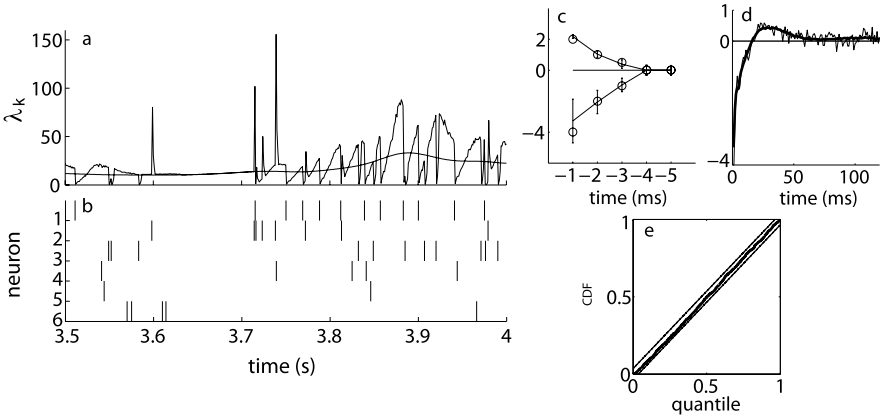

Fig. 15.1 Simulation example. (**a**) The conditional intensity function given by (15.8) (in spikes/s, $\Delta = 1$ ms, $T = 200$ s, $C = 6$) and (**b**) the corresponding simulated point process for neuron 1. The *smoother curve* in (**a**) corresponds to the modulation due exclusively to the hand velocity. Excitatory and inhibitory inputs to neuron 1 consisted of the past 3-ms spiking histories of neurons 2 and 3, respectively. (**c**) The corresponding parameters for the inputs from these two neurons were correctly estimated from the 6-neuron simulated data. (**d**) True and estimated coefficients for the intrinsic history component. The *thicker curve* in (**d**) corresponds to the true coefficients. Negative values following a spike relate to refractory and recovery periods. (**e**) The estimated conditional intensity passed the K–S goodness-of-fit test based on the time-rescaling theorem (Brown et al. 2001). (Modified with permission from Truccolo et al. 2004.)

covariates are often left out of the estimated model (Lesaffre and Marx 1993). Different alternatives to handling collinearity problems include other dimensionality reduction techniques and regularization. Efficient algorithms for L1 (Lasso) and L2 (ridge regression) regularization in the context of GLMs have been presented by Friedman et al. (2008). (See also penalized splines below.) Zhao and Iyengar (2010) examined convergence problems due to complete or quasi-complete data separation in the case of logistic link functions, or due to bursty behavior or particular choices of time discretization in the case of log link functions. In most statistical packages (e.g., functions glmfit and glm in Matlab and R, respectively), maximization of the log-likelihood function is typically implemented via iterative reweighted least squares (IRLS) algorithms. Direct maximization via gradient methods is also a common choice (Friedman et al. 2008). Which approach performs numerically better is still an issue being examined.

For concreteness, Fig. 15.1 shows a simple simulation example where the conditional intensity function of a modeled M1 neuron is given by

$$\log\left[\hat{\lambda}^c(t_k|z_k,\theta)\Delta\right] = \mu^c + \sum_{i=1}^{C}\sum_{q=1}^{\mathcal{O}} \alpha_{i,q}^c \Delta N_{k-q}^i + \beta_1^c x_{1,k+\tau} + \beta_2^c x_{2,k+\tau}, \quad (15.8)$$

with model parameters $\theta = \{\mu^c, \alpha_{i,q}^c, \beta_1^c, \beta_2^c\}$. The parameter μ relates to a background level of activity, and $x_{k+\tau}$ is the two-dimensional hand velocity at a single

time lag corresponding to 150 ms. This model can be easily extended by including hand velocities at multiple time lags (Hatsopoulos et al. 2007) and by letting these velocities enter the model via more complex, nonlinear functions. The double summation term captures the effects of the intrinsic and ensemble spiking histories. The order of the history component $\mathcal{O}$ can be estimated via model comparison (e.g., Akaike Information Criterion or cross-validation). In this simple model, previous spikes act as "impulses" on temporal filters defined by the $\{\alpha\}$ coefficients. Related models for temporal filters, estimated via basis functions, have been presented in Pillow et al. (2008). Spiking history effects can also be modeled in terms of the elapsed times since past spikes, as, for example, in the inhomogeneous Markov interval process model (Kass and Ventura 2001). See Chap. 1 for an overview of stochastic neural point processes with correlated ISIs. Johnson (1996) presents an interesting discussion about single-neuron intrinsic dynamics models based on renewal, Markov and Hawkes' point processes, and their relationship to the classic Hodgkin–Huxley model. A review of stochastic point process models, diffusion and continuous-state hidden Markov models (HMM) in the context of integrate-and-fire neurons was presented by Paninski et al. (2009b).

15.2.2 Penalized Generalized Linear Models and Penalized B-Splines

As the number of parameters increases and the general functions become more and more flexible in the model (15.7), it also becomes important to prevent overfitting and preserve generalization of the model. This can be achieved by adding a penalty term $\mathcal{P}$ to the log-likelihood function

$$\log \mathcal{L}^* = \log \mathcal{L} - \eta \mathcal{P}, \tag{15.9}$$

where η is a regularization parameter. A modified version of the IRLS algorithm or gradient methods can be used to maximize this penalized log-likelihood function. Computationally efficient and flexible models in this approach can be implemented via penalized B-splines. B-splines have the important property of being local basis functions that look like Gaussians (Hastie et al. 2001). Reformulate (15.7) as

$$\log\left[\hat{\lambda}(t_k|z_k,\theta)\Delta\right] = \sum_i F_i(z_{i,k}) + \sum_{i,j} F_{i,j}(z_{i,k}, z_{j,k}) + \cdots + z'_k\gamma, \tag{15.10}$$

where $z'_k\gamma$ is the strictly linear part of the model. Eilers and Marx (1996) assume that F_i and $F_{i,j}$ are smooth and can be approximated by B-splines and tensor product B-splines, respectively. In particular, F_i is approximated by M_i B-splines of degree l (usually cubic) with n_i equally spaced knots

$$F_i(z_{i,k}) = \sum_{m=1}^{M_i} \beta_{i,m} B_{i,m}(z_{i,k}), \tag{15.11}$$

where $M_i = n_i + l$, $B_{i,m}(z_{i,k})$ is the projection of $z_{i,k}$ onto the mth basis function, and the $\{\beta_{i,m}\}$ are parameters to be estimated. A relatively large number of knots

(e.g., 20 to 40) is chosen to ensure enough flexibility. Because roughness is controlled by a penalty term, once a minimum number of knots is reached, the fit given by a penalized spline should be almost independent of the knot number and location. Typically, the penalty term is based on squared rth-order differences applied to adjacent B-spline coefficients. The penalty term, for a model containing only F_i terms, corresponds to

$$\mathcal{P} = \frac{1}{2} \sum_i \beta_i' H_j^r \beta_i, \tag{15.12}$$

where $H_j^r = (D_j^r)' D_j^r$, and D_j^r corresponds to the matrix representation of the difference operator of order r. If $r = 0$, the regularization corresponds to ridge regression with B-splines. A common choice is to use second-order differences. Wood (2006) presents extensions of this approach for additive models, and a software package (MGCV) is available for R.

15.2.3 Hierarchical Bayesian P-Spline Models

Penalized splines can be readily extended into a Bayesian framework. In Brezger and Lang (2006), second-order differences are replaced with their stochastic analogues, i.e., second-order random walks defined by

$$\beta_{i,m} = 2\beta_{i,m-1} - \beta_{i,m-2} + \varepsilon_{i,m}, \tag{15.13}$$

with Gaussian error $\varepsilon_{i,m} \sim \mathcal{N}(0, \sigma_i^2)$ and diffuse priors $\beta_{i,1}, \beta_{i,2} \propto \text{const}$ for initial values. While first-order random walks penalize abrupt jumps $\beta_{i,m} - \beta_{i,m-1}$ between successive spline parameters, second-order random walks penalize deviations from the linear trend $2\beta_{i,m-1} - \beta_{i,m-2}$. The amount of smoothing is controlled by a parameter σ_i^2, which corresponds to the inverse smoothing parameter in the traditional smoothing spline setting. By defining an additional hyperprior for the variance parameters, the amount of smoothness can be estimated simultaneously with the regression coefficients. A common choice is to assign the conjugate prior for σ_i^2, which is the inverse gamma prior $\sigma_i^2 \sim IG(a_j, b_j)$ with hyperparameters a_j and b_j. The posterior of the model's parameters $\theta = \{\beta_1, \beta_2, \ldots, \sigma_1^2, \sigma_2^2, \ldots, \gamma\}$ is given by

$$\begin{aligned} p(\theta | \Delta N_{1:K}, \mathcal{H}_K, x_{1:K}) \propto\ & \mathcal{L}(\Delta N_{1:K}; \mathcal{H}_K, x_{1:K}, \theta) \\ & \times \prod_i \frac{1}{(\sigma_i^2)^{R/2}} \exp\left(-\frac{1}{2\sigma_i^2} \beta_i' H_i^2 \beta_i \right) \\ & \times \prod_i \left(\sigma_i^2\right)^{-a_i - 1} \exp\left(-\frac{b_i}{\sigma_i^2} \right), \end{aligned} \tag{15.14}$$

where the last two terms on the right are the priors for the spline coefficients and for the variance of the random walk, respectively; the term H_i^2 corresponds to a second-order difference matrix with rank denoted by R. (Parameters related to tensor product splines were omitted here for simplicity.) Spline parameters and smoothing

terms are estimated via MCMC sampling. Good mixing properties of the Markov chain are provided by a sampling scheme that combines an approximation of the full conditionals of regression parameters via IRLS and uses them as proposals in a Metropolis–Hastings algorithm (Gamerman 1997). BayesX (Brezger et al. 2005) is a publicly available software for Bayesian P-spline modeling. Its algorithmic implementation has been shown (Brezger et al. 2005) to be exceptionally faster than WinBUGS, another commonly used software for Bayesian inference (Spiegelhalter et al. 2003). Initial exploration of penalized B-splines and Bayesian P-splines in the context of neural point processes was presented in Truccolo and Donoghue (2007). A free-knot Bayesian spline approach where the number of knots is inferred via reversible-jump MCMC sampling was proposed by DiMatteo et al. (2001). Related Bayesian inference approaches for GLMs have also appeared in Rigat et al. (2006) and in Stevenson et al. (2009). A general approach based on variational Bayesian inference for GLMs has been presented by Chen et al. (2010).

15.2.4 Nonparametric Function Approximation

The development of nonparametric methods for the approximation of conditional intensity functions of neural point processes is a recent topic of interest (Truccolo and Donoghue 2007; Coleman and Sarma 2007; Cunningham et al. 2008; Rahnama Rad and Paninski 2008). In this case, estimation of the conditional intensity function can be formulated as an optimization in function space. Truccolo and Donoghue (2007) extended a greedy function approximation approach based on stochastic gradient boosting (Friedman 2001) to neural point processes. Briefly, the function in (15.5) is represented as a target function

$$F^* = \arg\min_F E_{[\Delta N, z]}\left[L\left(\Delta N_{1:K}, F(z_{1:K})\right)\right] \tag{15.15}$$

to be approximated, where $L(\cdot)$ is a loss function which corresponds here to the negative of the discrete-time point process log-likelihood. Greedy function approximation approaches commonly involve an iterative procedure in which the approximation to $F^*(z)$ takes the form

$$F^*(z) \approx \hat{F}(z) = \sum_{m=0}^{M} f_m(z), \tag{15.16}$$

where $f_0(z)$ is an initial guess, and $\{f_m(z)\}_{m=1}^M$ are successive increments ("steps" or "boosts"), each dependent on the preceding step. In the particular case of function approximation via steepest-descent gradient, the mth step has a (steepest-descent) direction defined by the negative gradient vector of the loss function with respect to $F(z)$ and a magnitude obtained via line search. Generalization to new data is achieved by fitting a parametric model to the negative gradient at each of the successive steps

$$\hat{F}(z) = \hat{F}_0(z) + \sum_{m=1}^{M} \eta\beta_m h(z; a_m), \tag{15.17}$$

where $\hat{F}_0(z)$ is a constant, $h(z, a)$ is a regressor or base learner in the form of indicator functions (regression trees) with parameters β_m and $a_m = a_1, a_2, \ldots,$ and $0 < \eta \leq 1$ is a regularization parameter. The conditional intensity function is thus, in this case, approximated by piecewise constant functions. The update rule at each iteration is given by

$$\hat{F}_m(z) = \hat{F}_{m-1}(z) + \eta\beta_m h(z; a_m). \tag{15.18}$$

Intuitively, stochastic gradient boosting can be seen as the iterative and regularized fitting of the "residuals" in a gradual fashion. Note also that, in principle, other regressors or base learners, such as GLMs, could be chosen. However, among many advantages, the choice of indicator functions also provides an easy way to control for the interaction order of the approximation

$$F^*(z) = \sum_i F_i(z_i) + \sum_{ij} F_{ij}(z_i, z_j) + \sum_{ijk} F_{ijk}(z_i, z_j, z_k) + \cdots, \tag{15.19}$$

i.e., second- and higher-order interactions among the covariates (see Truccolo and Donoghue 2007 for details). Function approximation based on stochastic gradient boosting offers a robust nonparametric approach particularly when dealing with large datasets and a large number of covariates. Kernel-based methods (e.g., Gaussian processes) and tensor product splines are often not practical in these cases. Bayesian approaches still require a substantial amount of "fine tuning" of hyperparameters and considerable expertise in MCMC sampling.

Another interesting function approximation approach was proposed in Coleman and Sarma (2007). The optimization problem corresponds to minimization of the negative log-likelihood function under the constraint that the canonical parameter is Lipschitz continuous:

$$\left|\log[\hat{\lambda}_{k_1}\Delta] - \log[\hat{\lambda}_{k_2}\Delta]\right| \leq M\|z_{k_1} - z_{k_2}\|_\infty \tag{15.20}$$

for $k_1, k_2 = 1, \ldots, K$. M is the Lipschitz constant which works here as a regularization parameter or model selection term. For example, as $M \to \infty$, the estimated conditional intensity converges pointwise to the spike train data. This parameter can be determined according to some goodness-of-fit measure (e.g., via the time-rescaling theorem) or some other performance measure under a crossvalidation scheme. Coleman and Sarma (2007) implemented the minimization in terms of a computationally efficient dual problem.

Figure 15.2 illustrates some of the estimation approaches described in this section. The conditional intensity function of a primary motor cortex (M1) neuron, from a monkey performing a center-out point-to-point reaching task, was modeled as a function of the time elapsed since the last spike, the hand movement direction, and the two-dimensional hand kinematics (position, velocity, acceleration). This particular neuron showed little, if any, modulation for movements in the lower-left quadrant of the workspace. Four different models were used: log-linear both in the parameters and covariates, penalized B-splines, Bayesian P-splines, and gradient boosting regression using indicator functions. The gradient boosting regression model performed best (based on comparison of log-likelihood functions evaluated

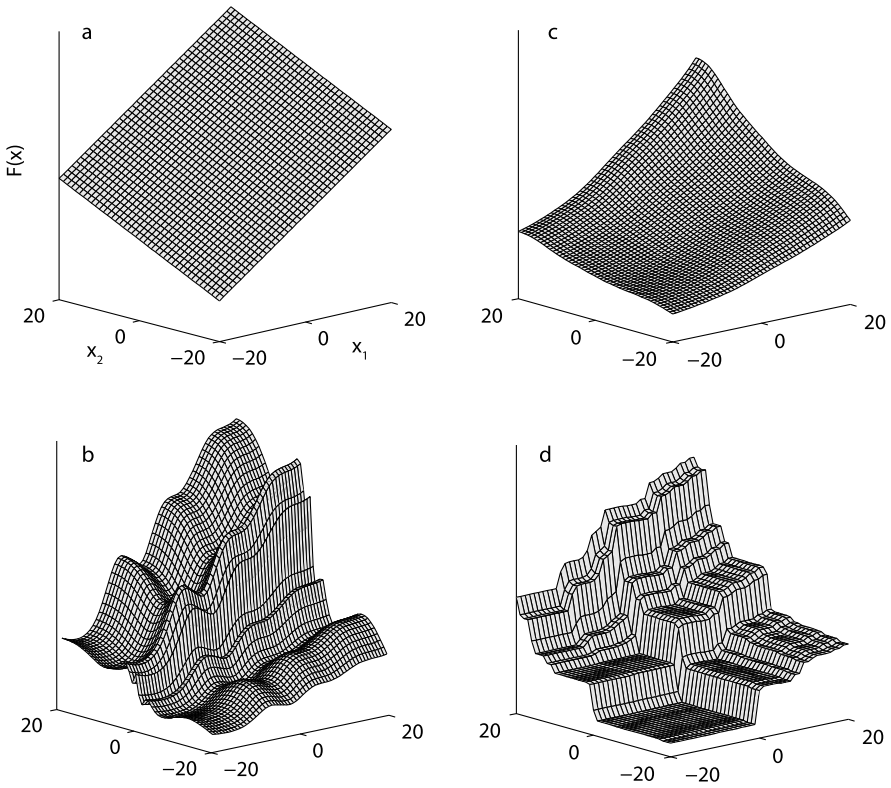

Fig. 15.2 Examples of conditional intensity functions (in log scale) estimated for an M1 neuron using: (**a**) simple log-linear model, (**b**) penalized B-splines, (**c**) Bayesian P-splines, and (**d**) gradient boosting regression. Only the partial dependencies on velocity (in cm/s) are shown. (Modified with permission from Truccolo and Donoghue 2007.)

on test data), while the penalized splines model incurred large oscillations in specific ranges of the velocity space. Note, however, that these differences in performance might also reflect the choices of regularization parameters and hyperparameters in these different approaches.

15.2.5 Statistical Inference

We conclude this section by briefly reviewing a few aspects of statistical inference for conditional intensity function models. The important issue of consistency and efficiency of estimators in the context of neural spike-train data has been discussed in Kass et al. (2005). Approximate confidence intervals for GLMs are typically derived from the observed Fisher information matrix. In this case, standard errors for estimated parameters are given by the square root of the inverse of the observed Fisher

information matrix evaluated at the maximum likelihood estimate. Confidence intervals, corresponding Wald-statistics and p-values based on this approximation are readily available in most statistical packages for GLMs. Estimation of confidence intervals where the Fisher-information-based approximation may fail (due to, for example, small sample size) or is not feasible, as in the nonparametric function approximation approaches described above, can be attempted via parametric bootstrap methods (Chap. 18). Efficient simulation of the neural point process, required in parametric bootstrap, can be implemented via the time-rescaling theorem (Brown et al. 2001).

Goodness-of-fit tests for continuous-time conditional intensity models can be derived from the time-rescaling theorem (Brown et al. 2001; Truccolo et al. 2004). Specifically, once a conditional intensity function model has been fit to a spike-train data series, rescaled times v_j can be computed as

$$v_j = \int_{u_j}^{u_{j+1}} \hat{\lambda}\big(t\big|z(t),\theta\big)\,dt \tag{15.21}$$

for $j = 1, \ldots, J-1$ spike times. Via this change of measure, the rescaled times should correspond to a homogeneous Poisson process with rate 1 if the estimated conditional intensity function is a good approximation of the true conditional intensity of the process. Thus the rescaled times should be independent. Further, the transformation

$$v_j^* = 1 - \exp(v_j) \tag{15.22}$$

results in v_j^* being uniformly distributed random variables in the interval $[0, 1)$. Therefore, to assess whether an estimated conditional intensity function fits the data, we need to check whether the v_j^* are independent and uniformly distributed. The Kolmogorov–Smirnov (K–S) test can be used to check for departure from the uniform distribution. Independence is a more difficult property to check. Czanner et al. (2008) suggested applying a further transformation

$$v_j^{**} = \Phi^{-1}\big(v_j^*\big), \tag{15.23}$$

where $\Phi(\cdot)$ is the cumulative distribution of a zero-mean and unit-variance Gaussian random variable. If the rescaled times v_j^* are independent and uniformly distributed in $[0, 1)$, then the v_j^{**} will be independent Gaussian random variables with mean zero and variance 1. Independence up to time lag τ of consecutive interspike intervals can then be checked by testing if the corresponding autocorrelation function departs significantly from zero.

The direct application of the above time-rescaling based goodness-of-fit assessment to discrete-time point process models can be problematic if Δ is not small enough. In this case, the rescaled ISIs are no longer exponentially distributed, and K–S tests might indicate poor fit even for exact conditional intensity function models. To address this issue, an extension of the time-rescaling theorem to discrete-time point processes has been proposed by Haslinger et al. (2009). A different approach for goodness-of-fit assessment, based on a generalization of the Rosenblatt's transformation, has been developed by Brockwell (2007). This generalized

transformation will map any arbitrary random vector (including binary and count variables) into a sequence of independent and uniformly distributed variables if the underlying probabilistic model is correct. K–S tests based on the time-rescaling theorem or on the generalized Rosenblatt's transformation provide an absolute measure of goodness-of-fit. When comparing different models, one will also be interested in relative measures such as the Akaike information criterion (Truccolo et al. 2004), predictive power based on receiver operating characteristic curves computed on test data (Truccolo et al. 2008a, 2009) and information rates (Harris et al. 2003).

15.3 Neural Ensemble Decoding

We can use the conditional intensity function models and estimation approaches described above to capture the encoding properties and dynamics of an ensemble of neurons. A reciprocal question is then: What is the probability of a covariate state, given observed ensemble spiking activities and estimated conditional intensity models? We follow the discrete-time stochastic state-space approach developed in Brown et al. (1998) and Eden et al. (2004). Among many applications, this approach has been used to decode from hippocampus (Brown et al. 1998) and sensorimotor cortex of humans and monkeys (Truccolo et al. 2004, 2008a). We will focus on the case where both the state and neural observations are jointly measured in a training dataset. The estimation of latent variable models (e.g., continuous-state HMMs) will not be treated here (see Smith and Brown 2003; Brockwell et al. 2007; Paninski et al. 2009a; Lawhern et al. 2010). We will consider state-space models that assume Markov state evolution. In addition, we will focus on the forward inference problem, rather than on smoothing, since the former is more relevant to real-time neural decoding applications, such as in neuroprostheses. More formally, the state and observation equations are given by

$$x_k \sim p(x_k|x_{k-1}), \tag{15.24}$$

$$\Delta N_k^c \sim \Pr\big(\Delta N \big| \lambda^c(t_k|\mathcal{H}_k, x_k)\big), \tag{15.25}$$

where $x_k \in \mathbb{R}^d$. The state transition probability $p(x_k|x_{k-1})$ is commonly defined by a linear Gaussian dynamical system

$$x_k = Ax_{k-1} + \varepsilon_k \tag{15.26}$$

with $\varepsilon_k \sim \mathcal{N}(\mathbf{0}, Q)$. The state and covariance matrices A and Q can be easily estimated from training data via maximum likelihood. Note that this first-order process can represent higher-order Markov processes by simple augmentation of the state space. The distribution for the observations in (15.25) is given by (15.6). Using Bayes' theorem and letting $\mathrm{N}_k = \{N_k^c\}_{c=1}^C$, we can write the posterior density as

$$p(x_k|\mathrm{N}_{1:k}) = \frac{p(\mathrm{N}_{1:k}, x_k)}{\Pr(\mathrm{N}_{1:k})}$$

$$= \frac{\Pr(\Delta N_k | N_{1:k-1}, x_k)\, p(N_{1:k-1}, x_k)}{\Pr(N_{1:k})}$$
$$= \frac{\Pr(\Delta N_k | \mathcal{H}_k, x_k)\, p(x_k | \mathcal{H}_k)}{\Pr(\Delta N_k | \mathcal{H}_k)}.$$

The one-step predictive density $p(x_k|\mathcal{H}_k)$ can be obtained recursively via the Chapman–Kolmogorov equation, which corresponds to marginalizing $p(x_k, x_{k-1}|\mathcal{H}_k)$ over x_{k-1}. The exact posterior density thus corresponds to

$$p(x_k|N_{1:k}) = \Pr(\Delta N_k|\mathcal{H}_k, x_k) \times \frac{\int p(x_k|x_{k-1})\, p(x_{k-1}|\Delta N_{k-1}, \mathcal{H}_{k-1})\, dx_{k-1}}{\Pr(\Delta N_k|\mathcal{H}_k)}. \quad (15.27)$$

Exact posterior computation can be implemented via numerical integration or via sequential Monte Carlo methods such as particle filters (Doucet et al. 2001; Brockwell et al. 2004; Ergun et al. 2007). However, these are currently not computationally efficient for real-time applications or for higher-dimensional state spaces (e.g., in particle filtering, the number of particles grows exponentially with the dimension of the space). Exact *maximum a posteriori* (MAP) solutions in the context of point process smoothing filters have been presented in Koyama and Paninski (2009) and Paninski et al. (2009a). In most applications, first- and second-order (fully exponential) Laplace approximations are the typical choice (Tierney et al. 1989; Koyama et al. 2010). Koyama et al. (2010) have shown that the approximation error in these sequential approximations is stable over time. Further, in typical applications to neural decoding, the inherent statistical error of the posterior tends to be much larger than the gain in accuracy obtained by using the second-order approximation. In this case, the first-order approximation will be as good (Koyama et al. 2010). The approach presented here is based on the first-order Laplace approximation (Eden et al. 2004; Truccolo et al. 2004).

The goal is to obtain a recursive expression for the mean and covariance of the posterior density of x_k in terms of observed spiking and estimated previous states. Let

$$x_{k|k-1} = E[x_k|\mathcal{H}_k], \quad (15.28)$$
$$W_{k|k-1} = \mathrm{var}[x_k|\mathcal{H}_k] \quad (15.29)$$

be the one-step prediction mean and covariance of the posterior density. The above definitions, along with the assumed evolution model, are sufficient to compute the one-step prediction to be defined below. Let

$$x_{k|k} = E[x_k|\Delta N_k, \mathcal{H}_k], \quad (15.30)$$
$$W_{k|k} = \mathrm{var}[x_k|\Delta N_k, \mathcal{H}_k] \quad (15.31)$$

be the mean and covariance of the posterior density, respectively. We seek the approximation

$$p(x_k|N_{1:k}) \approx \mathcal{N}(x_{k|k}, W_{k|k}) \propto \exp\left\{-\frac{1}{2}(x_k - x_{k|k})' W_{k|k}^{-1}(x_k - x_{k|k})\right\} \quad (15.32)$$

in terms of parameters $x_{k|k}$ and $W_{k|k}$ to be estimated. These two parameters are recursively estimated via the following stochastic state-space point process filter.

One-step prediction equations:

$$x_{k|k-1} = Ax_{k-1|k-1}, \tag{15.33}$$

$$W_{k|k-1} = AW_{k-1|k-1}A' + Q; \tag{15.34}$$

posterior variance and mean equations:

$$W_{k|k}^{-1} = W_{k|k-1}^{-1} + \sum_{c=1}^{C}\left[\left(\frac{\partial \log \lambda_k^c}{\partial x_k}\right)'\left[\lambda_k^c\Delta\right]\left(\frac{\partial \log \lambda_k^c}{\partial x_k}\right) - \left(\Delta N_k^c - \lambda_k^c\Delta\right)\frac{\partial^2 \log \lambda_k^c}{\partial x_k \partial x_k'}\right]_{x_{k|k-1}}, \tag{15.35}$$

$$x_{k|k} = x_{k|k-1} + W_{k|k}\sum_{c=1}^{C}\left[\left(\frac{\partial \log \lambda_k^c}{\partial x_k}\right)'\left(\Delta N_k^c - \lambda_k^c\Delta\right)\right]_{x_{k|k-1}}. \tag{15.36}$$

The derivation of the above stochastic point process filter involves two main steps (Eden et al. 2004). First, the one-step predictive density is given by $p(x_k|\mathcal{H}_k) \sim \mathcal{N}(x_{k|k-1}, W_{k|k-1})$. Under the true conditional intensity functions and according to the single neuron spiking probability in (15.6), the conditional spiking probability for the entire ensemble factorizes into

$$\Pr(\Delta \mathrm{N}_k|\mathrm{N}_{1:k-1}, x_k) = \prod_{c=1}^{C}\left[\lambda_k^c\Delta\right]^{\Delta N_k^c}\exp\left\{-\lambda_k^c\Delta\right\} + o(\Delta). \tag{15.37}$$

Therefore, the exact posterior, up to $o(\Delta)$, corresponds to

$$p(x_k|\mathrm{N}_{1:k}) \propto \prod_{c=1}^{C}\left[\lambda_k^c\Delta\right]^{\Delta N_k^c}\exp\left\{-\lambda_k^c\Delta\right\} \times \exp\left(-\frac{1}{2}(x_k - x_{k|k-1})'W_{k|k-1}^{-1}(x_k - x_{k|k-1})\right). \tag{15.38}$$

This expression represents the posterior as a function of the variable x_k. Second, after equating the target approximation (15.32) to the above and taking the log of both sides, we obtain

$$-\frac{1}{2}(x_k - x_{k|k})'W_{k|k}^{-1}(x_k - x_{k|k}) = \sum_{i=1}^{C}\left[\Delta N_k^i \log\left[\lambda_k^c\Delta\right] - \lambda_k^c\Delta\right] - \frac{1}{2}(x_k - x_{k|k-1})'W_{k|k-1}^{-1}(x_k - x_{k|k-1}) + \text{const.} \tag{15.39}$$

Taking partial derivatives with respect to the state variables results in an equation with only linear terms in the posterior mean and covariance. This equation can then be solved for these two parameters. Figure 15.3 illustrates the application of this

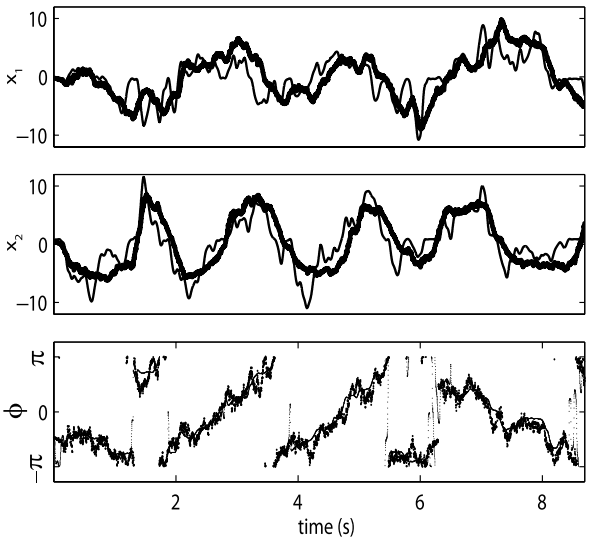

Fig. 15.3 Neural decoding of hand velocity. Decoded hand velocities (*horizontal* and *vertical coordinates*, in cm/s) are shown by the *thicker curves*. The *bottom plot* shows the decoded direction (in radians, *thicker dots*) of the velocity vector. (Modified with permission from Truccolo et al. 2004.)

stochastic state-space point process filter to the decoding of hand velocity from an ensemble of M1 neurons from a monkey performing a pursuit tracking task (Truccolo et al. 2004). The conditional intensity functions were given by velocity tuning models. A similar algorithm can also be implemented for neural decoding based on state-space models with count process observations, i.e., spike counts in specified time bins.

Note that the above algorithm requires the computation of partial derivatives of the conditional intensity function with respect to the decoded covariates. When using conditional intensity functions estimated via the nonparametric function approximation approaches described in Sect. 15.2.4, this might not be feasible. In those cases, neural decoding via particle filters provides an alternative. Neural decoding involving hybrid states (e.g., hand kinematics trajectories and discrete endpoint targets) have been explored in Srinivasan et al. (2007), Kulkarni and Paninski (2008). The stochastic state-space point process filter can also be used to track changes in the parameters of estimated conditional intensity function models. In this case, model parameters are represented as states in the state-space model (Eden et al. 2004; Czanner et al. 2008). Similar versions of this state-space approach can be extended to the estimation of hidden covariates and hidden common inputs, that is, continuous-state HMMs with point process observations (Smith and Brown 2003; Yu et al. 2006; Kulkarni and Paninski 2007; Brockwell et al. 2007; Wu et al. 2009; Lawhern et al. 2010). Further, instead of applying the commonly used expectation-maximization (EM) algorithm when estimating the parameters in these models, one can directly maximize an approximation to the marginal log-likelihood (Koyama and Paninski 2009). In this case, a Laplace approximation, centered at the exact MAP path for the hidden states, is used to approximate and efficiently compute the marginal log-likelihood and its gradients.

15.4 Neuronal Ensemble Collective Dynamics

It is widely accepted that the coordinated activity of neuronal ensembles, both locally and across different cortical areas, is the basis for cognition and adaptive behavior. Nonetheless, the study of this coordination at the level of single neurons remains a major challenge at both technological and statistical levels. Point process models to address this issue have been introduced in Brillinger (1988), Chornoboy et al. (1988), Martignon et al. (2000), Truccolo et al. (2004), Okatan et al. (2005), Nykamp (2007), Pillow et al. (2008), Truccolo et al. (2009). See also Chaps. 8, 9, 10, 11, and 12 for other approaches addressing the analysis of multineuron spatiotemporal patterns based on precise spike times. Here we briefly describe how the stochastic neural point process framework presented in this chapter can be used to investigate the collective dynamics of neuronal ensembles. Ultimately, we would like to estimate the joint distribution of neuronal states at any time instant conditioned on their previous history. We assume that this distribution, conditioned on the modeled history effects, factorizes into the product of the individual recursive distributions given in (15.6). As before, the main issue is then to approximate the conditional intensity of single neurons as a function of previous spiking history. We consider the following specific model:

$$\log\left[\hat{\lambda}^c(t_k|\mathcal{H}_t,\theta)\Delta\right] = \mu_c + \sum_{j=1}^{10} G_{1,c,j}\cdot\Delta N^c + \sum_{i\neq c}^{C}\sum_{j=1}^{4} G_{2,c,i,j}\cdot\Delta N^i, \quad (15.40)$$

where ΔN^c is the spiking train of the cth neuron in the past 100 ms, and $G_{1,c}$ and $G_{2,c,i}$ are temporal filters for the intrinsic and ensemble spiking histories, respectively, consisting of raised cosine basis functions with coefficients to be estimated (Pillow et al. 2008; Truccolo et al. 2009). Ten and four basis functions are used for the intrinsic and ensemble history filters, respectively. Model parameters are estimated via direct maximization of the penalized likelihood functions, with a ridge regression penalty term for the parameters related to the ensemble history. This and similar types of models can be used to study how functional connectivity is modulated in different task stages and experimental conditions (Harris et al. 2003; Okatan et al. 2005), the effect of correlations on neuronal responses to stimuli (Pillow et al. 2008), and collective dynamics in human and monkey sensorimotor cortex (Truccolo et al. 2009). As an illustration, we compare this model to a simpler model where the joint distribution of instantaneous collective states is approximated by a maximum entropy model (Jaynes 1982) constrained on mean spiking rates and (zero-lag) pairwise correlations

$$\Pr\left(\left\{\Delta N_k^c\right\}_{c=1}^{C}\right) = \frac{1}{Z_{(\alpha,\beta)}}\exp\left\{\sum_i \alpha_i z_{i,k} + \frac{1}{2}\sum_{i\neq j}\beta_{i,j} z_{i,k} z_{j,k}\right\}, \quad (15.41)$$

where $z_{i,k} = 1$ if $\Delta N_k^i = 1$ and $z_{i,k} = -1$ if $\Delta N_k^i = 0$, $Z_{(\alpha,\beta)}$ is a normalization term (Partition function), the $\{\alpha_i\}$ reflect constraints imposed by the empirical mean spiking rates, and the $\{\beta_{ij}\}$ reflect constraints imposed by the pairwise correlations. Equation (15.41) corresponds to the Ising model in statistical physics (Landau and Lifshitz 1958) and has been successfully used to study

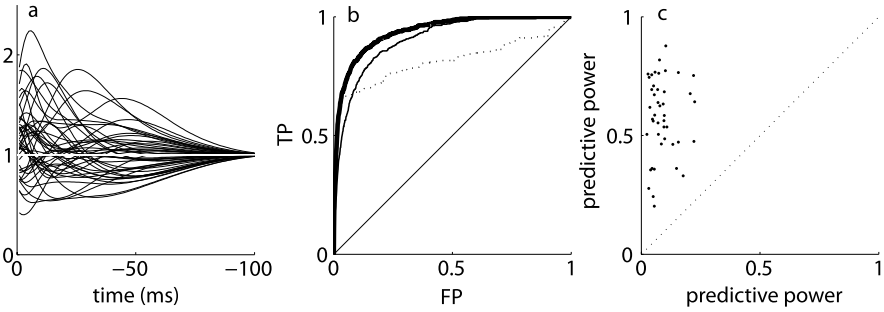

Fig. 15.4 Collective dynamics: predicting single-neuron spiking from ensemble spiking history. Intensity function models conditioned on intrinsic and ensemble histories were fitted to 45 M1 neurons of a monkey performing a reach and grasp task. (**a**) Each *curve* represents the contribution (exponentiated temporal filters) of the spiking history of each input neuron in the ensemble to an example target neuron. (**b**) ROC curves for spike prediction of the same example neuron based on the history model. TP and FP denote the true and false positive prediction rates. The *thicker curve* relates to the prediction based on both intrinsic (*dashed curve*) and ensemble histories (*thin curve*). The *diagonal line* represents the expected chance level prediction. (**c**) Comparison of the predictive power of ensemble history models (*vertical axis*) and of the pairwise maximum entropy joint distribution (Ising model, *horizontal axis*). Each *dot* corresponds to a target neuron

collective dynamics of ganglion cells in the retina (Schneidman et al. 2006; Shlens et al. 2006). This pairwise maximum entropy model does not explicitly incorporate history effects, but see Marre et al. (2009) for maximum entropy models constrained on multiple-lag auto- and pairwise cross-correlation functions. Both models were fitted to an ensemble of neurons recorded from M1 in a monkey performing point-to-point hand reaching movements. We assessed the strength of the collective dynamics by measuring how well the recent spiking history of the ensemble predicted single-neuron spiking. A measure based on Receiver Operating Characteristics (ROC) curves was used. Predictive power was computed as $2 \times (AUC - AUC^*)$, where AUC is the area under the ROC curve, and AUC^* is the estimated area for the chance-level prediction. The predictive power of the ensemble history can be substantial (Fig. 15.4). In addition, these history models showed higher predictive power than the instantaneous collective state captured by the zero-lag pairwise maximum entropy model. Truccolo et al. (2009) have shown that the spiking history of small, randomly sampled ensembles of cortical neurons in both human and nonhuman primates can predict subsequent single-neuron spiking. Spiking could be predicted by both local ensemble spiking histories and those in other cortical areas. These results provide evidence for strong collective cortical dynamics at the level of neuronal spikes in humans and monkeys performing sensorimotor tasks.

As mentioned before, the conditional spiking probabilities of all of the neurons in the ensemble (15.40) are recursively computed via a system of stochastic nonlinear equations with delays for the conditional intensity functions. Mean-field approximations to this system can be used to facilitate the analysis of the statistical and qualitative properties of the modeled collective dynamics and the relationship of these dynamics to other covariates such as sensory stimuli and behavior. Toy-

oizumi et al. (2009) derived corresponding ordinary differential equations for the mean spiking rates, auto- and cross-correlation functions, under the assumption of weak network coupling and weak history effects. To deal with the case of strong history effects (absolute refractory period), Toyoizumi et al. (2009) introduced a generalized linear model with Markov refractoriness. Cardanobile and Rotter (2010) applied mean-field approximations to study the dynamics of recurrent neuronal networks consisting of multiplicatively interacting point processes. Related theoretical work on mean-field approximations based on different neuronal and network models has been presented in Meyer and van Vreeswijk (2002) and in Ginzburg and Sompolinsky (1994). Commonly, the analysis of statistical models of stochastic neuronal networks has to rely on Monte Carlo simulations, which besides being computationally intensive, do not always provide direct insight into the network's properties. If assumptions and required conditions are satisfied, mean-field approximations might provide a complementary and efficient approach for the analysis of the network's statistical and dynamical system properties.

15.5 Summary and Future Directions

Spiking probability models for neuronal ensembles play a fundamental role in state-space neural decoding algorithms and in the analysis of neuronal collective dynamics. The framework described in this chapter offers computationally efficient ways to estimate this probability via the statistical estimation of conditional intensity functions. One important application of this framework will be the study of how neuronal collective dynamics affect the encoding properties of single neurons and the performance of decoding algorithms that incorporate information about neuronal interdependencies.

References

Barbieri R, Quirk MC, Frank LM, Wilson MA, Brown EN (2001) Construction and analysis on non-Poisson stimulus-response models of neural spiking activity. J Neurosci Methods 105:25–37

Brezger A, Kneib T, Lang S (2005) BayesX: analyzing Bayesian structured additive regression models. J Statist Software 14(11):1–22

Brezger A, Lang S (2006) Generalized structured additive regression based on Bayesian P-splines. Comput Stat Data Anal 50:967–991

Brillinger DR (1988) Maximum likelihood analysis of spike trains of interacting nerve cells. Biol Cybern 59:189–200

Brockwell AE, Rojas AL, Kass RE (2004) Recursive Bayesian decoding of motor cortical signals by particle filtering. J Neurophys 91(2):1899–1907

Brockwell AE, Kass RE, Schwartz AB (2007) Statistical signal processing and the motor cortex. Proc IEEE 95(5):881–897

Brockwell AE (2007) Universal residuals: a multivariate transformation. Stat Probab Lett 77:1473–1478

Brown EN, Frank LM, Tang D, Quirk MC, Wilson MA (1998) A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. J Neurosci 18:7411–7425

Brown EN, Barbieri R, Ventura V, Kass RE, Frank LM (2001) The time-rescaling theorem and its application to neural spike train data analysis. Neural Comput 14:325–346

Cardanobile S, Rotter S (2010) Multiplicatively interacting point processes and applications to neural modeling. J Comput Neurosci 28(2):267–284

Chen Z, Kloosterman F, Wilson M, Brown EN (2010) Variational Bayesian inference for point process generalized linear models in neural spike train analysis. In: Proc. IEEE ICASSP'10, Dallas, TX, pp 2086–2089

Chornoboy ES, Schramm PL, Karr AF (1988) Maximum likelihood identification of neuronal point process systems. Biol Cybern 59:265–275

Coleman T, Sarma S (2007) A computationally efficient method for modeling neural spiking activity with point processes nonparametrically. In: Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans

Cunningham JP, Yu BM, Shenoy KV (2008) Inferring neural firing rates from spike trains using Gaussian processes. In: Platt JC, Koller D, Singer Y, Roweis S (eds) Advances in neural information processing systems, vol 20. MIT Press, Cambridge

Czanner G, Eden UT, Wirth S, Yanike M, Suzuki W, Brown E (2008) Analysis of between-trial and within-trial neural spiking dynamics. J Neurophys 99:2672–2693

Daley D, Vere-Jones D (2003) An introduction to the theory of point processes. Springer-Verlag, New York

DiMatteo I, Genovese C, Kass R (2001) Bayesian curve fitting with free-knot splines. Biometrika 88:1055–1073

Donoghue JP (2008) Bridging the brain to the world: a perspective on neural interface systems. Neuron 60:511–521

Doucet A, de Freitas N, Gordon N (2001) Sequential Monte Carlo methods in practice. Springer-Verlag, New York

Eden UT, Frank LM, Barbieri T, Solo V, Brown EN (2004) Dynamic analyses of neural encoding by point process adaptive filtering. Neural Comput 16(5):971–998

Eilers PHC, Marx BD (1996) Flexible smoothing with B-splines and penalties. Statist Sci 11:89–102

Ergun A, Barbieri R, Eden UT, MA Wilson, Brown EN (2007) Construction of point process adaptive filter algorithms for neural systems using sequential Monte Carlo methods. IEEE Trans Biomed Eng 54(3):416–428

Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Statist 29(5):1189–1232

Friedman J, Hastie T, Tibshirani R (2008) Regularization paths for generalized linear models via coordinate descent. Available online at http://www-stat.stanford.edu/~hastie/Papers/glmnet.pdf

Gamerman D (1997) Sampling from the posterior distribution in generalized linear mixed models. Stat Comput 7:57–68

Ginzburg I, Sompolinsky H (1994) Theory of correlations in stochastic neural networks. Phys Rev E 50(4):3171–3191

Harris K, Csicsvari J, Hirase H, Dragoi G, Buzsaki G (2003) Organization of cell assemblies in the hippocampus. Nature 424:552–556

Haslinger R, Brown EN, Pipa G (2009) Discrete time rescaling theorem: determining goodness of fit for discrete time models of neural spiking. Abstract No 789.9. Society for Neuroscience, Washington

Hastie T, Tibshirani R, Friedman J (2001) The elements of statistical learning. Springer-Verlag, New York

Hatsopoulos NG, Ojakangas CL, Paninski L, Donoghue JP (1998) Information about movement direction obtained from synchronous activity of motor cortical neurons. Proc Natl Acad Sci USA 95:15706–15711

Hatsopoulos NG, Xu Q, Amit Y (2007) Encoding of movement fragments in the motor cortex. J Neurosci 27(19):5105–5114

Hochberg LR, Serruya MD, Friehs GM, Mukand JA, Saleh M, Caplan AH, Branner A, Chen D, Penn RD, Donoghue JP (2006) Neuronal ensemble control of prosthetic devices by a human with tetraplegia. Nature 442:164–171

Jaynes ET (1982) On the rationale of maximum entropy methods. Proc IEEE 70:939–952

Jermakowicz WJ, Chen X, Khaytin I, Bonds AB, Casagrande VA (2009) Relationship between spontaneous and evoked spike-time correlations in primate visual cortex. J Neurophysiol 101:2279–2289

Johnson DH (1996) Point process models of single-neuron discharges. J Comput Neurosci 3(4):275–299

Kass RE, Ventura V (2001) A spike-train probability model. Neural Comput 13:1713–1720

Kass R, Ventura V, Brown EN (2005) Statistical issues in the analysis of neuronal data. J Neurophys 94:8–25

Koyama S, Paninski L (2009) Efficient computation of the maximum a posteriori path and parameter estimation in integrate-and-fire and more general state-space models. J Comput Neurosci. doi:10.1007/s10827-009-0150. Advance online publication

Koyama S, Perez-Bolde LC, Shalizi CR, Kass RE (2010) Approximate methods for state-space models. J Amer Statist Assoc 105(489):170–180

Kulkarni J, Paninski L (2007) Common-input models for multiple neural spike-train data. Network Comput Neural Syst 18:375–407

Kulkarni J, Paninski L (2008) State-space decoding of goal-directed movements. IEEE Signal Process Mag Jan:78–86

Landau LD, Lifshitz EM (1958) Statistical physics. Pergamon, London

Lawhern V, Wu W, Hatsopoulos NG, Paninski L (2010) Population decoding of motor cortical activity using a generalized linear model with hidden states. J Neurosci Methods 189(2):267–280

Lesaffre E, Marx BD (1993) Collinearity in generalized linear regression. Commun Stat Theory Methods 22(7):1933–1952

Marre O, El Boustani S, Fregnac Y, Dextexhe A (2009) Prediction of spatiotemporal patterns of neural activity from pairwise correlation. Phys Rev Lett 102(138101):1–4

Marmarelis VZ (2004) Nonlinear dynamic modeling of physiological systems. Wiley, Hoboken,

McCullagh P, Nelder JA (1989) Generalized linear models, 2nd edn. Chapman & Hall/CRC, Boca Raton

Martignon L, Deco G, Laskey K, Diamond M, Freiwald W, Vaadia E (2000) Neural coding: higher-order temporal patterns in the neuro-statistics of cell assemblies. Neural Comput 12:2621–2653

Meyer C, van Vreeswijk C (2002) Temporal correlations in stochastic networks of spiking neurons. Neural Comput 14:369–404

Nicolelis MAL, Dimitrov D, Carmena JM, Crist R, Lehew G, Kralik JD, Wise SP (2003) Chronic, multisite, multielectrode recordings in macaque monkeys. Proc Natl Acad Sci USA 100(19):11041–11046

Nicolelis MAL, Lebedev MA (2009) Principles of neural ensemble physiology underlying the operation of brain-machine interfaces. Nature Rev Neurosci 10:530–540

Nykamp D (2007) A mathematical framework for inferring connectivity in probabilistic neuronal networks. Math Biosci 205:204–251

Okatan M, Wilson MA, Brown EN (2005) Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. Neural Comput 9:1927–1961

Paninski L, Ahmadian Y, Ferreira DG, Koyama S, Rahnama Rad K, Vidne M, Vogelstein J, Wu W (2009a) A new look at state-space models for neural data. J Comput Neurosci. doi:10.1007/s10827-009-0179. Advance online publication

Paninski L, Kass R, Brown E, Iyengar I (2009b) Statistical analysis of neuronal data via integrate-and-fire models. In: Laing, C, Lord, G (eds) Stochastic methods in neuroscience. Oxford University Press, London

Paninski L (2004) Maximum likelihood estimation of cascade point-process neural encoding models. Network Comput Neural Syst 15:243–262

Pillow JW, Shlens J, Paninski L, Sher A, Litke AM, Chichilnisky EJ, Simoncelli EP (2008) Spatio-temporal correlations and visual signaling in a complete neuronal population. Nature 454:995–999

Rahnama Rad K, Paninski L (2008) Efficient estimation of two dimensional firing rate surfaces via Gaussian process methods. In: Computational systems neuroscience (COSYNE) conference

Riehle A, Grun S, Diesmann M, Aertsen A (1997) Spike synchronization and rate modulation differentially involved in motor cortical function. Science 278:1950–1953

Rigat F, de Gunst M, van Pelt J (2006) Bayesian modelling and analysis of spatio-temporal neuronal networks. Bayesian Anal 1(1):733–764

Schneidman E, Berry M, Segev R, Bialek W (2006) Weak pairwise correlations imply strongly correlated network states in a neural population. Nature 440:1007–1012

Shlens J, Field GD, Gauthier JL, Grivich MI, Petrusca D, Sher A, Litke AM, Chichilnisky EJ (2006) The structure of multi-neuron firing patterns in primate retina. J Neurosci 26:8254–8266

Smith AC, Brown EN (2003) Estimating a state-space model from point process observations. Neural Comput 15:965–991

Spiegelhalter D, Thomas A, Best N, Lunn D (2003) WinBUGS user manual (Version 1.4). Medical Research Council Biostatistics Unit, Cambridge

Srinivasan L, Eden UT, Mitter SK, Brown EN (2007) General-purpose filter design for neural prosthetic devices. J Neurophysiol 98:2456–2475

Stevenson IH, Rebesco JM, Hatsopoulos NG, Haga Z, Miller LE, Körding KP (2009) Bayesian inference of functional connectivity and network structure from spikes. IEEE Trans Neural Syst Rehabil Eng 17(3):203–213

Tierney L, Kass RE, Kadane JB (1989) Fully exponential Laplace approximations to expectations and variances of nonpositive functions. J Amer Statist Assoc 84:710–716

Toyoizumi T, Rahnama Rad K, Paninski L (2009) Mean-field approximations for coupled populations of generalized linear model spiking neurons with Markov refractoriness. Neural Comput 21:1203–1243

Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN (2004) A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. J Neurophysiol 93:1074–1089. doi:10.1152/jn.00697.2004

Truccolo W, Donoghue JP (2007) Non-parametric modeling of neural point processes via stochastic gradient boosting regression. Neural Comput 19(3):672–705

Truccolo W, Friehs GM, Donoghue JP, Hochberg LR (2008a) Primary motor cortex tuning to intended movement kinematics in humans with tetraplegia. J Neurosci 28(5):1163–1178

Truccolo W, Hochberg LR, Eskandar E, Cole A, Cash SS (2008b) Multielectrode array recordings of single unit activity in humans with epilepsy. In: Neural interfaces conference. Cleveland

Truccolo W, Hochberg LR, Donoghue JP (2009) Collective dynamics in human and monkey sensorimotor cortex: predicting single neuron spikes. Nature Neurosci. doi:10.1038/nn.2455. Advance online publication, 6 Dec 2009

Wedderburn RWM (1976) On the existence and uniqueness of the maximum likelihood estimates for certain generalized linear models. Biometrika 63:27–32

Wilson MA, McNaughton BL (1993) Dynamics of the hippocampal ensemble code for space. Science 261:1055–1058

Wood SN (2006) Generalized additive models. Chapman & Hall/CRC, Boca Raton

Wu W, Kulkarni JE, Hatsopoulos NG, Paninski L (2009) Neural decoding of hand motion using a linear state-space model with hidden states. IEEE Trans Neural Syst Rehabil Eng 17(4):370–378

Yu BM, Afshar A, Santhanam G, Ryu S, Shenoy K, Sahani M (2006) Extracting dynamical structure embedded in neural activity. In: Advances in neural information processing systems, vol 18. MIT Press, Cambridge, pp 1545–1552

Zhao M, Iyengar S (2010) Nonconvergence in logistic and Poisson models for neural spiking. Neural Comput 22:1231–1244

Part V
Practical Issues

Chapter 16
Simulation of Stochastic Point Processes with Defined Properties

Stefano Cardanobile and Stefan Rotter

Abstract We describe procedures that allow one to numerically simulate artificial spike trains matching real spike trains with respect to interspike interval distributions, in particular firing rates, interspike interval irregularity, and spike-count variability, and also time-varying firing rates and the corresponding properties in the nonstationary case.

Spike trains recorded from neocortical neurons result from complicated interactions among very many cells. Due to a notorious lack of knowledge about the structure of the underlying network, it is currently impossible to capture the dynamical processes directly in terms of explicit biophysical models. It is useful, though, to consider abstract stochastic models that summarize all unknown details in terms of appropriate statistical ensembles.

Stochastic point processes represent a useful mathematical abstraction of neuronal spike trains (see Chap. 1, this volume). Therefore, numerical simulations of point processes with defined properties are an important tool for exploration and for the reliable interpretation of measured data. Certain statistical procedures critically depend on the availability of data in a format analogous to measured data, but with well-defined probabilistic properties.

16.1 Point Processes and Thinning

This section gives a general definition of stochastic point processes along with a general framework to simulate them in a continuous-time environment. A stochas-

S. Cardanobile (✉)
Bernstein Center Freiburg, Albert-Ludwig University, Hansastrasse 9a, 79104 Freiburg, Germany
e-mail: cardanobile@bccn.uni-freiburg.de
url: http://www.bcf.uni-freiburg.de/people/details/cardanobile

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_16, © Springer Science+Business Media, LLC 2010

tic point process in time is, loosely speaking, a sequence of spikes occurring at random times. An additional formal requirement is that only finitely many spikes are generated during an observation of finite duration. Here, we are interested in causal point processes, i.e., processes for which the instantaneous firing rate at a given time only depends on the previous history of the process and on some external covariates (Cox and Isham 1980). Let $\mathcal{H}(t)$ represent the history of the system until time t. Assuming a sufficiently regular process, the requirement of causality reads

$$\mathbb{P}\big[\text{spike in } [t, t+dt)\big] = h\big(x(t), \mathcal{H}(t)\big)\, dt$$

for small dt. In a neuroscientific setting, the covariates $x(t)$ could be some sensory input, the activity of neighboring neurons, or any other relevant external information. The function h has many names: hazard rate, conditional intensity, age-specific failure rate. We will refer to it as the hazard rate of the process.

The simplest example is where the hazard rate is constant,

$$h\big(x(t), \mathcal{H}(t)\big) = \lambda.$$

In this case, the actual firing rate is a constant and depends neither on any external variable nor on the previous history of the process itself. This process is known as the *Poisson process*. We will return to it in Sect. 16.2.

In this paper we are concerned with the generation of point process with bounded hazard rate, where a positive number $\lambda_{\max}$ exists such that $h(x(t), \mathcal{H}(t)) \leq \lambda_{\max}$ for all times t. Many useful processes, including the Poisson process with dead time, certain gamma-processes, and log-normal processes, are of this type. The bounded hazard is, from a practical point of view, a very weak assumption.

We will follow a general strategy to simulate point processes. We will start by realizing a Poisson process with rate $\lambda_{\max}$ (see Sect. 16.2) and then randomly reject spikes according to the hazard rate of the process. We discuss a simple example first. Assume that a neuron is receiving sinusoidal input $x(t) = 1 + \sin(t)$ and we want to simulate it by using

$$h\big(x(t), \mathcal{H}(t)\big) = x(t). \tag{16.1}$$

This case is known as the inhomogeneous Poisson process. The hazard h satisfies our assumption, since $h(t) \leq 2$ for all t. We start the simulation by generating random spikes at rate 2, following the algorithm described in Sect. 16.2. Using this method, we obtain a spike train, and we denote from now on the sequence of spikes by (t_i). If we now retain each spike in the original Poisson train independently with probability $h(t)/2$, we obtain a spike train which corresponds to the point process defined by the hazard (16.1): Since we are keeping spikes randomly, the probability of having a spike at time t is given by

$$\mathbb{P}\big[\text{“raw” spike in } [t, t+dt)\big] \times \mathbb{P}[\text{keep spike around } t] = \lambda_{\max}\, dt\, \frac{x(t)}{\lambda_{\max}} = x(t)\, dt,$$

which is exactly what we wanted.

There are two main alternatives to the thinning approach. The first uses discretized time and approximates the continuous-time point process by a discrete sequence of 0s and 1s. This method is usually fast, but it has the distinct disadvantage

of a finite time resolution determined by the bin width of the discretization. The second simulates renewal processes by directly sampling i.i.d. interspike intervals (ISIs) τ_i from the corresponding interspike interval distribution and stacking them on top of each other to obtain the actual spike times. This method can also be used in cases where the ISI distribution is known, but not the conditional rate, or if the conditional rate diverges. If the ISIs are sampled by a rejection method, this leads to algorithms very similar to the ones described in this manuscript. One could also draw independent random numbers u_i that are uniformly distributed on $[0, 1]$ and then solve

$$u_i = F(\tau_i),$$

where F is the cumulative distribution function (c.d.f.) of the ISI distribution. This method is computationally expensive if no closed expression is known for the inverse of the cumulative distribution function.

We prefer the thinning method mainly because of its flexibility. It can be applied whenever the conditional intensity is bounded. But, in contrast to discrete-time methods, it yields spike times with machine precision. It is computationally disadvantageous only if the hazard rate has large excursions such that too many events are rejected. Let us finally mention that thinning is nothing but rejection sampling in the context of point processes, see, e.g., Ripley (1987).

16.2 Poisson Processes

16.2.1 Homogeneous Poisson Process

The Poisson process is the simplest conceivable point process, and it is the basis to generate other point processes via thinning. We now explain in detail how to simulate a Poisson process with constant rate λ on an observation interval $[0, T]$. A Poisson neuron fires spikes at any point in time with the same probability, independently of the own history. From this assumption it is possible to derive the distribution of the spike times, *conditional on the number of spikes*. The number of spikes in an observation is random, but let us for a moment assume that it is given. Then, by the definition of the process, spikes will occur with the same probability at any point in time. This means that each spike is *uniformly distributed* on the interval $[0, T]$ and independent on the position of all the others.

We can thus generate the Poisson process if we know the distribution of the number of spikes in an observation. As a matter of fact, the number of spikes is a Poisson-distributed random variable:

$$\mathbb{P}\big[N \text{ spikes in } [0, T]\big] = \frac{(\lambda T)^N}{N!} e^{-\lambda T}.$$

This can be proven by representing the Poisson process as a sum of Bernoulli variables for small time bins and resorting to the Euler definition of the exponential function, see, e.g., Wikipedia, the free encyclopedia (2009) for details.

To summarize, the Poisson process can be simulated by the following algorithm:

Poisson process:

(1) Draw a random number N_T distributed according to the Poisson distribution with mean λT.
(2) For $k = 1, \ldots, N_T$, draw an independent spike time uniformly distributed on $[0, T]$.

16.2.2 Inhomogeneous Poisson Process

We now move to the generation of nonstationary Poisson processes. This will be the first application of the thinning approach that we have described in Sect. 16.1. Before describing the algorithm, however, we have to formally define the inhomogeneous Poisson process. We fix a positive function $\lambda(t)$, where t is a real variable ranging in the observation interval $[0, T]$. We assume that the function $\lambda(t)$ has an upper bound $\lambda_{\max}$ for all $t \in [0, T]$. For the inhomogeneous Poisson process, the hazard rate may depend on time, but not on the history of the process,

$$\mathbb{P}\big[\text{spike in } [t, t + dt) \,\big|\, \mathcal{H}(t)\big] = \lambda(t)\, dt \tag{16.2}$$

for small dt, independently on the history $\mathcal{H}(t)$. Since the hazard is independent of the history, it is possible to generate an inhomogeneous Poisson process by copying spikes from a homogeneous Poisson process with a certain probability. We call the original homogeneous process the reference Poisson process, and the spike train we want to generate the target inhomogeneous Poisson process. Our strategy is to randomly copy spikes from the reference to the target, since spikes from the target process occur at a rate that is $\frac{\lambda(t)}{\lambda_{\max}}$ times the rate of the reference process.

These considerations lead us to the following algorithm to generate an inhomogeneous Poisson process with rate profile $\lambda(t)$:

Inhomogeneous Poisson process:

(1) The set of the target spikes s_i is initially empty.
(2) Generate a Poisson random number P with mean $\lambda_{\max} T$.
(3) Place P random spikes t_i uniformly and independently on the interval $[0, T]$.
(4) For every spike t_i, draw an independent random number r_i uniformly distributed in $[0, 1]$.

For every reference spike t_i:

(5) Check whether

$$r_i < \frac{\lambda(t_i)}{\lambda_{\max}}.$$

(6) If yes, add t_i to the list of target spikes.

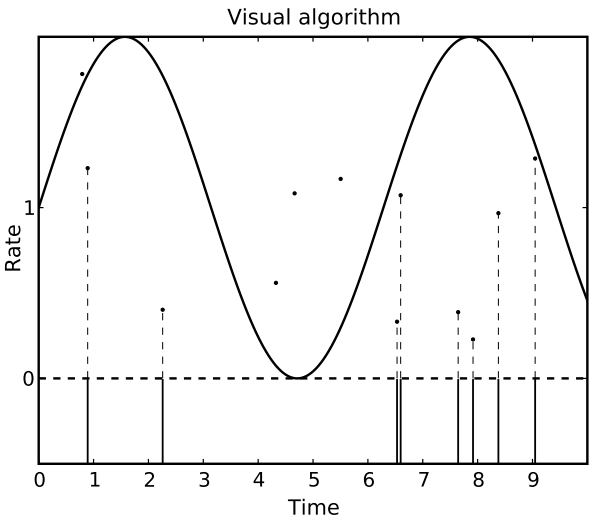

Fig. 16.1 Visualization of the algorithm for the simulation of an inhomogeneous Poisson process. *Continuous line*: rate profile. *Points*: reference random numbers. *Bars*: surrogate spikes, corresponding to the x-coordinates of the reference random numbers that are below the rate profile

Observe that the expected number of random numbers needed for the simulation is $1 + 2\lambda_{\max}T$. This is of advantage in the case where the rate profile is highly dynamic, but the maximal rate is of moderate size. In this case, low-resolution time discretization is a source of errors, but for fine discretizations, a very large number of random numbers is needed.

This algorithm can be rephrased as follows (see Fig. 16.1): Consider a rectangle with base T and height $\lambda_{\max}$ that contains the rate profile. Draw a random number N from a Poisson distribution with mean $\lambda_{\max}T$. Place N points randomly (uniform density) and independently within the rectangle. The target spikes are the t-coordinates of the points with a λ-coordinate below the rate profile.

16.2.3 Count Distribution and Operational Time

The count distribution of the number of spikes observed in an inhomogeneous Poisson process can be understood from the simulation algorithm. The inhomogeneous Poisson process is obtained by independently rejecting spikes of a homogeneous Poisson process, where the rejection probability is a function of time. Therefore, their total number is also Poisson distributed, and we only have to know its mean to determine the distribution completely. In view of the simulation algorithm (see Fig. 16.1), the expected number of spikes in the observation interval $[0, t]$ is proportional to the corresponding area under the rate profile $\lambda(t)$:

$$\Lambda(t) := \int_0^t \lambda(s)\, ds. \tag{16.3}$$

The function $\Lambda(t)$ plays an important role in the theory of point processes. It represents the transformation from *real time* to *operational time*: If t_i are the points of

an inhomogeneous Poisson process with rate profile $\lambda(t)$, then $s_i = \Lambda(t_i)$ are the points of a homogeneous Poisson process with unit rate. They all lie in the interval $[0, \Lambda(T)]$ in (unit-less) operational time. The number of spikes in an inhomogeneous Poisson processes is distributed as

$$\mathbb{P}[N \text{ spikes until } T] = \frac{\Lambda(T)^N}{N!} e^{-\Lambda(T)}.$$

We will come back to this concept in the next section, when introducing nonstationary renewal processes.

16.2.4 Correlated Poisson Processes

In real data correlations between different neurons are often observed, see Chap. 12, this volume. Here, we present a simple procedure to generate correlated Poisson spike trains. The algorithm generates surrogate data with the following properties: each individual spike train should be a Poisson process, and there is a nonvanishing probability for spikes to occur simultaneously in pairs or larger groups of spike trains. Independent Poisson processes cannot have the latter property, since the probability of having two spikes in the interval $(t, t + dt)$ is $(\lambda_1\, dt)(\lambda_2\, dt) = (\lambda_1\lambda_2)\, dt^2$, and so the rate (probability per unit time) of simultaneous spikes is necessarily 0.

To define *correlated* Poisson processes, we exploit the fact that a superposition of two *independent* Poisson processes with rates λ_1 and λ_2, respectively, is again a Poisson process with rate $\lambda_1 + \lambda_2$. If S_1 and S_2 denote the corresponding spike trains, $S_1 + S_2$ is then used to denote the superposition.

We can now formulate our algorithm, referring to a population of n neurons. A *pattern* is a spike that occurs simultaneously in group $M \subseteq \{1, \ldots, n\}$ of neurons. For each of the possible $2^n - 1$ different nonempty patterns in such a population, we assume an independent Poisson process S_M with rate λ_M that governs the occurrence of the corresponding joint spikes. The spike train of neuron i is then exactly the superposition of all those pattern processes S_M that include neuron i:

$$S_i = \sum_{i \in M} S_M,$$

which has the rate

$$\lambda_i = \sum_{i \in M} \lambda_M.$$

Two neurons S_i and S_j have spikes in common if there are patterns M with $\{i, j\} \subseteq M$ that occur with nonzero rate λ_M. The two neurons will then be (positively) correlated. More generally, consider a group $G \subseteq \{1, \ldots, n\}$ comprising k neurons. If a pattern M exists such that $G \subseteq M$, all neurons of G will have spikes in common with rate $\sum_{G \subseteq M} \lambda_M$. In this case, correlations of order k are present (see Fig. 16.2). For details, see Chap. 12, this volume.

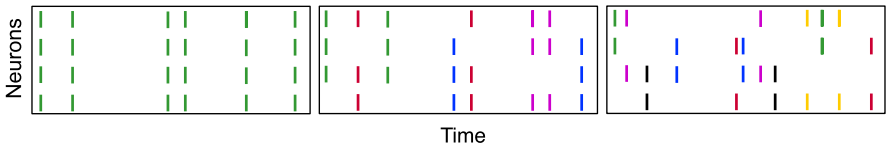

Fig. 16.2 Correlated multivariate Poisson processes (Staude et al. 2009). Shown are examples of an ensemble of four neurons, colors indicate different patterns. A subgroup of neurons exhibits a "higher-order correlation", if and only if appropriate patterns occur with nonzero rate. The size of patterns differs, and so do the pairwise correlation coefficients in the three examples shown. *Left*: pattern size 4, $\gamma = 1$; *center*: pattern size 3, $\gamma = 1/2$; *right*: pattern size 2, $\gamma = 1/6$

We also mention that the generation of correlated point processes with controlled properties is an important research topic, see, for example, Brette (2009), Krumin and Shoham (2009), Macke et al. (2009), Niebur (2007).

16.3 Renewal Processes

16.3.1 Ordinary Renewal Processes

In the previous section we have generalized the Poisson process by defining

$$h\big(x(t), \mathcal{H}(t)\big) = x(t),$$

and we called the resulting process the inhomogeneous Poisson process. Let us now assume that at time t the most recent spike of the neuron was fired at time $t - \tau$ and that

$$h\big(x(t), \mathcal{H}(t)\big) = h(\tau).$$

Such processes are called *renewal processes*, and it is justified to call τ the *age* of the neuron. In this case, the instantaneous firing rate depends only on the time elapsed since the most recent spike. For small dt, we have

$$\mathbb{P}\big[\text{spike in } (t, t+dt) \,\big|\, \text{last spike at time } t-\tau\big] = h(\tau)\,dt.$$

The function $h(\tau)$ is called the *hazard rate* of the renewal process. A typical example is the Poisson process with dead time: the neuron fires with constant rate like a regular Poisson process, provided that it did not fire a spike within a fixed dead time d.

An effective algorithm for the simulation of a renewal process can be devised, provided that

$$h(\tau) \le H \quad \text{for all } \tau \ge 0$$

for some finite bound H. This condition is satisfied for Poisson processes, Poisson processes with dead time, and also for a large class of other processes that are relevant for the applications (in particular, gamma-processes and log-normal processes, see Chap. 1, this volume).

The algorithm to generate surrogate data on the interval $[0, T]$ is the following:

Renewal process:

(1) The set of the target spikes s_i initially contains only $s_0 = 0$.
(2) Generate a Poisson random number P with mean HT.
(3) Place P random spikes t_i uniformly and independently on the interval $[0, T]$.
(4) Sort the spikes so that $t_i < t_{i+1}$.
(5) For every spike t_i draw an independent random number r_i uniformly distributed in $[0, 1]$.

For every reference spike t_i:

(6) Compute the age

$$\tau_i = t_i - \max\{s_k \colon s_k < t_i\}.$$

(7) Check whether

$$r_i < \frac{h(\tau_i)}{H}.$$

(8) If yes, add t_i to the list of target spikes.

Again, it can be shown easily that this algorithm indeed yields a renewal process with the desired hazard rate.

Renewal processes can also be simulated by summing up interspike intervals. But if the interspike interval distribution is not a standard one, there may be no fast and effective algorithm to sample from it. In contrast, the procedure we have just presented works for every bounded hazard rate. It needs only uniformly distributed random numbers, in addition to a single Poisson random number.

In the algorithm described above, the first interval has the same distribution as all the consecutive ISIs. Since the firing probability depends only on the age of the neuron, it has age 0 at the beginning of the simulation. In other words, we are simulating neuron that fired its last spike just before the observation started. This choice of the first interval distribution is arbitrary, and it generally implies that the process is not in equilibrium. In fact, the short-time statistics of a renewal process strongly depend on the choice of the initial distribution. Such dependencies do not exist for Poisson processes, since in that case the hazard rate does not depend on the neuron's age τ. For any reasonable ISI distribution, however, the age distribution will, in the course of time, converge to a uniquely defined equilibrium distribution. This equilibrium distribution of ages plays a special role, because it reflects the properties of a process that has been running for a long time, far away from any onset transients. A renewal process, where the first interval is sampled from this distribution, is called *renewal processes in equilibrium*. Later in this section, we will address the issue of generating surrogate data obeying the statistics of equilibrium renewal processes.

16.3.2 The Master Equation of a Point Process with Time-Dependent Hazard

Before we discuss how to simulate a renewal processes in equilibrium, we address a related topic. In the algorithm for the simulation of renewal processes described in the previous section, it is natural (and legitimate) to allow $h(\tau)$ to depend not only on the age τ, but also on time t. The algorithm then defines what is called a *renewal process with time-dependent hazard*, and one could wonder to which degree the resulting type of process can be characterized statistically. In particular, we want to predict the instantaneous firing rate of such a process. For stationary renewal processes, firing rate responses can be efficiently computed for arbitrary initial age distributions by methods involving Laplace transforms (see Chap. 1, this volume).

In the case of a time-dependent hazard this is much more difficult, and even if one simplifies the situation by assuming that the hazard factorizes according to $h(t,\tau) = \lambda(t)h_0(\tau)$, it is not possible to connect the statistics of the time-dependent process to the statistics of the renewal process with hazard $h_0(\tau)$ in a simple manner. For instance, it is not generally the case that the instantaneous rate of the nonstationary process is obtained through multiplying the instantaneous rate of the stationary one by $\lambda(t)$.

However, it is possible to write down the *master equation* for the age variable of the nonstationary equation (Gerstner and Kistler 2002). It is a partial differential equation which describes the time evolution of the age distribution $a(t,\tau)$,

$$\begin{aligned} \frac{\partial}{\partial t}a(t,\tau) &= -\frac{\partial}{\partial \tau}a(t,\tau) - h(t,\tau)a(t,\tau), \\ a(0,\tau) &= a_0(\tau), \\ a(t,0) &= \int_0^\infty h(t,\tau)a(t,\tau)\,d\tau. \end{aligned} \tag{16.4}$$

Here $a(t,\tau)$ is the probability density that the neuron has age τ at time t, i.e., that at time t its most recent spike was at time $t-\tau$. The terms on the right-hand side of the first equation are due to the ageing of the neuron and to the neuron being removed from the distribution because it spikes, respectively. The second equation represents the initial condition corresponding to the initial age distribution $a_0(\tau)$. The third equation is a boundary condition reflecting the reinsertion of neurons at age 0 after each spike. Therefore, the instantaneous firing rate is given by

$$f(t) = \int_0^\infty h(t,\tau)a(t,\tau)\,d\tau.$$

16.3.3 Renewal Processes in Equilibrium

The simulation method described above has both advantages and disadvantages in the case of time-dependent firing rates. If many observations (trials) of the same

experiment are available, it is, in principle, possible to estimate the time-dependent hazard and to come up with a model matching the data. The associate procedures are quite involved, though, and in practice it may be quite difficult to turn this into a reliable statistical procedure. To circumvent this problem, one can use the approach based on operational time described in Sect. 16.2.

For any given time-independent hazard rate, there exists an initial age distribution such that the rate profile is flat, the equilibrium distribution. In particular, for $h(t, \tau) = h(\tau)$, the equilibrium age distribution is the stationary solution of (16.4), given by

$$a_{\mathrm{eq}}(t) = \frac{1}{\mu} e^{-\int_0^t h(\tau)\, d\tau}, \tag{16.5}$$

where μ is the mean ISI. This leads us finally to the algorithm for simulating a renewal process in equilibrium.

Renewal process in equilibrium:

(1) Draw a random number X from the distribution (16.5).
(2) The set of the surrogate spikes s_i initially contains only $s_0 = X$.
(3) Generate a Poisson variable P with rate $H(T - X)$.
(4) Place P random spikes t_i $(i = 1, \ldots, P)$ uniformly on the interval $[X, T]$.
(5) Sort the spikes so that $t_i < t_{i+1}$.
(6) For every spike t_i, draw an independent random number r_i uniformly distributed in $[0, 1]$.

For every spike $i = 1, 2, \ldots$:

(7) Compute the age

$$\tau_i = t_i - \max\{s_k \colon s_k < t_i\}.$$

(8) Check whether

$$r_i < \frac{h(\tau_i)}{H}.$$

(9) If yes, add t_i to the set of surrogate spikes.

16.3.4 Nonstationary Renewal Processes and Operational Time

We already have observed in Sect. 16.2 that the map from real time to operational time transforms an inhomogeneous Poisson process into a homogeneous one. The same rationale can be applied to renewal processes. In fact, the inverse transformation can be employed to construct nonstationary renewal processes. For any given rate profile $\lambda(t)$, the transformation from real time to operational time is given by

$$\Lambda(t) = \int_0^t \lambda(s)\, ds.$$

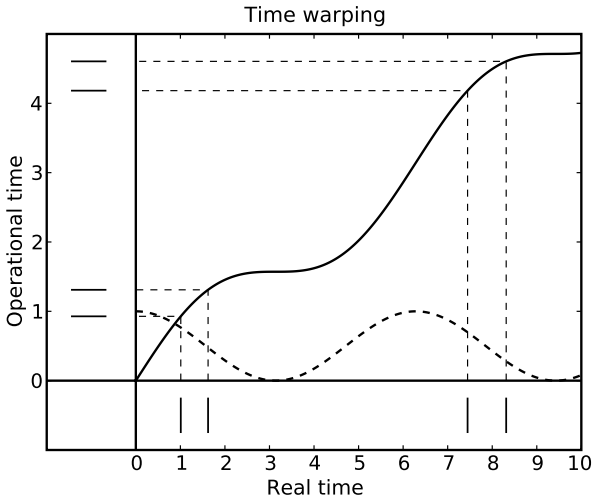

Fig. 16.3 General procedure to map stationary to nonstationary processes. Assume that the firing rate profile (*dashed line*) and a realization of a nonstationary process are given. Based on the rate profile, a nonlinear transformation from real time to operational time can be computed, based on the integrated rate function (*continuous line*). A stationary spike train in operational time is obtained by applying the transformation to every individual event time $s_i = \Lambda(t_i)$. Also the inverse operation can be realized, yielding an ensemble of nonstationary spike trains, the PSTH of which exactly matches the prescribed firing rate profile

The inverse transformation is obtained by

$$\Lambda^{-1}(s) = \min\{s' \colon \Lambda(s') \geq s\}.$$

What is the effect of this inverse transformation? Start with a renewal process s_i with unit rate, which is in equilibrium. It has a flat rate profile, i.e., the instantaneous firing rate is constant in time. Apply the above inverse transformation and obtain the process $t_i = \Lambda^{-1}(s_i)$. This process will retain some statistical properties of the original renewal process, but it will have a rate profile prescribed by $\lambda(t)$. This procedure is known as *time warping*. A visual explanation of this procedure can be found in Fig. 16.3. Below, a formal description of the algorithm is given.

Time warping:

(1) Draw spikes (s_i) for a renewal process in equilibrium.
(2) Compute $t_i := \Lambda^{-1}(s_i)$ employing, e.g., bisection or Newton's method.
(3) The target process is (t_i).

16.3.5 Operational Time and Real Neural Data

Operational time can be employed for neural data analysis (see Chap. 3, this volume, for applications). Assume that repeated trials of an experiment are performed, each

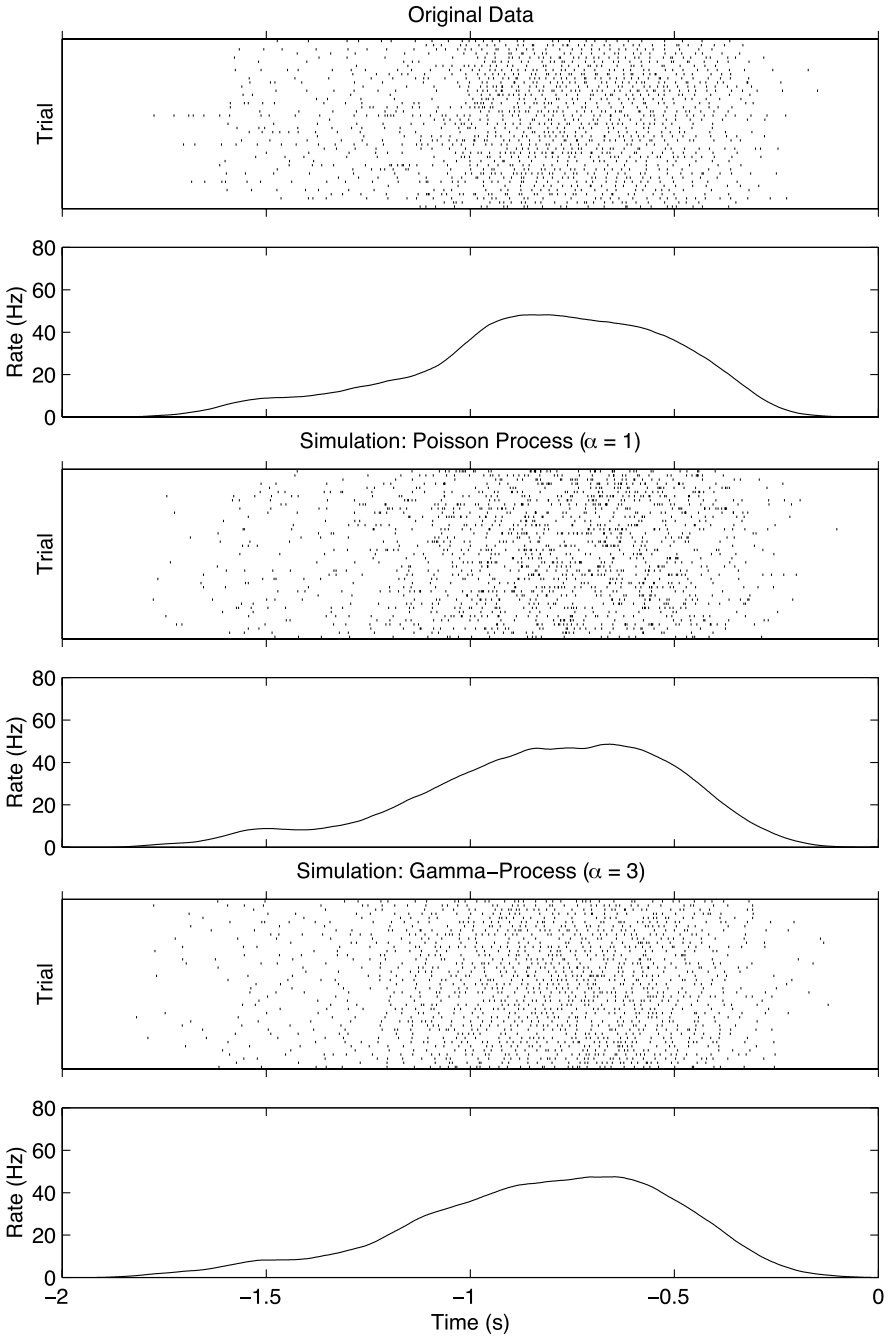

Fig. 16.4 Continued

Fig. 16.4 Spike trains recorded from a motor cortical neuron before the execution of an arm movement at time 0 (*panel 1*). The peristimulus time histogram (PSTH) estimated from 40 repeated trials shows a clear modulation of firing rate several hundred milliseconds before movement onset (*panel 2*). The firing rate profile extracted in this way was used to simulate 40 trials of a nonstationary Poisson process (*panel 3*), of which the PSTH was again extracted for control purposes (*panel 4*). Although the rate profile of the simulated Poisson ensemble is the same as the original apart from inevitable statistical fluctuations, the irregularity of individual traces does not match the original spike trains. This problem is solved by simulating a nonstationary gamma process with suitable order parameter α instead of a Poisson process (*panel 5*). This can again be done matching the original firing rate profile (*panel 6*). Data courtesy of A. Riehle, simulations by S. Grün

yielding a spike train of a given duration, such that the PSTH can be computed. If this analysis shows that spike trains are nonstationary, it is difficult to infer a point process to generate surrogate data directly based on the interspike intervals.

A possible solution is to map the recorded spikes to operational time, via the transformation $\Lambda(t)$. The point process is now stationary in operational time, and a renewal process (or a stationary nonrenewal process) can be identified based on the transformed spike data in operational time. Surrogate data are then obtained by mapping the simulated stationary spike trains from operational time back to real time, via the inverse transformation $\Lambda^{-1}(s)$. The method is explained with a concrete example in Fig. 16.4.

References

Brette R (2009) Generation of correlated spike trains. Neural Comput 21(1):188–215

Cox DR, Isham V (1980) Point processes. Monographs on applied probability and statistics, vol 12. Chapman & Hall, London

Gerstner W, Kistler WM (2002) Spiking neuron models: single neurons, populations, plasticity. Cambridge University Press, Cambridge

Krumin M, Shoham S (2009) Generation of spike trains with controlled auto- and cross-correlation functions. Neural Comput 21(6):1642–1664

Macke JH, Berens P, Ecker AS, Tolias AS, Bethge M (2009) Generating spike trains with specified correlation coefficients. Neural Comput 21(2):397–423

Niebur E (2007) Generation of synthetic spike trains with defined pairwise correlations. Neural Comput 19(7):1720–1738. PMID: 17521277

Ripley BD (1987) Stochastic simulation. Wiley, New York

Staude B, Rotter S, Grün S (2009) CuBIC: cumulant based inference of higher-order correlations. J Comput Neurosci. doi:10.1007/s10827-009-0195-x

Wikipedia, the free encyclopedia (2009) Exponential function. Online; accessed 28 December 2009

Chapter 17
Generation and Selection of Surrogate Methods for Correlation Analysis

Sebastien Louis, Christian Borgelt, and Sonja Grün

Abstract Generating artificial data from experimental data as a means for implementing a null hypothesis is becoming widely used. The reason is twofold: increasing computer power now allows for this type of approach, and it has become clear that the complexity of experimental data does not in general allow one to formulate a null hypothesis analytically. This is particularly true for the correlation analysis of parallel spike trains. Neglecting statistical features of experimental data can easily lead to the occurrence of false positive results, which of course needs to be avoided. Therefore surrogate data are used to generate the predictor by modifying the original data in such a way that the feature of interest (temporal coordination of spikes) is destroyed but other features of the data are preserved. The latter aspect is the most demanding and requires the selection of a surrogate type that best fits the data at hand. This chapter will demonstrate the need for such a selection and will show selection criteria.

17.1 Introduction

In the correlation analysis of experimentally recorded parallel spike trains, one has to thoroughly consider the statistical features of the data in order to prevent false positive results (Grün 2009). Typically, the complexity of the data prevents us from using analytical expressions for evaluating the significance of a correlation. Similarly, parametric tests presuppose models that are typically simplifications of the real neuronal data and thus may ignore important features. An alternative to these approaches is to use surrogate data, i.e., modified versions of the original data, to assess the significance (see Chap. 18). The objective of surrogate data generation is to leave all statistical features of the original experimental data intact, except

S. Grün (✉)
Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan
e-mail: gruen@brain.riken.jp
url: http://www.cnpsn.brain.riken.jp

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_17, © Springer Science+Business Media, LLC 2010

Table 17.1 Definition of example data sets used in our experiments. Each data set is composed of 40 trials of 1,000-ms duration for two neurons recorded simultaneously, with a time resolution of 1 ms. The complexity of the data sets is increasing from top to bottom; type 5 may be considered as a representative example of experimental data (see also Fig. 17.2 for a visualization)

Type	Process	Nonstat. in time	Cross-trial nonstat.	γ	Δ
1	Poisson	no	no	1	0
2	Poisson	yes	no	1	0
3	Poisson	no	yes	1	0.235
4	Poisson	yes	yes	1	0.5
5	Gamma	yes	yes	3	0.5

those we want to test for; these we wish to destroy. Stochastic modeling of neuronal spike trains is an alternative way to generate artificial data that may be used for significance evaluation (see Chap. 16). However, such approaches often imply model assumptions that may not be tolerated.

In this chapter we illustrate how surrogates can be used to estimate the presence and significance of spike correlation. The main issue is to avoid false positive (FP) results that originate from the destruction of statistical features inherent to the original data. If one is interested in the spike synchronization between neurons, one should aim at destroying the precise timing of the spikes, while conserving as far as possible all other features, such as the time course of the firing rate and interspike interval (ISI) statistics. In order to study the applicability of surrogates, we will define a number of example data sets exhibiting different statistical features found in typical experimental data. We then proceed to apply different surrogate methods to these data sets in order to explore how reliably spike correlation can be detected, in the form of false negative (FN) results, and how significant the danger is to get false positive (FP) results. This will be illustrated through the cross-correlogram (CCG) analysis method (Perkel et al. 1967) (see also Chaps. 5 and 6) before quantifying the performances of the various surrogates in the context of the Unitary Event analysis method (Grün et al. 2002) (see also Chap. 10). In the final section, we discuss the results and argue how to select the proper surrogate for a specific data set, and in particular which choice is suitable for the analysis of experimental data.

17.2 Example Data Sets

In this section we explain the generation and choice of the data types used for demonstrating the performance of the surrogate methods. Data are generated by stochastic models exhibiting controlled statistical properties similar to those found in experimental data. The properties are: nonstationarity of firing rates in time, cross-trial nonstationarity of firing rates (parameter Δ), and (ir)regularity of the spike trains (parameter γ). The data sets are generated as renewal point processes following a given rate profile (for details, see Chap. 16). Table 17.1 shows the chosen parameter values and combinations, explained in the following.

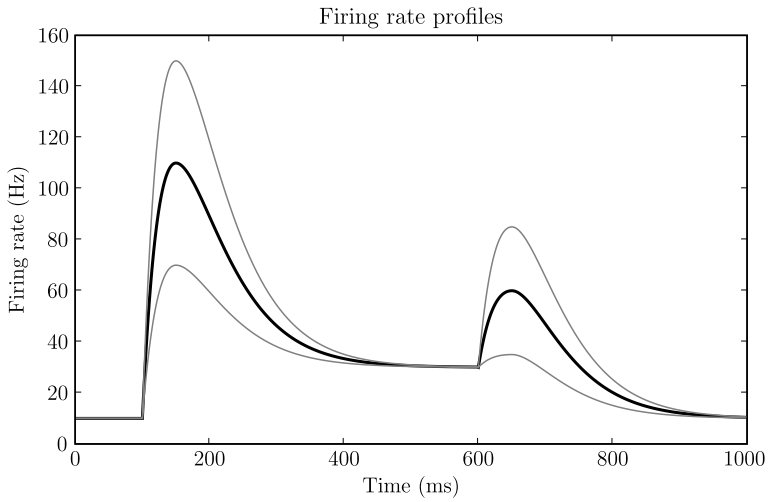

Fig. 17.1 Rate profiles for data sets with nonstationary firing rates. The *thick black line* represents the reference profile. This rate profile is used for all trials for both neurons in the case of cross-trial stationarity (data set of type 2). For data sets including cross-trial nonstationarity (type 4 and 5), the rate profile per trial is chosen randomly from two profiles. These differ from the default profile by an enhanced or reduced (modulation factor $\Delta = 0.5$ in *gray*) amplitude of the response peaks. The mean profile averaged across trials corresponds to the reference profile

In case of stationary firing rates in time (types 1 and 3), the spike trains are generated according to a constant rate of $\lambda = 37.75$ Hz, insuring an average spike count consistent with richer data sets. Data sets with nonstationary firing rate in time (types 2, 4, 5) are generated according to rate profiles that mimic a phasic-tonic rate response (starting at 100 ms after trial onset at 0 ms), followed by an off-response (beginning at 600 ms) (Fig. 17.1). The two rate modulations are modeled as sums of two exponential functions with different amplitudes (Nawrot et al. 1999).

In case of absence of cross-trial nonstationarity (type 2), all trials of both neurons are modeled according to the profile shown in black in Fig. 17.1, called reference profile. Cross-trial nonstationarity is then introduced by a modulation of the maxima of the reference profile by a factor Δ to derive a pair of profiles with higher and lower rates. The strength of this modulation is quantified by $0 \leq \Delta \leq 1$, defining for the high profile, the peak heights relative to the peak heights of the reference profile: $\mathrm{Max}_{\Delta,1,2} = \mathrm{Max}_{\mathrm{ref},1,2}(\Delta + 1)$, where Max is the maximum relative to the base firing rate (see Fig. 17.1, e.g., gray curves for $\Delta = 0.5$). The higher rate profile is obtained by increasing the function amplitudes, and the lower profile is reconstructed by exploiting the fact that the reference profile is the mean of the two. Thus, the average rate (summed over trials) is conserved. For a given Δ, the rate profile is selected by randomly choosing one of two profiles per trial. Since we aim to study the worst case scenario, which is given by trial-by-trial rate covariation of the neurons (see Grün et al. 2003 for details), we select trial-by-trial the same rate profile for both neurons.

For comparability, the trials of data type 3 (stationary within but nonstationary across trials) are created with a constant rate corresponding to the mean rate of the

upper and the lower profile leading to a modulation factor of $\Delta = 0.235$ (compared to $\Delta = 0.5$ in the nonstationary case).

To account for the (ir)regularity of the spike trains, we model these as Gamma processes (Chap. 1). The shape factor γ allows for the generation of spike trains with arbitrary coefficients of variation (CVs; see Chap. 3). In data sets of types 1–4 we model the spike trains as Poisson processes ($\gamma = 1$), which have in the stationary case a CV of 1. To account for the fact that experimental spike trains may exhibit more regular spike trains (Nawrot et al. 2007, 2008; Shinomoto et al. 2009), we include a data set with regular spike trains ($\gamma = 3$; type 5). The spike trains are initially generated as unit rate processes with a chosen γ in operational time. The resulting processes are then mapped to real time through the integrated rate profile, producing rate and interspike interval modulated spike trains (Chap. 16). The mapping from real time (t) to operational time (τ) is given by $\tau = \int_0^t \lambda(t)\, dt$, where $\lambda(t)$ is the rate profile (Chap. 3).

Table 17.1 shows the parameter values and combinations selected for our example data types that will be used in the following. The complexity of the data types increases from type 1 to type 5. The most complex data set (type 5), exhibits nonstationarity within and across trials, in addition to a shape factor of 3 (in operational time), introducing regularity in the spiking activity. The visualization of this data type in Fig. 17.2 clearly shows the strong cross-trial variability of spike counts and a nonexponential ISI distribution.

The reference data sets are generated as "independent" spike trains without correlated spiking on a finer time scale than the rate covariation. To create data sets that include spike synchronization, we generate the data types of Table 17.1 as background activity and additionally insert coincident spike events at a rate λ_c in the order of 1 to 3 Hz homogeneously into the spike trains (see details in Grün et al. 1999). The coincident spikes are inserted with a small temporal jitter (± 1 ms) to mimic experimental data. From the data sets that include injected synchrony we examine the FN rates and compare how sensitive the different surrogate methods are.

17.3 Surrogate Generation

In order to demonstrate the impact of surrogate schemes on correlation analysis, we examine six surrogate generation methods, which are commonly used in the literature (for a review, see Grün 2009) and cover many issues that need to be considered: trial shuffling, spike time randomization, spike train dithering, spike time dithering, joint-ISI dithering, and spike exchanging (see the top part of Table 17.2).

Some other surrogate methods are also included in Table 17.2 for completeness. ISI shuffling within and across trials (Nádasdy et al. 1999; Masuda and Aihara 2003; Ikegaya et al. 2004), for example, will have similar effects as spike time randomization and trial shuffling respectively, the former flattening the rate profile to a stationary one, and the latter ignoring cross-trial nonstationarities. A localized form of

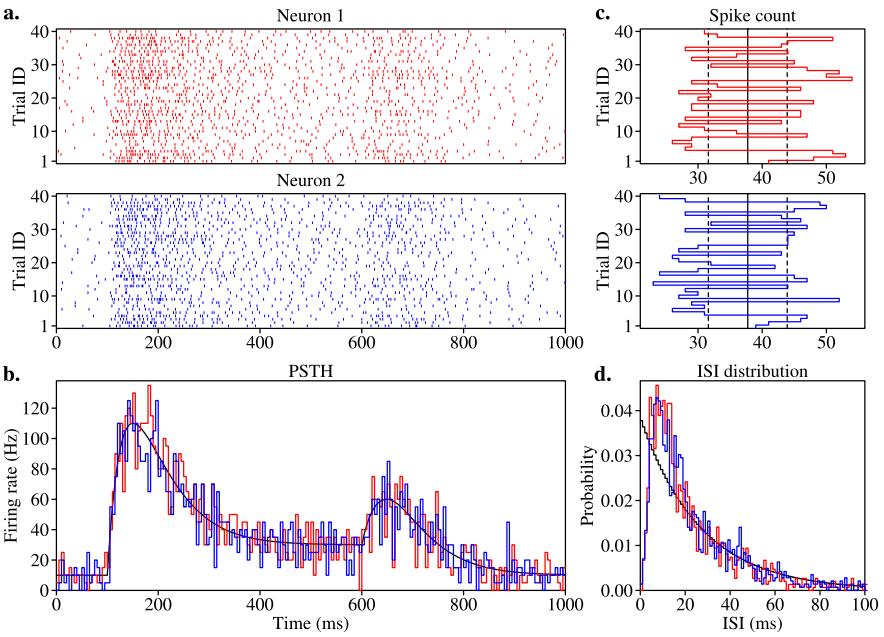

Fig. 17.2 Visualization of a type 5 data set. **a**. Dot displays of the two simulated neurons; **b**. Peri-stimulus time histograms (PSTHs, bin width 5 ms) of each of the neurons; **c**. Spike counts per trial. *Solid lines* mark the mean spike count across trials, and *dashed lines* mark ± one standard deviation for the case in which counts were produced by a stationary Poisson process of identical mean firing rate, as a reference for the observed variability. The observed variability is due to the fact that the count distribution is in fact bimodal, being a mixture of two count distributions with different means. **d**. Interspike interval (ISI) distributions of both neurons. Note the deviation from the exponential distribution corresponding to stationary Poisson processes with the same average firing rate (*black*). Colors correspond to the two neurons

within-trial ISI shuffling would constitute an interesting additional surrogate. However, it would be difficult to compare the "locality" parameter of the shuffle with the dither width of dither-based methods. Cross-neuron spike shuffling (within the same trial) has also been used in the analysis of experimental data (Nádasdy et al. 1999; Ikegaya et al. 2004) but is only applicable to population recordings. For two neurons, exchange of spikes does not destroy correlation.

The shift predictor (Gerstein and Perkel 1972) was one of the earliest surrogate methods proposed, gaining over trial shuffling through the possibility to accommodate for slow drifts in firing rates across trials. However the cross-trial nonstationarity exhibited by data sets 3, 4, and 5 is random: each trial rate profile is drawn independently from one of two profiles. Thus the performance of the shift predictor is expected to be very similar to trial shuffling.

In the following we briefly describe the generation of the surrogates and the features they preserve, together with Python routines (sometimes simplified in a pseudocode manner) by which we implemented them.

Table 17.2 List of surrogates with execution times. The first six methods are those being compared in this book chapter. Durations are expressed as factors relative to the duration of trial shuffling, which is the fastest method for generating surrogates. Values are rounded to integer multiples of 0.5. *Sp-exg* runtime depends on the data as explained in the text

Surrogate type	Abbreviation	Req. time
trial shuffling	*tr-shu*	1
spike time randomization	*sp-rnd*	2.5
spike train dithering	*tr-di*	1.5
spike time dithering	*sp-di*	6
joint-ISI dithering	*jisi-di*	7
spike exchange	*sp-exg*	7–10
ISI shuffling (within trials)		
ISI shuffling (across trials)		
spike shuffling (across neurons)		
shift predictor		

The input to any surrogate generation method is a list $\mathtt{TS} = [\mathtt{T}_0, \mathtt{T}_1, \ldots, \mathtt{T}_{N-1}]$ of trials for one neuron. Each trial consists of a list of spike times, that is, $\mathtt{T} = [\mathtt{t}_0, \mathtt{t}_1, \ldots, \mathtt{t}_{n-1}]$ with $0 \leq \mathtt{t}_i < \mathtt{tmax}$. Here `tmax` is the trial duration, which we assume to be the same for all trials. A surrogate data set is generated from the trials of one of the two neurons in each data set, while the trials of the other neuron are kept intact. This is sufficient to destroy correlations on a chosen time scale between the spike trains (Pazienti and Grün 2007). For correlation analysis approaches including more than two parallel spike trains, typically more than one of the spike trains has to be modified.

Figure 17.3 illustrates each method schematically and is meant to clarify what the pseudocodes are effectively implementing. We now proceed to consider each surrogate method separately and begin with trials shuffling.

17.3.1 Trial Shuffling

Trial shuffling (*tr-shu*) is a well-established way of destroying the relationship between simultaneously recorded spike trains. The trials of one of the neurons are randomly permuted, so that each trial is no longer paired with the corresponding trial of the other neuron, but with a randomly selected one. Provided that it is reasonable to assume that possible coincidences occur at different times in different trials, trial shuffling can be expected to effectively destroy all existing spike correlations across neurons (Gerstein and Perkel 1972). However, the internal structure of the spike trains, including time dependent variations of parameters within trials, is left intact.

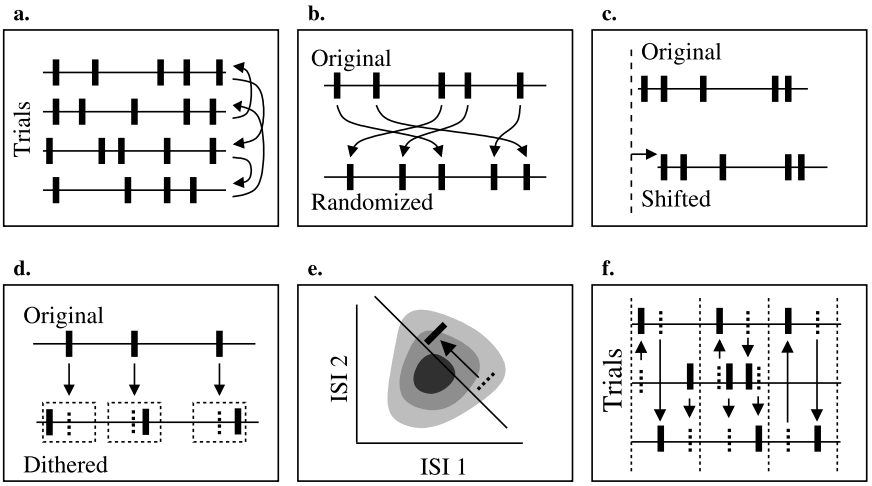

Fig. 17.3 Illustration of the surrogate methods examined. **a**. Trial-shuffling (*tr-shu*), which randomizes trial IDs. **b**. Spike time randomization (*sp-rnd*). **c**. Spike train dither (*tr-di*) relative to the trial onset (*dashed line*). **d**. Uniform spike dithering (*sp-di*) with dither range (box). **e**. Joint-ISI dithering (*jisi-di*) according to the joint-ISI distribution. **f**. Spike exchange across trials (*sp-exg*) within predefined windows (*dotted lines*)

Trial shuffling is easily implemented with the built-in Python function `shuffle` (available in the package `random`) which permutes the elements of a list:

```
from random import shuffle
def TrialShuffling(TS):
    ouTS = [T for T in TS]  # copy the top level, i.e., the list of trials
    shuffle(ouTS)           # shuffle the copied list of trials
    return ouTS             # return the shuffled copy
```

Following the # signs are comments indicating the function of the code. Note that the trial set `TS` given as an argument is copied, because `shuffle` works in place and thus modifies its argument. This is a fundamental difference to MatLab, where all arguments to functions are passed "by value". That is, in MatLab a function receives only a copy of any object passed to it, and any changes made to an argument inside a function do not have any effect outside it. Python, however, uses "call by value" only for elementary data types (like integer and real-valued numbers) and "call by reference" for all more complex data types (like lists and tuples). That is, a list passed to a Python function is only a reference (or pointer) to this list, and thus the same list that is accessible outside the function. As a consequence, any changes made to the list inside the function are visible outside.

The advantage of "call by reference" is much higher efficiency, because unnecessary copying is avoided. The disadvantage is that one has to be more careful when calling functions: they may modify the arguments passed to them. Likewise, when

writing functions, one should strive not to modify arguments that are passed by reference, as these changes will be reflected outside the function.

Here the function `shuffle` modifies its argument, and thus applying it directly to the trial set `TS` would change it, and this change would be visible outside the function `TrialShuffling`. Hence `TS` has to be copied—although it suffices to copy its top level, as no changes are made to the individual spike time lists. Note that MatLab would always copy both levels, thus introducing unnecessary overhead.

The copying is done with a list comprehension (creating a list from an existing list), a very powerful feature of the Python language, which makes handling lists and tuples very easy and efficient. A list comprehension is a statement enclosed in square brackets, indicating that a list is to be formed (round parentheses indicate a tuple—an "unchangeable" list). The statement consists of an expression describing the list elements and an iterator (here a **`for`** loop over the elements of `TS`), which provides the range of values for which the expression is to be evaluated. Each iteration generates one list element.

The created copy `ouTS` of the trial set `TS` is then shuffled and returned. Note that the original argument `TS` is left unchanged and thus not affected outside the function. Trial shuffling is clearly the fastest surrogate generation method, since it does not go down to the level of the spike times (only the top level of the list is copied and shuffled). To convey an idea of the speed of the other methods, we state their execution times relative to this method as summarized in Table 17.2.

17.3.2 Spike Time Randomization

Randomizing the times at which spikes occur (*sp-rnd*) is another very simple surrogate generation method, in which the number of spikes in a trial is preserved, but the spike times are randomly chosen in the trial duration, that is, in the interval [0, `tmax`[.

Spike time randomization is easily implemented with the built-in Python function `sample` (also available in the package `random`), which produces a sample of given size, without replacement, from a provided list or range. Here we sample `len(T)` spikes times from the trial duration, that is, from the interval [0, `tmax`[:

```
from random import sample
def SpikeTimeRandomization (TS):
    return [sample(xrange(tmax),len(T)) for T in TS]
            # for each trial: sample as many new spike times as there are spikes
```

Note how list comprehension makes it possible to write this procedure as a single expression: for each trial `T`, a sample of `len(T)` spike times is drawn from the range [0, `tmax`[. These samples are combined into a list and then returned.

Spike time randomization is a relatively fast surrogate generation method, but about 2.5 times slower than trial shuffling, mainly due to the need of generating many (pseudo)random numbers for the sampling.

17.3.3 Spike Train Dithering

In order to destroy possibly existing coincidences, but conserve firing rate and ISI properties, one may consider spike train dithering (*tr-di*). In this method an entire spike train of one neuron is shifted relative to that of the other neuron by a uniformly distributed amount of time (Pipa et al. 2008; Harrison and Geman 2009). To be more precise, for each trial, a random value is sampled from a given interval (symmetric around 0, like [−w, w], with the width w to be specified by a user), by which all spike times are then shifted. Interspike intervals are fully maintained, and firing rates are smoothed on the timescale of the dither width.

We implement spike train dithering with the built-in Python function `randint` (also available in the package `random`) to generate the random shift value:

```
from random import randint
def SpikeTrainDithering (TS,w):     # TS: trial set, w: shift window size
    D=[randint(-w,+w) for T in TS] # create dither values—one per trial
    return [[(t+d+tmax)%tmax for t in T] for d,T in zip(D,TS)]
                                    # shift/roll spike times in each trial
```

Note how nested list comprehensions and the function `zip` make it possible to write this procedure as two simple expressions. The first expression generates a different shift value for each trial. In the second expression these shift values are paired with the trials using the built-in Python function `zip`. This function returns a list of tuples (here specifically: pairs), the ith element of which consists of the ith elements of each argument list. These individual elements are then accessed by `d` (shift value) and `T` (trial). In the inner list comprehension, the shift value is added to each spike time `t`. Note here that, in order to conserve the net time overlap between the trials of the two neurons and thus not to underestimate the expected coincidence count, we actually roll the spike train. That is, spikes that are shifted out of the train at one end are shifted in at the other end (implemented by using the new spike time modulo the trial duration `tmax`). This is acceptable for our data sets, because the rate profiles are such that start and end rates are the same. In general, however, this may not be the case, and a readjustment of the analysis time window may be necessary.

Spike train dithering is almost as fast as trial shuffling, taking only about 1.5 times as long to execute in Python. It is faster than spike time randomization, because it has to generate much fewer (pseudo)random numbers.

17.3.4 Spike Time Dithering

Spike dithering (also known as “jittering” or “teetering”) was introduced in (Date et al. 1998) and has been used in many studies (Abeles and Gat 2001; Hatsopoulus et al. 2003; Gerstein 2004; Shmiel et al. 2006; Jones et al. 2004; Maldonado et

al. 2008; Butts et al. 2007). Spike time dithering (*sp-di*) randomly displaces each individual spike within a small time window around its original position. We prefer to use the term dithering to indicate the manipulation of the data in contrast to the temporal jitter of joint spike events found in the experimental data.

The dither probability distribution within the time window is typically chosen to be uniform or Gaussian. We will restrict ourselves to the former as it is more common in the literature. This procedure destroys the exact timing of the spikes and, in consequence, the temporal relations between spikes of simultaneously observed neurons (Pazienti and Grün 2007; Pazienti et al. 2008).

To implement spike time dithering, we use the `randint` function again and replace each spike time `t` by `randint(t-w,t+w)`. To make sure that the dithered spike still falls into the interval [0, `tmax`[, we bound the acceptable range between 0 and `tmax`-1:

```
from random import randint
def SpikeTimeDithering (TS,w):      # TS: trial set, w: dither window size
    return [[randint(max(t-w,0),min(t+w,tmax-1))
                for t in T] for T in TS]
                # for each spike in each trial: sample a new time in dither window
```

Note again how list comprehensions make it possible to write this procedure in one line. The lower bound of the sample window is `max(t-w,0)`, where `t` is the old spike time and `w` the window size, to make sure that any new spike time is ≥ 0, and the upper bound is `min(t+w,tmax-1)` to make sure that any new spike time is $<$`tmax`. A new spike time is sampled for each spike time `t` in each trial `T`. Note, however, that the above code is a simplification of our actual implementation, because it does not ensure that all dithered spikes are distinct, inducing a possible loss of spikes. For reasons of simplicity, we skip the technical details of how we achieved distinct dithered spikes without losing spikes and without a prohibitively large execution time penalty.

Spike time dithering is one of the slower surrogate generation methods, taking (in the more sophisticated version that guarantees distinct spike times) about 6 times as long as trial shuffling but does not take prohibitively long to execute.

17.3.5 Joint Interspike Interval Dithering

While in spike time dithering the new position of a spike is sampled from a local uniform distribution centered at the old spike time, joint interspike interval dithering (*jisi-di*) restricts the new spike time in such a way that it falls in between the preceding and the following spikes. That is, the sampling window is the intersection of the dithering window and the window spanned by the preceding and succeeding spikes. Furthermore, the sampling distribution is no longer uniform but is chosen in such a way that it reflects the distribution of the lengths of pre- and post-spike intervals as it is present in the original spike trains. More precisely, Gerstein (2004)

suggested to dither spikes according to the square-rooted joint-ISI (JISI) histogram for adjacent intervals. Dithering a spike on such a two-dimensional surface corresponds to a movement along a trajectory perpendicular to the principal diagonal (see Fig. 17.3e).

Since our actual Python code is somewhat lengthy (due to several technical details arising from the added complexity of the method), the following is a simplified version, which spells out the key steps of the procedure:

```
def JISIdithering (TS,w):        # TS: trial set, w: dither window size
    M = sqrt(JISI(TS))           # take square root of joint-ISI distribution
    ouTS = []                    # initialize the output trial list
    for T in TS:                 # traverse the trials, expanded by pseudo-spikes
       U   = [-1]+T+[tmax]       # to unify handling of first and last interval
       ouT = set()               # initialize the output set of spike times
       for i in xrange(1,len(U)-1): # traverse the spikes of the trial
          S = slice of M corresponding to U[i+1]-U[i-1]
          truncate S to [max(U[i]-w,0),min(U[i]+w,tmax-1)]
          o = U[i]+x with x distributed according to S
          ouT.add(o)             # dither the spike and add it to the output
       ouTS.append([t for t in ouT])
                                 # add the output set as a list to the trial list
    return ouTS                  # return the created dithered trial set
```

The matrix slice referred to in the above code contains those matrix elements, for which the sum of the row and column indexes equals `U[i+1]-U[i-1]` (the time difference between the pre- and the post-spikes and thus the sum of the lengths of the pre- and post-spike intervals). These elements are located on a diagonal that is perpendicular to the main diagonal of the matrix `M`.

In our actual Python code the JISI matrix is smoothed with a Gaussian kernel in order to avoid problems that could result in sparsely populated parts of the matrix. In addition, sophisticated preprocessing takes place, which extracts the matrix slices beforehand and transforms them into cumulative distribution functions (one for each possible value of `U[i+1]-U[i-1]`), from which one can sample much more quickly. Finally, additional code tries to avoid that the same spike time is chosen more than once (inducing spike loss). The idea is to repeat a sampling if it has yielded a spike time that already exists in the output and to drop the spike only if several repetitions failed.

Due to various optimizations, which are mainly concerned with preprocessing the joint interspike interval distribution, this method is (in our implementation) not much more costly than simple spike time dithering. It takes about 7 times as long as trial shuffling, compared to a factor of 6 for normal spike time dithering.

17.3.6 Spike Exchanging

Spike exchanging (*sp-exg*) splits all spike trains into exclusive windows of a user-specified size and then reassigns the spike times within the windows across the trials

(Harrison et al. 2007; Smith and Kohn 2008). That is, a spike can only be moved to a time at which a spike occurs in at least one trial. In this process both the number of spikes per trial window and the number of spikes per time bin are preserved. Necessarily, this places fairly tight restrictions on the possible reassignments, which sometimes make it difficult to find a spike assignment that differs (significantly) from the one in the original data.

As for *jisi-di*, we present here a simplified version of our Python code, because the actual implementation is again, due to certain technical subtleties, somewhat lengthy.

```
def SpikeExchanging (TS,w):          # TS: trial set, w: window size
    ouTS = [[] for T in TS]          # initialize the output trials
    for i in xrange(0,tmax,w):       # traverse the time windows
        W,C = [],[0]                 # initialize the spike time and offset lists
        for T in TS:                 # traverse the trials
            X = [t for t in T if i <= t and t < i+w]
            W.extend(X)              # add spikes of trial T in current window
            C.append(C[-1]+len(X))   # note offset of last spike per trial
        conflict = True              # execute the following loop at least once
        while conflict:              # while no valid reassignment found
            shuffle(W)               # shuffle spikes (reassign spikes to trials)
            conflict = False         # default: no conflict in reassignment
            for j in xrange(len(TS)):
                                     # traverse the trials
                if len(set(W[C[j]:C[j+1]])) != C[j+1]-C[j]:
                    conflict = True  # if duplicate spike times in a trial,
                    break            # the reassignment is invalid, so repeat
        for j in xrange(len(TS)):    # append reassigned spikes to the output
            ouTS[j].extend(W[C[j]:C[j+1]])
    return ouTS                      # returned the reorganized spike trains
```

Intuitively, the above code generates permutations of a list of all spikes in a window `[i*w,(i+1)*w[` and checks whether the sections that refer to a trial do not contain duplicates. If this is the case, the permutation specifies a valid reassignment of the spikes to the trials and thus is appended to the output. Otherwise a new permutation is generated and tested, until a valid permutation is found.

In the actual implementation, the splitting into windows is achieved in a better (faster) way than in the above code (all windows are generated in a single traversal of the trials and stored in a matrix), and the permutation generation is only executed a very limited number of times. After that, if no valid permutation could be found, a more sophisticated procedure is executed, which is slower but much more likely to produce a valid reassignment. The basic idea is to collect all spikes that occur at the same time, because such spikes must be assigned to different trials and thus are the reason for invalid reassignments. Counters for the number of spikes already assigned to a trial keep track of the trials further spikes may still be assigned to, so that the number of spikes a trial has in a window is not changed. The spikes are reassigned by processing the occurring spike times in decreasing order of their frequency. This

order increases the chances that enough trials are still available to receive another spike. Nevertheless, even this procedure may fail, and thus it is repeated only up to a limit, and if no valid reassignment is found, the original assignment is preserved. However, in our experiments we never observed that this was necessary.

Spike exchanging is the most costly surrogate generation method, and its execution time depends strongly on the data: usually it takes (in our implementation) between 7 and 10 times as long as trial shuffling. Nevertheless, compared to the other methods, these times are still bearable and thus not an argument against the application of this method.

17.4 Correlation Analysis

In this section we show how surrogates can be used in testing for the presence of correlation between spike trains and for deriving its significance. This will also include a study showing how well various types of surrogates perform in this task. We begin with demonstrations in the context of the cross-correlation analysis and then compare the surrogate methods by concentrating on coincident events and their significance as done in the Unitary Events analysis (Grün 2009; Grün et al. 2002).

17.4.1 Cross-Correlogram Analysis

The cross-correlogram (CCG) is a histogram of spikes of one neuron relative to the spike times of another neuron (for details, see Chaps. 5, 6). Thus the CCG contains coincidence counts as a function of the time difference τ of joint-spike events. It can be computed on arbitrary time scales by changing the bin width h. In Fig. 17.4a, the black line shows the CCG on $h = 1$-ms resolution obtained from a data set of type 5. It shows a broad, symmetrical peak around zero delay which lasts approximately ± 100 ms. Superimposed onto this one observes a very fine peak (a few ms wide) centered at $\tau = 0$ ms, which results from the inserted coincidences ($\lambda_c = 2$ Hz with 1-ms jitter). The broad peak is due to the nonstationarity in the firing rates of the neurons which leads to a trivial increase in joint-spike events. Typically the aim of cross-correlation analyses is to differentiate temporally coordinated spike events from chance coincidences produced by (covarying) firing rates. This is done by comparing the empirical CCG to a predictor which conserves the firing rates. The simplest way to do so is to make use of surrogate data.

CCGs computed from different types of surrogate data are shown in Fig. 17.4a (colored curves). The thick and fine lines represent the mean and the mean plus two standard deviations, respectively. Such a confidence limit is sometimes used as a threshold to detect significant correlations between neurons (Berger et al. 2007). We observe that the threshold (representing a significance level of approximately 5%) is exceeded at several time lags through random fluctuations, as expected by multiple testing.

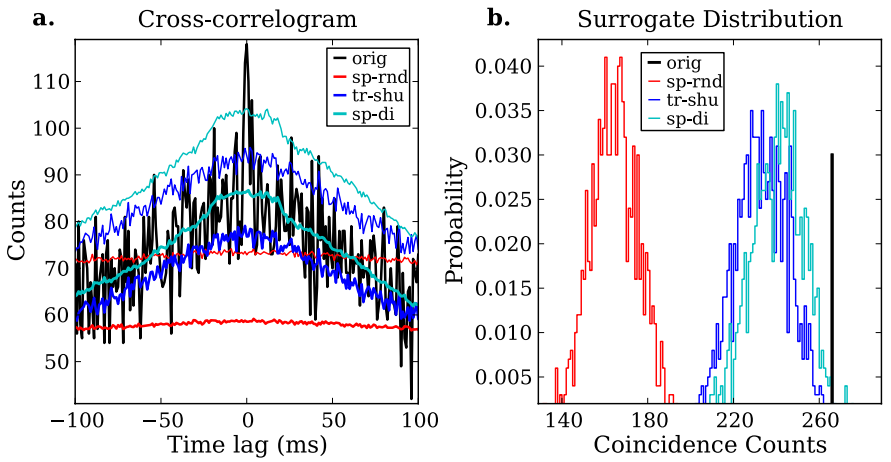

Fig. 17.4 Surrogate cross-correlograms (CCGs) and coincidence distributions. **a**. Comparison of CCGs of original and surrogate data. The CCG of an original data set of type 5 (40 trials, $\Delta = 0.5$, $\gamma = 3$, $\lambda_c = 2$ Hz, jitter $= \pm 1$ ms) is shown in *black*. Surrogates generated by trial shuffling (*tr-shu*, *blue*), spike time randomization (*sp-rnd*, *red*), and spike dithering (*sp-di*, generated with a ± 20 ms dither window; *cyan*) are shown as the mean over 1,000 surrogates (*thick lines*) and the mean plus twice the standard deviation over the surrogates (*fine lines*). The slight curvature seen in the CCG for *sp-rnd* is due to a smaller number of samples at larger time lags. **b**. Estimating significance. The coincidence count distribution constructed from 1,000 surrogates allows for the estimation of the significance (p-value) of the empirical coincidence count (*thick black vertical line*) of the original data (type 5) with a jittered (± 1 ms) injection of coincidences at a rate of $\lambda_c = 0.5$ Hz. Significance of the empirical count is derived using the same three surrogates as in **a**. To account for the temporal jitter of the coincidences, we use as test statistics the sum of coincidences up to a ± 1 ms time lag

Obviously, the different surrogates differ in the shapes of their CCGs. Some, such as those generated by *tr-shu* (blue) or *sp-di* (cyan), follow the shape of the original CCG quite closely. In contrast, *sp-rnd* (red) leads to a total destruction of the original shape and produces a flat histogram. This is due to the fact that *sp-rnd* reduces an inhomogeneous process to a homogeneous one, removing all rate fluctuations. Using this surrogate method to detect precise temporal correlation in such a data set would be completely inadequate, as even the broad peak itself (in the absence of excess synchrony) becomes highly significant (i.e., above the two standard deviations threshold), leading to FP results. In consequence, this surrogate method should only be used when firing rates are stationary within trials.

The *sp-di* method (± 20 ms) in contrast follows the original CCG closely, except for the narrow peak around 0 ms time lag. This demonstrates a basic feature of the spike dithering approach. Although the dithering somewhat smooths the original rate profile, the slow components are nonetheless preserved which is reflected directly in the CCG. The center peak of the original CCG exceeds the two-standard-deviation threshold (corresponding to a p-value of 5%) and would convincingly qualify the pair as exhibiting excess synchrony.

The trial shuffling of the data (*tr-shu*) destroys the cross-trial covariation in firing rates between the two neurons. This means trials of one neuron following the upper rate profile are occasionally paired with trials of the other neuron following the lower profile. This leads to a lower predicted number of chance coincidences than by using *sp-di* (the blue CCG has a smaller peak than the cyan one). Here again the method would detect the excess synchrony as significant, but the result is now questionable. More extreme cases of nonstationarity across trials may lead to a significant result even though no excess synchrony is present.

The other three surrogate schemes we consider in this chapter are generated on a time resolved, trial-by-trial basis and thus conserve rate modulations. Their resulting CCGs are not distinguishable from the spike dither CCG (not shown here).

The above discussion shows how useful the CCG is in visualizing the effect a surrogate method may have on the original data. Thus we would recommend as a starting point, before proceeding to more advanced analyses, to view the resulting CCG. This could help in immediately ruling out an inadequate method.

17.4.2 Significance of Spike Coincidences

To quantify the applicability of surrogates for significance estimation of spike correlation, we concentrate now on the entries of the central peak of the CCG, i.e., spike coincidences. We count the total number of coincidences n_{emp} in the original data (test statistic) and decide whether they occur significantly more often than expected by chance given other features of the data. To estimate the p-value of the empirical count, we compare it to the coincidence count distribution derived from the surrogate data which has destroyed fine temporal correlation. This basically constitutes the Unitary Event (UE) analysis approach (for details, see Chap. 10 and Grün et al. 2002). However UE analysis is applied in a time-resolved manner to resolve the dynamics of correlation and can also be used to detect the correlation between more than two simultaneously recorded neurons. We focus here on pair coincidences. As the excess synchrony is injected homogeneously, we consider our data as one time segment.

Under specific conditions (homogeneous Poisson process), one can express the coincidence count distribution for the significance estimation analytically as a Poisson distribution with a mean given by the expected coincidence count derived as the product of the firing probabilities of the neurons. However, if the data do not fulfill such conditions, one should instead use surrogate data to generate the distribution. From the repetitive creation of surrogates we can construct a coincidence count distribution. The vertical black line in Fig. 17.4b marks the empirical coincidence count n_{emp} derived from a data set of type 5. It is compared to the distributions of coincidence counts (n_{surr}) derived from different surrogate types. Obviously, these distributions differ in their mean and width. In all the three cases shown (same methods as illustrated for the CCG in Fig. 17.4a), the excess spike synchrony ($\lambda_c = 0.5$ Hz) present is detected as significant for a significance level of 0.05. However, as observed in the CCG analysis, the *sp-rnd* method, which is neglecting the modulation

of the firing rates, severely underestimates the expected number of coincidences. The distribution generated by trial shuffling is shifted to higher coincidence counts, and for the dithering, it is even shifted a little more to the right. From our previous CCG analysis we know that spike dithering accounts best for the changes in the firing rates, out of these three surrogate methods. Consistent with this, its distribution is the most conservative.

17.4.3 Performance of Surrogates

To evaluate the true performance of the surrogates, we now look at FP and FN rates with data sets of type 5, as they contain many typical features of experimental data. For each parameter configuration and surrogate method, the FP and FN rates are obtained as follows. We begin by generating 1,000 data sets with the same parameter configuration. For each data set, we produce 1,000 surrogate versions. From these surrogate versions we construct a surrogate coincidence count distribution and calculate the p-value of the original coincidence count. Thus, for each parameter configuration and surrogate method, we obtain 1,000 p-values (p_i for $i = 1, \ldots, 1{,}000$). Given a significance level α, which we fix at 0.01 in the following, we convert these results into counts of positive (significant) results, i.e., $N_+ = \sum_i \varphi(p_i \leq \alpha)$, and counts of negative (nonsignificant) results, i.e., $N_- = \sum_i \varphi(p_i > \alpha)$, where $\varphi(x) = 1|0$ if x is true|false. If the chosen parameter configuration involves injected synchrony, then the FN percentage is given by $N_-/N \cdot 100\%$ (where $N = N_+ + N_- = 1{,}000$), i.e., the percentage of falsely undetected correlation. Conversely, if the parameter configuration does not involve injections, then the FP percentage reads $100 \cdot N_+/N$ indicating the percentage of falsely detected correlation. We found that this level of repetition provided us with stable estimates of FN and FP rates (standard deviation in the order of 1%).

In Fig. 17.5 we explore the impact of specific parameter configurations on the occurrence of FN and FP results, over all surrogate methods. We start with Fig. 17.5a, in which the percentage of FN results is shown as a function of the degree of cross-trial nonstationarity Δ of the firing rates. The general anticipated trend is for FN rates to decrease as the strength of the nonstationarities increases, at least for surrogate methods which smooth the rate profile within trials or across trials. This is due to the fact that the chance contributions originating from neglected rate modulations increase and thereby "reinforce" the impact of injections. The effect is quite noticeable for *tr-shu* and *sp-exg*. *Tr-di* is the most conservative method and keeps high FN rates throughout, between 10% and 12%. The *sp-di* and *jisi-di* also show a weak dependence on Δ, although they are less conservative. As expected, *sp-rnd* has 0% FN throughout, which is due to the overall destruction of the rate modulation in the surrogates, leading to an underestimation of chance coincidences.

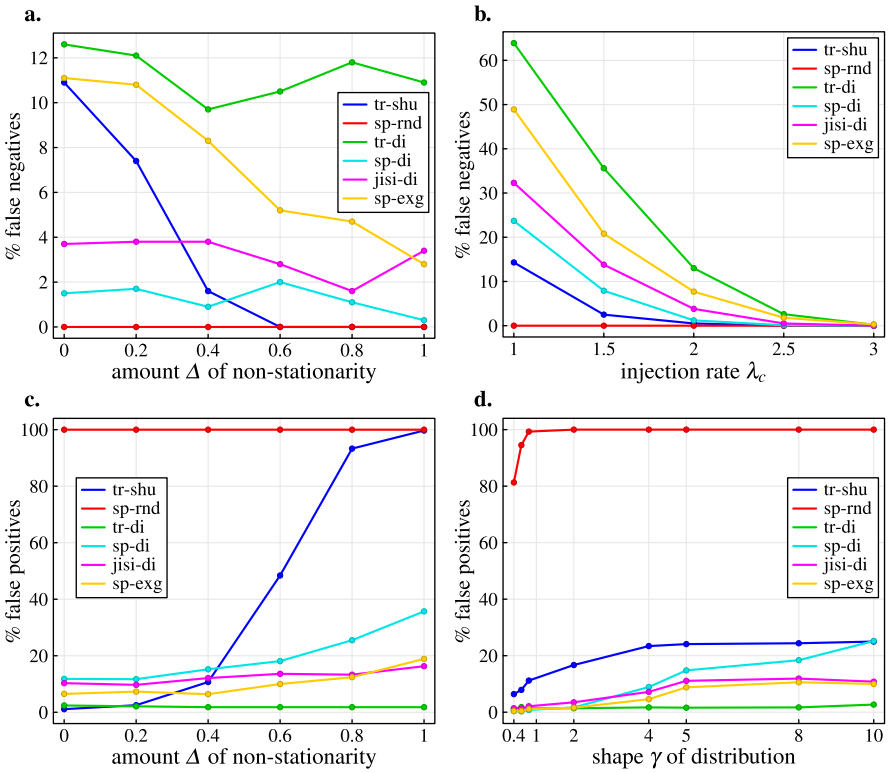

Fig. 17.5 FP and FN rates as functions of data parameters for different surrogate types. *All four panels* are based on data sets of type 5, whose parameters were systematically varied. Surrogate types are the same as introduced in Sect. 17.3, and the injected synchrony was jittered by ± 1 ms. The significance level was set to 1%. **a**. Percentages of FN as a function of the degree of cross-trial nonstationarity parameterized by Δ. The injection of coincidences is homogeneous and at a constant rate of $\lambda_c = 2$ Hz, and γ is set to 3. **b**. FN percentages as a function of coincidence injection rate (homogeneously injected). Other parameters are set constant to $\Delta = 0.5$ and $\gamma = 3$. **c**. FP percentages are shown as a function of cross-trial nonstationarity parameterized by Δ, γ is set to 3, $\lambda_c = 0$ Hz. **d**. FP percentages as a function of the regularity of the spike trains expressed by the shape factor $\gamma = 0.2$ to 10 (not equidistant), $\Delta = 0.5$, and $\lambda_c = 0$ Hz

Not surprisingly, we find for the same surrogate 0% FN rate throughout various injection rates (from 1 Hz to 3 Hz) and fixed $\Delta = 0.5$ (Fig. 17.5b). For the other surrogate types, the FN rate decreases with increasing injection rate due to an increase of signal-to-noise (chance coincidence) level. We find again that *tr-di* is the most conservative (lowest FP rate), followed by *sp-exg*, *jisi-di*, *sp-di*, and *tr-shu* in order. Above an injection rate of 3 Hz, all surrogate methods reliably detect the presence of excess synchrony.

The initial surrogate ordering of FN rates found in Fig. 17.5b is reversed in the FP rates observed in Fig. 17.5c for high Δ. The breakdown of the *tr-shu* method is evident, rapidly worsening once $\Delta > 0.4$. Interestingly, it performs better than

other methods for small cross-trial nonstationarities ($\Delta < 0.2$), with the exception of *tr-di*. This is because *tr-shu* does not erode the rate profile like other dithering methods do. *Sp-di* shows to be quite susceptible to sharp rate transitions, provided that the dithering window is large enough. We clearly see a pronounced increase in FP rates for this method, nearly reaching 40% at $\Delta = 1$, corresponding to a first rate burst of 190 Hz. The *jisi-di* method is much more robust, keeping an FP rate below 20%. This can be explained by the fact that for the same dither range, the *jisi-di* method does not displace spikes as much as the uniform method (*sp-di*), as it follows the square-rooted JISI profile and can have forbidden regions within the range. The *sp-exg* method is more noticeably affected by the increase in Δ, starting at 7% and ending just below 20%. The train dither on the other hand is the most conservative, seemingly unaffected by the strength of the modulations.

This is also the case in Fig. 17.5d, in which we plot the FP rate as a function of the shape parameter γ (from 0.2 to 10) used in the generation process with fixed $\Delta = 0.5$. This corresponds to a coefficient of variation (CV) lying between 0.3 and 1.6. *Tr-di* remains in the FP range of 1% to 4%, whereas all other methods see their FP rates increasing as the processes become more regular. This is understandable: the strength of the rate modulation is kept constant (first burst either 150 Hz or 70 Hz), while the regularity is increased. In these high-rate regions, regularity will produce a wider coincidence count distribution (Pipa et al. 2010), and if the surrogate method destroys this regularity, FP results are to be expected. This is the case for the *sp-di* method whose performance again worsens for $\gamma > 2$, nearly reaching 30% for $\gamma = 10$. Even though it is designed to conserve the ISI distribution, the performance of the *jisi-di* dither method also deteriorates with regularity. The reason is that we are applying it to an inhomogeneous process, for which the ISI distribution is not as meaningful. This ISI distribution is modulated by the rate profile, within and across trials, and thus a single JISI distribution produced over all times and trials cannot be an adequate reference. A potential solution to this problem is to first map the spike train to operational time, where it becomes stationary (Louis et al. in press). The *sp-exg* method performs relatively well even though it does not conserve the ISI distribution, ending at FP rates of 12% for $\gamma = 10$. The spike randomization method rapidly reaches 100% FP rates, the level also found in Fig. 17.5c. Finally, on a smaller scale, the *tr-shu* method progressively worsens, even though the method conserves the overall ISI distribution. This points to the observation that the effects of rate mismatch are amplified in the presence of spiking regularity.

The results discussed so far described the limitations of various surrogate methods with respect to parameter variations of the most complex data type. How they perform in the face of different data types is the subject of the next section.

17.4.4 Suitability of Surrogates for Different Data Sets

Next, we test the suitability of our surrogate methods for data having specific features (see Table 17.1). As before, we generate 1,000 data sets for each data type with

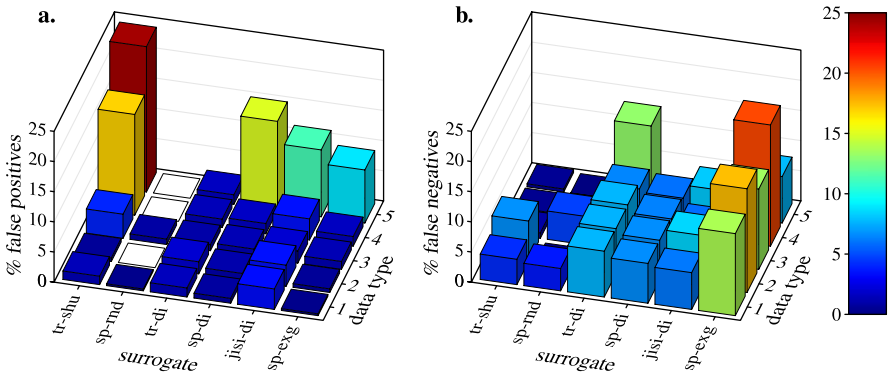

Fig. 17.6 False positive (**a**) and false negative (**b**) percentages for all tested surrogate methods across the five different data types listed in Table 17.1. *Colors* code FP and FN percentages. *White squares* mark the position of bars (not shown for convenience) of 100% FP

the parameters given in Table 17.1. For each data set, we produce 1,000 surrogate versions of one type and calculate the percentage of FN and FP.

Figure 17.6 shows the FP and FN results for all combinations of our data and surrogate types. The results for data type 5 are identical to the ones shown in Fig. 17.5 for the corresponding parameter configurations. In addition, we find here the results for the less complex data sets (types 1–4), giving a better understanding of the parameters relevant to FP and FN rates.

We first discuss the FP rates and observe that *sp-rnd* is strictly inadequate for data sets which exhibit nonstationary firing rates in time, since it produces 100% FP rates. For stationary data sets, the FP rate is very low, and thus *sp-rnd* would be an acceptable method in such a setting.

Tr-shu provides some surprising results. Even though data type 3 is nonstationary across trials, the underestimation of coincidences is much weaker than for the data set with firing rate modulation within the trial (type 4). The reason is that in the latter case the firing rate differences are locally (in particular at rate peaks) much larger than in the stationary case. Adding regularity in the data set (type 5), FP rates increase even further to 24%. Thus, the underestimation of chance coincidences is even larger for destroyed rate covariance combined with regularity of the spike trains, as concluded at the end of the previous section.

A similar type of dependence is observed for the *sp-exg* method, although the final rise of FP rates in type 5 is much more acceptable. It remains at a level of 1% until regularity is included (type 5), at which point it increases to 9%. The deletion of the original ISIs by spike exchange induces FPs. An analogous effect is also seen with *sp-di*, which performs reasonably well for all data types except the one including regularity (type 5). The study by Pipa et al. (2010) has shown that the coincidence distribution of independent Gamma processes is typically wider than the one of its dithered version. Consequently, if the latter is used as a reference, the chance for FPs is enhanced. More surprising is that for *jisi-di*, the FP rate also increases for data type 5, although in this case the regularity is kept. As argued in

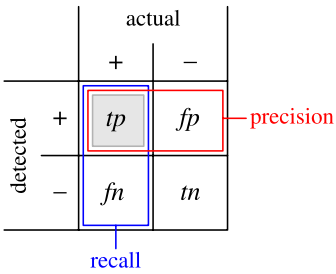

Fig. 17.7 Table illustrating the classification probabilities with the extracted precision and recall

the previous section, the JISI is an average over different firing rates and thus does not constitute a valid reference.

In contrast, *tr-di* leads to very low FP rates and, in particular, for regular processes. Thus the preservation of the ISI sequence and the local firing rates avoids an increase of FPs, although there is some rate smoothing involved due to the train dither. In consequence, train dithering seems to be by far the most robust method.

One can easily see the reverse situation for the FN rates in Fig. 17.6b. The *tr-shu* and *sp-rnd* methods show very low FN rates for the complex data sets as the erroneous contributions to the detection of excess synchrony are higher. The *tr-di* method is consistently conservative across data types (around 7%), in particular, for type 5, where it reaches a 13% FN level. However, the most conservative method to detect spike synchrony is the *sp-exg* method, with FN rates reaching 20% (the maximum across all combinations of surrogate methods and data types) for Poisson processes which are nonstationary within and across trials (type 4). This may be explained by the limited effect of spike permutations across trials in high-rate regions. *Sp-di* sees a fall in FN rates from 6% to 1% in type 5 presumably due to erroneous coincidence contributions. The *jisi-di* method operates at an FN rate around 6% throughout all data types and thus constitutes an acceptable method.

To summarize these results, we now use a common method in information retrieval. First we define the false positive probability fp that a data set is negative (no injection) and that it is detected as significant. This being a joint probability, we can use the above FP probability FP, which is conditional on the data set being negative, and write $fp = \frac{1}{2}\mathrm{FP}$. Similarly, the false negative probability fn is given by $fn = \frac{1}{2}\mathrm{FN}$, where FN is the false negative probability obtained above. Finally we define the true positive probability tp that a data set is positive and is detected as significant. It is given by $tp = \frac{1}{2} - fn$, as the probability that a data set is positive $p(+) = tp + fn = \frac{1}{2}$. The precision π and the recall ρ of a classifier (van Rijsbergen 1979) are then given by

$$\pi = \frac{tp}{tp + fp} \quad \text{and} \quad \rho = \frac{tp}{tp + fn}.$$

The precision corresponds to the probability that a data set detected as significant is indeed a positive data set (accuracy), while the recall is the probability that a positive data set is detected as significant (sensitivity). These quantities are illustrated in Fig. 17.7.

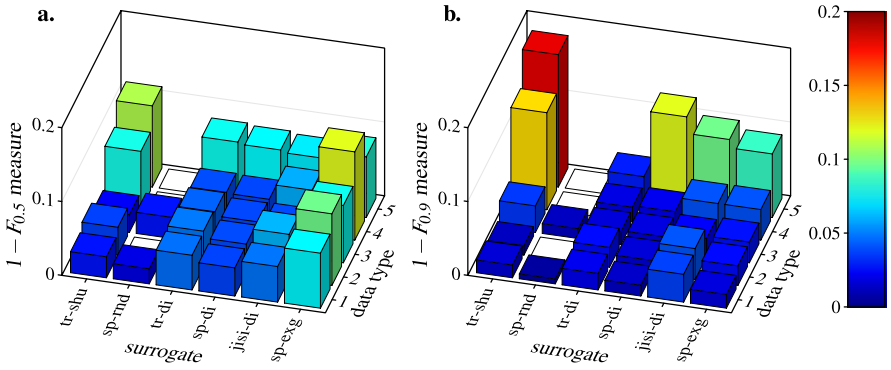

Fig. 17.8 F_β measures extracted from the bar charts in Fig. 17.6. The measure is calculated as explained in the text, from the precision and the recall properties of the surrogates. $F_{0.5}$ corresponds to an equal weighting of precision and recall, whereas $F_{0.9}$ places more importance on the precision. The lower $1 - F_\beta$ is, the better the surrogate

A standard way to combine precision and recall into one number is to compute the F_β measure (van Rijsbergen 1979), which is the weighted harmonic mean of precision and recall

$$F_\beta = \frac{1}{\frac{\beta}{\pi} + \frac{1-\beta}{\rho}}, \quad \beta \in [0, 1].$$

With $\beta = 0.5$, one attaches equal importance to precision and recall (Fig. 17.8). However, if presented with a single experimental data set, one would naturally tend to place more importance on the precision. If a hypothesis is being tested for on this data set, an inconclusive result (FN) is preferable, over the mistaken validation of this hypothesis (FP). A possible weighting would then be $\beta = 0.9$ (Fig. 17.8), setting the emphasis on precision.

We find that for $\beta = 0.5$, the three dither-based surrogate methods (*tr-di*, *sp-di*, and *jisi-di*) are the most reliable, with minor differences across data types. However, once the emphasis is put on precision ($\beta = 0.9$), *tr-di* is by far the most adequate method, showing robust detection across data types. The other dither methods show a similar performance up to data type 5, at which point their F_β shows a sharp decrease. Adding regularity to the processes severely impairs all methods, with the exception of *tr-di*.

17.5 Discussion

What is a good surrogate? Surrogates implement a null hypothesis and should be as far as possible from the alternative hypothesis. In our case we want to test if spike synchrony at a fine temporal scale exists (alternative hypothesis), such that it may not be trivially explained by the firing rates and the spiking regularity of the neurons (null hypothesis). The analysis should then proceed by generating surrogates

that have the same firing rate and ISI statistics as the original data, but in which the spike trains of the neurons are independent on this fine temporal scale. As the results presented above suggest, different surrogate methods conserve different features of the original data. For example, *sp-rnd* only conserves the spike count per trial, destroying any rate modulation and non-Poissonian interval statistics. In consequence, it places very stringent requirements on the original data in order to itself be a meaningful surrogate method. On the other hand, *sp-di* aims at conserving the rate profile, all the while destroying the potential correlation present on the time scale of interest. It has become a very widely used surrogate generation method, but as we observed above, it also has its limitations. Because the spikes are randomly displaced within a fixed dither range, whether they are in high or low firing rate regions, the rate profile is necessarily smoothed. The gravity of the smoothing effect will depend both on the strength of the rate fluctuations found in the data and the chosen dither width. Thus, for slowly varying firing rates, it can be considered adequate, but this again is only true if the ISI statistics do not differ too strongly from that of a Poisson process.

Considering all the simulation results presented above, it is clear that the safest surrogate method proposed so far is the *tr-di* method. It has shown to be highly robust to, if not unaffected by, both the strength of rate transients and the regularity of the spiking activity. FP rates remained between 1% and 2% for all data types, while the FN rates were at 7% before increasing to 13% for data type 5. These levels are satisfactory and could be explained by the exact conservation of the rate profile and ISI statistics. The superiority of the method is further illustrated by its high F_β measure, as soon as emphasis is put on precision (low FP) instead of recall (low FN), that is, $\beta > 0.5$. The main feature of this surrogate method which could raise some questions is the fact that for each surrogate data set, the whole profile is uniformly shifted. So for some large shifts, the effective difference with the original profile, integrated over time, could be considerable. But a type of process still remains to be found for which this full shift would lead to severe increases in FP and FN rates. Thus, based on the study presented in this chapter, we recommend the use of the *tr-di* surrogate method to detect excess synchrony and other types of fine time scale correlation in experimental data.

Of course, we did not cover all the surrogate methods available in the literature, since this was not the goal of this contribution. We rather wanted to demonstrate how to generate surrogates and wanted to deliver criteria on how to select the proper surrogate for the data at hand. Thus we carefully selected a representative set of surrogates that address a wide range of features present in data. For example, we did not explicitly treat the shift predictor method (Gerstein and Perkel 1972) since it is a special case of the trial shuffling method, involving a shift of trial ids. Another example is that we did not explicitly treat is the ISI-shuffle method, where the interspike intervals are either shuffled in time or across trials (Denker et al. 2007; Ito 2007). Instead, we treated a more advanced method aimed at conserving the JISI distribution. Recently it was shown that the transformation of the spike trains to operational time before applying a dither to the whole spike train preserves the rate and the ISI distribution perfectly and leads to an FP rate that

corresponds to the applied significance level (Louis et al. in press). We also left out some surrogate methods designed for spike pattern analysis in many parallel spike trains, involving cross-neuron spike exchanges (Ikegaya et al. 2004; Nádasdy et al. 1999). A recent review on data driven analysis methods discusses a large body of surrogate methods together with their assumptions and their applicability (Grün 2009).

The amount of execution time required for the generation of surrogates is an issue, in particular, if repetitively performed for the calculation of distributions and estimation of significance. In such cases one may consider the implementation of the numerical calculations in a parallelized fashion, for example, each data point could be calculated in parallel on a different computer. This leads to considerable savings in execution time and can easily be implemented (see Chap. 20 for a general introduction with Python and MatLab code).

The conclusion of this study is that the surrogate method needs to be carefully selected, taking into account both the statistical hypothesis at hand and the features present in the data. For example, testing for the existence of spatio-temporal firing patterns across neurons has particular requirements with respect to treating the ISI statistics of the data to avoid false positive outcomes (Abeles and Gat 2001; Gerstein 2004). Thus, we strongly suggest to carefully test the behavior of the surrogates in the context of the hypothesis to be tested for using artificial data, before applying them to experimental data. The artificial data should be designed such that their statistical features are as close as possible to the experimental data and requires a reliable *estimation* of these features.

The Python source code that we used in our experiments can be downloaded at http://www.borgelt.net/surrogates.html or http://www.apst.spiketrain-analysis.org/.

Acknowledgements We thank Michael Denker for technical assistance. Partially supported by the Helmholtz Alliance on Systems Biology.

References

Abeles M, Gat I (2001) Detecting precise firing sequences in experimental data. J Neurosci Meth 107(1–2):141–154

Berger D, Warren D, Normann R, Arieli A, Grün S (2007) Spatially organized spike correlation in cat visual cortex. Neurocomputing 70:2112–2116

Butts DA, Weng C, Jin J, Yeh C, Lesica NA, Alonso J, Stanley GB (2007) Temporal precision in the neural code and the timescales of natural vision. Nature 449:92–95

Date A, Bienenstock E, Geman S (1998) On the temporal resolution of neural activity. Technical report. Division of Applied Mathematics, Brown University

Denker M, Roux S, Timme M, Riehle A, Grün S (2007) Phase synchronization between LFP and spiking activity in motor cortex during movement preparation. Neurocomputing 70:2096–2101

Gerstein GL (2004) Searching for significance in spatio-temporal firing patterns. Acta Neurobiol Exp (Wars) 64(2):203–207

Gerstein GL, Perkel DH (1972) Mutual temporal relationships among neuronal spike trains. Statistical techniques for display and analysis. Biophysical J 12(5):453–473

Grün S (2009) Data-driven significance estimation of precise spike correlation. J Neurophysiol 101:1126–1140 (Review)

Grün S, Diesmann M, Grammont F, Riehle A, Aertsen A (1999) Detecting unitary events without discretization of time. J Neurosci Meth 94(1):67–79
Grün S, Diesmann M, Aertsen A (2002) Unitary events in multiple single-neuron spiking activity. I. Detection and significance. Neural Comput 14(1):43–80
Grün S, Riehle A, Diesmann M (2003) Effect of cross-trial nonstationarity on joint-spike events. Biol Cybern 88(5):335–351
Harrison MT, Geman S (2009) A rate and history-preserving resampling algorithm for neural spike trains. Neural Comput 21(5):1244–1258
Harrison MT, Amarasingham A, Geman S (2007) Jitter methods for investigating spike train dependencies. Computat Systems Neurosci Abstracts **3**(17)
Hatsopoulus N, Geman S, Amarasingham A, Bienenstock E (2003) At what time scale does the nervous system operate?. Neurocomputing 52:25–29
Ikegaya Y, Aaron G, Cossart R, Aronov D, Lampl I, Ferster D, Yuste R (2004) Synfire chains and cortical songs: temporal modules of cortical activity. Science 304(5670):559–564
Ito H (2007) Bootstrap significance test of synchronous spike events a case study of oscillatory spike trains. Stat Med 26:3976–3996
Jones LM, Depireux DI, Simons DJ, Keller A (2004) Robust temporal coding in the trigeminal system. Science 304(5679):1986–1989
Louis S, Gerstein GL, Grün S, Diesmann M (in press) Surrogate spike train generation through dithering in operational time. Front Comput Neurosci
Maldonado P, Babul C, Singer W, Rodriguez E, Berger D, Grün S (2008) Synchronization of neuronal responses in primary visual cortex of monkeys viewing natural images. J Neurophysiol 100:1523–1532
Masuda N, Aihara K (2003) Duality of rate coding and temporal coding in multilayered feedforward networks. Neural Comput 15:103–125
Nádasdy Z, Hirase H, Czurkó A, Csicsvari J, Buzsáki G (1999) Replay and time compression of recurring spike sequences in the hippocampus. J Neurosci 19(21):9497–9507
Nawrot M, Aertsen A, Rotter S (1999) Single-trial estimation of neuronal firing rates: from single-neuron spike trains to population activity. J Neurosci Meth 94:81–92
Nawrot M, Boucsein C, Rodriguez-Molina V, Riehle A, Aertsen A, Rotter S (2007) Serial interval statistics of spontaneous activity in cortical neurons in vivo and in vitro. Neurocomputing 70:1717–1722
Nawrot M, Boucsein C, Rodriguez-Molina V, Riehle A, Aertsen A, Rotter S (2008) Measurement of variability dynamics in cortical spike trains. J Neurosci Meth 169:374–390
Pazienti A, Grün S (2007) Bounds of the ability to destroy precise coincidences by spike dithering. Advances in brain, vision, and artificial intelligence. Lecture notes in comput sci, vol 4729. Springer, Berlin, pp 428–437
Pazienti A, Maldonado P, Diesmann M, Grün S (2008) Effectiveness of systematic spike dithering depends on the precision of cortical synchronization. Brain Research 1225:39–46
Perkel DH, Gerstein GL, Moore GP (1967) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophys J 7(4):419–440
Pipa G, Grün S, van Vreeswijk C (2010) Impact of spike-train autostructure on probability distribution of joint-spike events. Neural Comput (under revision)
Pipa G, Wheeler DW, Singer W, Nikolić D (2008) NeuroXidence: reliable and efficient analysis of an excess or deficiency of joint-spike events. J Comput Neurosci 25(1):64–88
Shinomoto S, Kim H, Shimokawa T, Matsuno N, Funahashi S, Shima K, Fujita I, Tamura H, Doi T, Kawano K, Inaba N, Fukushima K, Kurkin S, Kurata K, Taira M, Tsutsui K, Komatsu H, Ogawa T, Koida K, Tanji J, Toyama K (2009) Relating neuronal firing patterns to functional differentiation of cerebral cortex. PLoS Comput Biol 7(5):12591–12603
Shmiel T, Drori R, Shmiel O, Ben-Shaul Y, Nádasdy Z, Shemesh M, Teicher M, Abeles M (2006) Temporally precise cortical firing patterns are associated with distinct action segments. J Neurophysiol 96(5):2645–2652
Smith MA, Kohn A (2008) Spatial and temporal scales of neuronal correlation in primary visual cortex. J Neurosci 28(48):12591–12603
van Rijsbergen CJ (1979) Information retrieval. Butterworth, London

Chapter 18
Bootstrap Tests of Hypotheses

Valérie Ventura

Abstract Simulation-based calculation of p-values is an important technique in situations where it is difficult to obtain an exact or approximate distribution for a test, or when such an approximation exists but is of dubious validity, either because the conditions it requires are not met, or because it relies on questionable assumptions about the distribution of the data. But even in applications where we are fairly confident in a particular parametric model and the statistical analysis based on that model, it can still be helpful, in the spirit of robustness, to see what can be inferred from the data without particular parametric model assumptions. A substantial literature has demonstrated both theoretically and in numerical studies that the bootstrap is widely effective (Davison and Hinkley in Bootstrap methods and their applications. Cambridge University Press, Cambridge, 1997; Efron and Tibshirani in An introduction to the bootstrap. Chapman and Hall, New York, 1993). But the simplicity of the bootstrap conceals an important point: its properties depend on how resampling is done; arbitrary shuffles of the data do not necessarily accomplish desired statistical goals. Moreover, in the context of hypotheses testing, the p-value must be obtained under the hypothetical reality imposed by the null hypothesis. In this chapter, we review the general framework for statistical tests of hypotheses, and introduce the basics for Monte Carlo, permutation, and bootstrap tests.

18.1 Tests of Hypotheses

Suppose that we want to know if some stimulus increases the firing rate of a neuron whose baseline rate is 10 Hz. A sensible approach proceeds as follow. We present the stimulus n times. For each trial $i = 1, \ldots, n$, we record y_i, the number of times

V. Ventura (✉)
Statistics Department and the Center for the Neural Basis of Cognition, Carnegie Mellon University, 5000 Forbes avenue, Baker Hall 132, Pittsburgh, PA 15213, USA
e-mail: vventura@stat.cmu.edu
url: http://www.stat.cmu.edu/~vventura

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_18, © Springer Science+Business Media, LLC 2010

that the neuron spikes in, say, one second. The sample average $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the observed mean firing rate of the neuron under presentation of the stimulus. Then we compare $\overline{y}$ to the baseline firing rate of 10 Hz. Consider the following two hypotheses:

The null hypothesis: the stimulus induced firing rate is 10 Hz.
The alternative hypothesis: the stimulus-induced firing rate is greater than 10 Hz.

If $\overline{y}$ is *much larger* than 10 Hz, we will reject the null hypothesis and conclude that the evidence supports the alternative hypothesis. This is an example of a test.

The difficult part is deciding what *much larger* should be. Should we favor the alternative hypothesis if $\overline{y}$ exceeds 10 Hz by more than 1 Hz, 2 Hz, 3 Hz? A small cut-off will make it easy to reject the null hypothesis in favor of the alternative, which is desirable if the latter is true, but not otherwise. Rejecting the null hypothesis when it is true is called a *type 1 error*. Conversely, a large cut-off will make it hard to favor the alternative hypothesis, which is desirable if the null is true, but not otherwise. Retaining the null hypothesis when it is false is a *type 2 error*. However, whatever cut-off we choose and decision we subsequently make about rejecting or retaining the null hypothesis, we cannot know with certainty if our decision is correct. Indeed, the true stimulus induced firing rate of the neuron is unknown. A statistical test of hypotheses is a method to choose the cut-off so that the probability of erroneously rejecting the null hypothesis (a type 1 error) is below some probability α, say 1% or 5%.[1] A test is performed as follows:

Test of hypotheses procedure

1. Set the **null** and **alternative hypotheses** H_0 and H_A.
2. Choose a **test statistic** T, a function of the data that helps determine if the data support H_0 or H_A.
   Denote by t_{obs} the value of T in the observed sample.
   Specify the values of T that provide evidence against H_0 in favor of H_A.
3. Determine the **null distribution** of T, the distribution of the values of T when H_0 is true.
4. Choose the **significance level** α, the chance you are prepared to take of committing a type 1 error.
5. Compare t_{obs} to the null distribution of T: values of t_{obs} that are unlikely under the null distribution give evidence in favor of H_A.
   This comparison is summarized by the ***p*-value**, the probability of observing t_{obs} when H_0 is true, or a value that would give even more evidence than t_{obs} in favor of H_A.

[1] At the end of this section, we show that in this example, the cut-off, also called critical value, is approximately 11.04 for $\alpha = 5\%$.

6. Make your decision: reject H_0 in favor of H_A if $p \leq \alpha$. The probability of erroneously rejecting H_0 is less than α.

Consider the previous example. The spike counts are in Table 18.1. We set H_0: $\mu = \mu_0$ and H_A: $\mu > \mu_0$, where μ is the stimulus-induced mean firing rate of the neuron, and $\mu_0 = 10$ Hz its baseline rate. Functions of the data that can help us choose between H_0 or H_A are estimators of μ, such as the sample mean, trimmed mean, and median. We use the former as the test statistic. Values of $T = \overline{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i$ that are larger than $\mu_0 = 10$ Hz provide evidence against H_0 in favor of H_A. The value of T in the sample is $t_{\text{obs}} = \overline{y} = 10.92$; it is larger than 10 Hz, but is it large enough to cast doubt about H_0? We answer this by determining what sort of values T would take if H_0 was true, summarized by the null distribution of T. It is often too difficult to derive analytically the distribution of a statistic, but here we can rely on the approximate normality of the sample mean. If we further assume that spike counts are Poisson, so that their mean and variance are equal, then $T = \overline{Y}$ has mean μ_0 and variance μ_0/n under H_0. Figure 18.1 shows the approximate $N(\mu_0, \mu_0/n)$ null distribution of T. We see that when H_0 is true, common values for T are around 10, while values less than 8 or greater than 12 are extremely unlikely. The observed value $t_{\text{obs}} = 10.92$ is not very likely under that distribution, since it lies in its right tail; this provides some evidence against H_0. Values larger than t_{obs} would provide stronger evidence against H_0. This statement guides how we calculate the p-value:

$$p = Pr(T \geq t_{\text{obs}} \mid H_0 \text{ true}), \tag{18.1}$$

where "$\mid$ H_0 true" means that (18.1) is calculated with respect to the null distribution of T. We obtain $p = 7.3\%$. This means that if H_0 is true, the probability of observing a value of T larger than $t_{\text{obs}} = 10.92$ is 7.3%. A high probability means that t_{obs} is a likely value when H_0 is true. A low probability indicates that t_{obs} is an unusual value for T, which means that either we observed a rare event (a very unusual value of T), or H_0 is not true. Therefore, a small p-value provides strong, although not foolproof, evidence against H_0; rejecting H_0 could be the wrong decision. The significance level α is the chance we are willing to take of rejecting the null hypothesis by mistake. Here, $p = 7.3\%$ is greater than $\alpha = 5\%$: we fail to reject H_0 and conclude that the stimulus does not significantly increase the firing rate of the neuron. Note that this could be the wrong decision if H_A is in fact true (a type 2 error). The testing procedure does not control the probability of making a type 2 error. Only the probability of a type 1 error is bounded by α. For this reason, a large p-value does not provide evidence to endorse H_0, since the probability of committing a type 2 error could be large.

Table 18.1 Spike counts in $n = 25$ independent and identical trials of a simulated experiment. Their sample mean is $\overline{y} = 10.92$. We want to know if their true mean is 10 Hz or more

Spike counts: 9 10 10 13 9 9 15 11 11 10 16 12 11 12 8 10 16 13 11 10 13 7 8 10 9

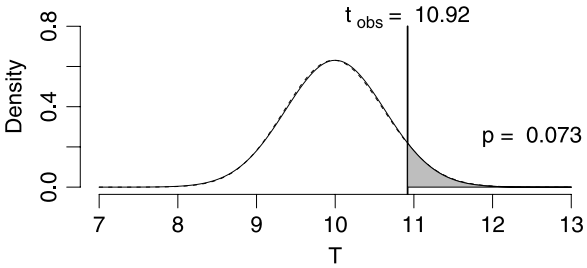

Fig. 18.1 Approximate $N(\mu_0, \mu_0/n)$ null distribution of the sample mean $T = \overline{Y}$, with $\mu_0 = 10$ and $n = 25$. The approximation relies on the central limit theorem and assumes that spike counts are Poisson

Critical value: it is the value c that t_{obs} would need to take to achieve $p = \alpha$. In our example, c is such that $\alpha = Pr(T \geq c \mid H_0 \text{ true})$, obtained by replacing p and t_{obs} by α and c in (18.1). A normal calculation yields $c \approx \mu_0 + 1.64\sqrt{\mu_0/n} = 11.04$ with $\alpha = 0.05$ and $\mu_0 = 10$. The relationship between α and c is such that we reject H_0 if $p \leq \alpha$ or equivalently if $t_{\text{obs}} \geq c$.

Two-sided null hypothesis: consider testing H_0: $\mu = \mu_0$ versus H_A: $\mu \neq \mu_0$. The sample mean remains an appropriate test statistic since it carries information about the true mean μ. However, values of T that are either larger *or smaller* than μ_0 now provide evidence against H_0. Hence the p-value is $p = Pr(T \geq t_{\text{obs}} \text{ or } T \leq t_s \mid H_0 \text{ true})$, where t_s is as small compared to $\mu_0 = 10$ as t_{obs} is large. In our example, the null distribution of T is symmetrical about μ_0, so $t_s = \mu_0 - (t_{\text{obs}} - \mu_0) = 9.08$, and $p = 14.6\%$. The two-sided test has a lower and an upper critical value, c_{low} and c_{up}, that satisfy $\alpha = Pr(T \geq c_{\text{up}} \text{ or } T \leq c_{\text{low}} \mid H_0 \text{ true})$ and that can be uniquely determined by solving $\alpha/2 = Pr(T \geq c_{\text{up}} \mid H_0 \text{ true}) = Pr(T \leq c_{\text{low}} \mid H_0 \text{ true})$. A normal calculation yields $c_{\text{low}} \approx \mu_0 - 2\sqrt{\mu_0/n}$ and $c_{\text{up}} \approx \mu_0 + 2\sqrt{\mu_0/n}$, with $\alpha = .05$. The relationship between α and the critical values is such that we reject H_0 if $p \leq \alpha$ or if $t_{\text{obs}} \geq c_{\text{up}}$ or $t_{\text{obs}} \leq c_{\text{low}}$.

18.2 Bootstrap Tests of Hypotheses

A Bootstrap test is a test of hypotheses for which the null distribution of T is obtained by a bootstrap simulation. The rest of the testing procedure outlined in the previous section remains unchanged.

Recall that the null distribution of a statistic T shows the values that T can take when H_0 is true. In the previous example, T is the sample mean of $n = 25$ Poisson spike counts. Thus, when H_0: $\mu = \mu_0 = 10$ Hz is true, the values that T can take are the means of $n = 25$ independent Poisson(10) random variables. Table 18.2 shows two such samples and their means, which tells us that possible values for T under H_0 are 8.44 and 9.84. Figure 18.2A displays the histogram of $R = 1{,}000$ such values of t^*, calculated from $R = 1{,}000$ samples of $n = 25$ Poisson(10) counts. This histogram is composed of 1,000 values that T can take when H_0 is true; it is therefore the approximate[2] null distribution of T under the assumption that spike counts

[2]The *true* null distribution is obtained as $R \to \infty$.

Table 18.2 Two bootstrap samples of size $n = 25$ simulated from Poisson(10), from which we calculate their sample means t^*

sample 1:	11 14 6 8 7 9 9 9 7 7 7 6 5 9 8 14 12 11 7 6 8 5 7 9 10	$t_1^* = 8.44$
sample 2:	11 14 8 4 8 10 7 13 9 9 16 8 11 17 12 6 4 12 12 10 8 8 8 9 12	$t_2^* = 9.84$

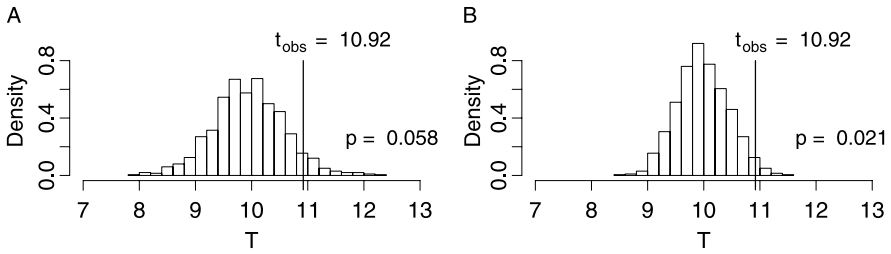

Fig. 18.2 (**A**) Parametric and (**B**) nonparametric bootstrap null distribution of $T = \overline{Y}$; t_{obs} is the value of T in the observed sample

are Poisson. Because it is obtained via simulation from a *parametric* model rather than via a theoretical calculation or an approximation, this histogram is referred to as the *parametric* bootstrap null distribution of T.[3] Note that the null distributions in Figs. 18.1 and 18.2A are almost identical. This is not surprising: both were obtained assuming that spike counts are Poisson(10) under H_0, and the normal approximation for the sample mean used to obtain Fig. 18.1 is typically accurate even for small sample sizes.

Parametric methods make the most efficient use of data, provided that model assumptions are correct. But spike counts are known to exhibit non-Poisson behavior, so it may be desirable to avoid the assumption of a specific parametric family of distributions. For this purpose, we assume instead that if H_0 was true, the distribution of the spike counts would be the same as that of the observed spike counts, but shifted to have mean $\mu_0 = 10$, to satisfy H_0: $\mu = \mu_0$. This may or may not be a reasonable assumption. In particular, it implies that the variance of the spike counts does not vary with their mean, which contrasts with the implication of equal mean and variance under the Poisson assumption. Next we estimate the distribution of the spike counts under H_0 with the *shifted* counts $x_1 = y_1 - (\overline{y} - \mu_0), x_2 = y_2 - (\overline{y} - \mu_0), \ldots, x_n = y_n - (\overline{y} - \mu_0)$, which have the same empirical distribution as the y_i's, except that their mean is $\mu_0 = 10$. Then to create one *nonparametric* bootstrap sample, we sample *at random and with replacement* $n = 25$ values from the set $\{x_1, \ldots, x_n\}$. We calculate t^*, the mean of that sample, and repeat R times. Figure 18.2B shows the histogram of $R = 1{,}000$ such values of t^*: this is the approximate *nonparametric* bootstrap null distribution of $T = \overline{Y}$ under the assumptions stated above. It does not match closely the null dis-

[3] In this particular application, it is also the Monte Carlo null distribution because H_0 fully specifies the data distribution under the Poisson assumption.

tributions in Figs. 18.1 and 18.2A, which is not surprising since it relies on different assumptions.

We have just described how to obtain parametric and nonparametric bootstrap null distributions for T in our simple example. The same process applies generally:

Bootstrap null distribution of T

For $r = 1, \ldots, R$, with R large:

1. Create the rth bootstrap sample—a bootstrap sample has the same size as the observed sample, and the same distribution the data would have if H_0 was true. That distribution is typically unknown, and the greatest challenge in conducting bootstrap tests is to design appropriate estimates for it.
2. Calculate t_r^*, the value of the test statistic T in bootstrap sample r.

The histogram of the R values of t_r^* approximates the null distribution of T.

To complete the test of H_0: $\mu = 10$ versus H_A: $\mu > 10$, we calculate the p-value in (18.1). Because the bootstrap null distribution of T is a histogram, (18.1) reduces to the proportion of samples for which $t^* \geq t_{\text{obs}}$,

$$p_{\text{boot}} = \frac{\text{number of bootstrap samples such that } \{t_r^* \geq t_{\text{obs}}\}}{R}, \tag{18.2}$$

which is the histogram area to the right of t_{obs} in Fig. 18.2AB. The parametric and nonparametric bootstrap p-values are thus $p_{\text{boot}} = 5.8\%$ and $p_{\text{boot}} = 2\%$, respectively: they lead to opposite decisions about H_0 at the $\alpha = 5\%$ significance level. Which p-value we should trust depends on whether or not spike counts are Poisson. See Sects. 18.3 and 18.4.

It is sometimes useful (for example, see Sects. 18.6 and 18.7) to calculate the critical value, that is, the value c that t_{obs} would need to take to achieve $p_{\text{boot}} = \alpha$. When the null distribution is a histogram, c is such that $\alpha\%$ of the t^*'s are greater than c. Hence, c is the $(R\alpha)$th ordered value of the t^*'s.

Simulation-based calculation of p-values is an important technique in situations where it is difficult to obtain an exact or approximate distribution for a test statistic T, or when such an approximation exists but is of dubious validity, either because the conditions it requires are not met, or because it relies on questionable assumptions about the distribution of the data. The test statistic in Sect. 18.1 was a sample mean, whose distribution is approximately normal (this result is known as the central limit theorem). This approximation is often accurate even for small sample sizes, so we could have foregone the parametric bootstrap tests of Sect. 18.2. The approximation also relied on the assumption that spike counts are Poisson under H_0. If this assumption is questionable, a different approximation, if available, or a nonparametric bootstrap test are recommended. But even in applications

where we are fairly confident in a particular parametric model and the statistical analysis based on that model, it can still be helpful, in the spirit of robustness, to see what can be inferred from the data without particular parametric model assumptions. A substantial literature has demonstrated both theoretically and in numerical studies that the bootstrap is widely effective (Davison and Hinkley 1997; Efron and Tibshirani 1993). But the simplicity of the bootstrap conceals an important point: its properties depend on how resampling is done; arbitrary shuffles of the data do not necessarily accomplish desired statistical goals. Moreover, in the context of hypotheses testing, the p-value must be obtained under the hypothetical reality imposed by H_0. In the example above, we needed to compute $P(\overline{Y} \geq 10.52)$ under the supposition that the observations $y_1, \ldots, y_n$ have mean $\mu_0 = 10$. We cannot simply resample the y_i's because their mean μ may not equal μ_0 (in fact, this is precisely what we want to test). Instead, we based the p-value calculation on data that resembled the y_i's except that they had mean μ_0, so that they would satisfy H_0.

18.3 Goodness of Fit Test

The test in Sect. 18.1 and the parametric bootstrap test in Sect. 18.2 relied on the assumption that the spike counts in Table 18.1 are Poisson. We now test this assumption. The null and alternative hypotheses are H_0: *the spike counts are Poisson* and H_A: *they are not Poisson*. The hypotheses are not expressed in terms of parameters, so choosing a test statistic T is not immediately obvious. In such cases, the alternative hypothesis should guide the choice of T, since we want to detect deviations from H_0 toward H_A. Poisson data have equal mean and variance, so deviations from equality is evidence of non-Poisson behavior. This suggests that we take as a test statistic the dispersion index (Fano factor) $T = V/\overline{Y}$, where $\overline{Y}$ and V are the sample mean and variance of the spike counts. Values of T that are *very different* from one give evidence in favor of H_A. The observed value of T for the data in Table 18.1 is $t_{\text{obs}} = 0.52$. An exact or approximate null distribution for T is not available, so we bootstrap. The null hypothesis specifies that spike counts are Poisson, but unlike the parametric bootstrap test in Sect. 18.2, H_0 does not specify the mean of the Poisson distribution. One solution is to use the sample mean $\overline{y} = 10.52$. We thus simulated $R = 1{,}000$ samples of size $n = 25$ from a Poisson(10.52) distribution and calculated the value of T for each of these samples. Figure 18.3A shows their histogram. Values of T smaller and larger than one provide evidence against H_0, and t_{obs} is in the left tail of the distribution, so the p-value is twice the probability of observing T smaller than t_{obs}, that is, $p_{\text{boot}} = 2 \times 2.3\% = 4.6\%$. This corresponds to $T \leq 0.52$ or $T \geq 1.66$. There is weak evidence in the sample to suggest that the counts are not Poisson. But see the next section for an amended conclusion.

If we wanted to test if the spike counts were consistent with some distribution G other than Poisson, we would need to design a test statistic whose deviation from a specific value provided evidence against G, and simulate bootstrap spike counts from G, the distribution of the data under H_0. If we wanted to test the Poisson

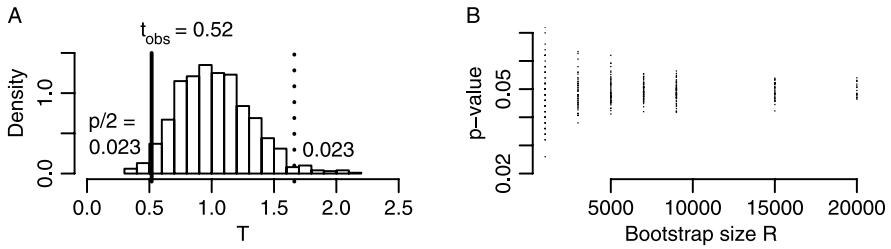

Fig. 18.3 (**A**) Bootstrap null distribution of the dispersion index $T = V/\overline{Y}$ under the null hypothesis that the spike counts are Poisson. (**B**) Bootstrap p-value as function of the bootstrap simulation size R

assumption against a more specific alternative than "not Poisson," say G, the *likelihood ratio* test statistic typically yields a powerful test, that is a test with a small probability of a type 2 error for a fixed significance level (Kass et al. 2005). An approximate χ^2 null distribution is sometimes available for the likelihood ratio, or it can be obtained from bootstrap samples simulated under the null hypothesis.

18.4 Simulation Error and Statistical Error

Bootstrap p-values such as (18.2) are *sample proportions*, so they vary across bootstrap simulations and their accuracies increase with R. This is illustrated in Fig. 18.3B, which shows the bootstrap p-values obtained from repeating the Poisson test of Sect. 18.3 for several values of R. We see that the p-values concentrate around 5% as we increase R, but with a simulation of size $R = 1{,}000$, they could vary between 2.5% and 6.5% and beyond, and thus lead to the rejection or retention of H_0 at random. The solution to eliminate simulation error is an infinite bootstrap simulation, which is impossible. We therefore recommend using $R = 1{,}000$. Then if $p_{\text{boot}} \gg \alpha$ or $p_{\text{boot}} \ll \alpha$, it is clear that there is, respectively, no evidence and compelling evidence in the data to support H_A. If $p_{\text{boot}} \approx \alpha$, either increase R, or mention that p_{boot} is variable and mitigate your conclusions. See also Davison and Hinkley (1997) for a more complete discussion of simulation size. In Sect. 18.3, the bootstrap Poisson test had p-value $p_{\text{boot}} = 4.6\%$ with $R = 1{,}000$, and although $p_{\text{boot}} < 5\%$, we cannot reject H_0 with the guarantee that the probability of committing a type 1 error is below $\alpha = 5\%$, since the true bootstrap p-value could deviate from 4.6% by several percentage points. In Sect. 18.2, the p-values of the parametric and nonparametric bootstrap tests were slightly above and below 5% respectively, but with $R = 1{,}000$, these two p-values may not actually be statistically different. Therefore we conclude that based on parametric and nonparametric bootstrap tests, the data provide weak evidence against H_0, with p-values close to 5%.

Statistical error can arise from using inappropriate data models. The optimal situation involves a null hypothesis H_0 that fully specifies the distribution of the data under H_0, for example, Poisson(μ_0) with $\mu_0 = 10$ in Sect. 18.2. But more often,

some aspects of the data distribution remain unknown under H_0. In Sect. 18.3, H_0 dictated that we simulate bootstrap samples from a Poisson distribution with unspecified mean; we used the sample mean of the spike counts as an estimate. In nonparametric problems, H_0 typically specifies very little about the data distribution. For the nonparametric bootstrap of Sect. 18.2, all we knew about the data under H_0 was their mean $\mu_0 = 10$ Hz. Therefore, to specify a sampling scheme, we further assumed that the distribution of the spike counts was the same as that of the observed data, but shifted to have mean $\mu_0 = 10$. We could instead have shifted the data to have mean 10 and further rescaled them to achieve a particular variance. In fact, there are many other distributions that satisfy H_0, and the bootstrap p-value will depend on which model is used to obtain bootstrap samples. One solution to avoid this problem is to design a test statistic whose null distribution does not depend, at least approximately, on unknown aspects of the data distribution. Another is to calculate the p-value conditional on sufficient statistics for the unknown parameters, since then the distribution of T no longer depends on those parameters. These solutions are seldom easy to apply, but they constitute good guiding principles.

To illustrate this, consider the test of H_0: $\mu = \mu_0$ versus H_A: $\mu > \mu_0$ in Sect. 18.1. We had assumed that spike counts were Poisson, so that the null distribution for the sample mean $T = \overline{Y}$ was $N(\mu_0, \mu_0/n)$. Without the Poisson assumption, the approximate normality of sample means still yields $T \sim N(\mu_0, \nu_T)$, where ν_T is the unknown variance of T. If we estimate ν_T by $V = S^2/n$, where S^2 is the sample variance of the spike counts, then $Z = (T - \mu_0)/\sqrt{V}$ has approximately a Student-t distribution. The t-distribution does not depend on the unknown ν_T; we say that Z is a *pivot*. This suggests a better nonparametric bootstrap test than that of Sect. 18.2: instead of $T = \overline{Y}$, we use $Z = (\overline{Y} - \mu_0)/\sqrt{V}$ as test statistic. As before, we simulate bootstrap samples $\{x_1^*, \ldots, x_n^*\}$ by sampling from the shifted spike counts $\{x_1, \ldots, x_n\}$, from which we calculate $z^* = (\overline{x}^* - \mu_0)/\sqrt{V^*}$, where V^* is the variance estimate of $\overline{x}^*$. The histogram of R such values of z^* approximates the distribution of Z and is independent of ν_T for large n. The nonparametric bootstrap p-value is the proportion of z^*s that exceed z_{obs}. Finding exact pivots relies on theoretical calculations, so they are generally hard to find, if they exist. *Studentized* bootstrap methods use statistics that are designed to be approximate pivots.

To illustrate the idea of conditioning on sufficient statistics, we consider the Poisson test in Sect. 18.3. Under H_0, the Poisson mean is unknown, so we estimated it with the sample mean; the bootstrap p-value depends on that estimate. But the sum of the spike counts $\sum_i Y_i$ is sufficient for the true mean, so that the bootstrap p-value will not depend on the unknown Poisson mean if it is calculated conditional on $\sum_i Y_i$. To do that, bootstrap samples $y_1^*, \ldots, y_n^*$ must be simulated from a Poisson distribution, as before, but they must satisfy $\sum_i y_i^* = \sum_i y_i$, the sum of the observed spike counts. It is often difficult or lengthy to obtain bootstrap samples that satisfy such conditions, but it is trivial in our example: a simple theoretical calculation implies that $y_1^*, \ldots, y_n^*$ can be simulated from a multinomial distribution with denominator $\sum_i y_i$ and n categories with equal probabilities n^{-1}. The *permutation* tests mentioned later are other examples of conditional tests.

Table 18.3 Spike counts of a neuron in a 400-ms window, in $n_p = n_s = 15$ repeats of two experimental conditions

Spike counts, pattern condition:	18 22 15 27 25 12 14 17 18 11 16 28 14 21 19
Spike counts, spatial condition:	18 10 11 17 17 13 16 24 15 15 18 18 14 17 9

18.5 Comparing Neuronal Responses

Many experiments involve comparing the responses of neurons to different stimuli. Consider the spike counts in Table 18.3, which come from a neuron in the supplementary eye field (SEF) of a macaque monkey (Olson et al. 2000). The monkey was shown visual cues to instruct him of the direction in which he was to look next. There were two cues: the "spatial" cue was cognitively simple, while the "pattern" cue was cognitively demanding. For a given eye movement, any significant difference between the neural responses in the two conditions can be attributed to cognitive rather than to motor functions.

Letting μ_s and μ_p denote the true mean spike counts in the two conditions, we test H_0: $\mu_p - \mu_s = 0$ versus H_A: $\mu_p - \mu_s \neq 0$. A natural test statistic is an estimator of $\mu_p - \mu_s$, for example, $T = \overline{Y}_p - \overline{Y}_s$, the difference in the sample mean firing rates in the two conditions. Its value for the data in Table 18.3 is $t_{\text{obs}} = 3$. Values of T larger or smaller than $\mu_p - \mu_s = 0$ provide evidence against H_0, so the p-value is $p = P(T \geq 3 \text{ or } T \leq -3 \mid \mu_p = \mu_s) = 2 \times 4.2\% = 8.4\%$, using a standard approximation[4]; see Fig. 18.4A. We fail to reject H_0 at significance level 5% and conclude that the data do not support the hypothesis that the SEF participates in cognitive functions. Note that Olson et al. (2000) reach the opposite conclusion based on a more carefully designed and more powerful test.

In this example, an approximate null distribution is available, and there is no reason to suspect its validity. However, for the sake of illustration, we suggest several bootstrap procedures that depend on various assumptions. We could first test if the spike counts in Table 18.3 were Poisson and if so, simulate spike counts in the two conditions from the same Poisson(μ) distribution; H_0 does not specify μ, so we replace it with the sample mean of the spike counts combined across conditions since under H_0, spike counts have the same true mean $\mu = \mu_p = \mu_s$ in the two conditions. Better yet, as suggested in Sect. 18.4, we can use the multinomial distribution Poisson spike counts conditional on the sum of the spike counts combined across conditions, which is sufficient for the common mean μ under H_0.

If spike counts are not Poisson and if we cannot identify an alternative model, we proceed with a nonparametric bootstrap. A simple option is to shift the spike counts in the spatial condition by some amount and the spike counts in the pattern condition by some different amount, so that the two shifted samples have the same sample means, to satisfy H_0. Then one bootstrap sample consists of n_s counts sampled from the spatial condition shifted spike counts and n_p counts sampled from the

[4] $(T - (\mu_p - \mu_s))/\sqrt{V_p + V_s}$ is approximately Student-t with $\frac{(V_p+V_s)^2}{V_p^2/(n_p-1)+V_s^2/(n_s-1)}$ degrees of freedom, where V_p and V_s are the usual variance estimates of $\overline{Y}_p$ and $\overline{Y}_s$.

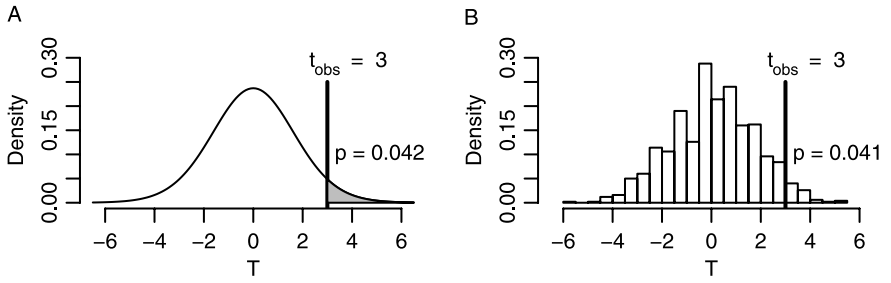

Fig. 18.4 (**A**) Approximate and (**B**) permutation like nonparametric bootstrap null distributions for $T = \overline{Y}_p - \overline{Y}_s$ for the equality of population means test

pattern condition shifted spike counts, from which the value of the test statistic T is calculated. Although we avoided explicit model assumptions, we assumed implicitly that if H_0 was true, the distributions of the spike counts in the two conditions would be the same as the distributions of the observed counts, except for a shift in their means. This assumption does not constrain the variance of the spike counts to be equal in the two conditions, which is odd given that H_0 specifies that the neuron has the same firing rate in the two conditions. It might therefore be more sensible to extend H_0: $\mu_s = \mu_p$ to mean H_0: *the spike counts have the same distribution in the two conditions*. However, it then becomes more tricky to obtain bootstrap samples, since shifting the spike counts as suggested earlier does not satisfy the extended H_0. The shifted counts would also need to be rescaled in some way to achieve at least equality of variances in the two conditions, if not equality of distributions. Alternatively, we can apply the ideas in Sect. 18.4, and use $Z = T/\sqrt{V_p + V_s}$ as the test statistic in place of T, where V_p and V_s are the sample variances of $\bar{Y}_p$ and $\bar{Y}_s$. Because it is standardized, Z is more pivotal than T, so that its null distribution and therefore the resulting p-value depend less on unknown aspects of the data distribution.

An exact conditional test is also available under the extended H_0: *the spike counts have the same distribution in the two conditions*: combine the *unshifted* spike counts in the two conditions, sample at random and with replacement $n_s + n_p$ spike counts from the combined sample, assign the first n_s to the spatial task, and the other n_p to the pattern task, and calculate the value t^* of $T = \overline{Y}_p - \overline{Y}_s$ in that sample. The histogram of $R = 1{,}000$ such values of t^* is in Fig. 18.4B, which we use to obtain the p-value $p_{\text{boot}} = 2 \times 4.1 = 8.2\%$. This p-value does not depend on unknown quantities, because it was calculated conditionally on a sufficient statistic for the unknown distribution of the spike counts under H_0 (the *empirical distribution function*). If sampling is done at random and *without replacement*, this procedure is the classic two-sample *permutation test*. See Olson et al. (2000) for a similar bootstrap simulation but use a more powerful test statistic.

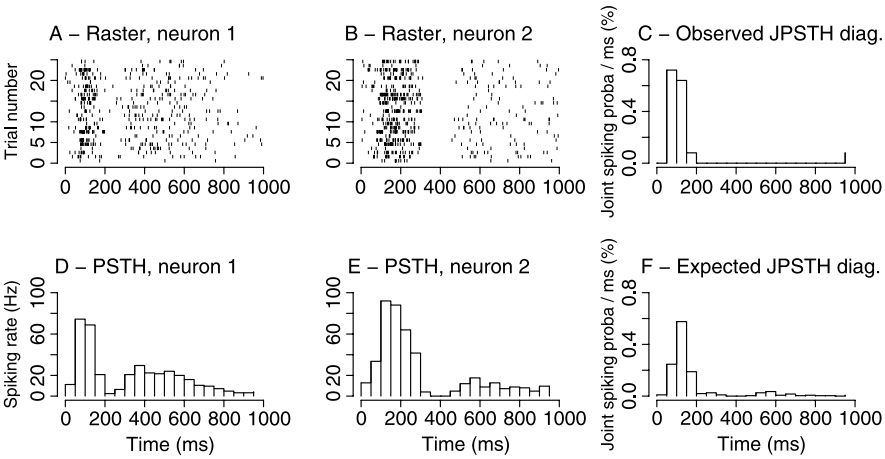

Fig. 18.5 Raster plots, PSTHs, main JPSTH diagonal, and product of the PSTHs, of the n joint spike trains of two neurons simulated jointly from a known model

18.6 Synchrony

We now examine the simultaneous firing of two neurons, with simultaneity defined as coincident spikes occurring within a specified window of time such as 1 ms (cf. Chaps. 10 and 17). The issue is whether neurons are independent. Letting $p_1(t)$ and $p_2(t)$ denote the probabilities of spiking at time t for neurons 1 and 2, and $p_{\text{joint}}(t)$ the probability of simultaneously spiking at t, independence implies $p_{\text{joint}}(t) = p_1(t)p_2(t)$. We therefore test H_0: $p_{\text{joint}}(t) = p_1(t)p_2(t)$ versus H_A: $p_{\text{joint}}(t) \neq p_1(t)p_2(t)$ and use an estimate of $p_{\text{joint}}(t) - p_1(t)p_2(t)$ as test statistic. Any apparent deviation of this estimate from 0 suggests extra correlation between the two neurons above the correlation induced by modulations in the firing rate.

An estimate of $p_1(t)$ is obtained as follows: discretize the spike trains of neuron 1: cut the time line of the experiment into, say, 1 msec bins, assign a value of 1 to the bins that contain a spike, and a value of 0 to the other bins; sum the n discretized spike trains bin by bin across trials and divide by n to obtain the 1 ms bins PSTH. Each bin contains the spiking proportion for the corresponding time: it is an estimate $\hat{p}_1(t)$ of the true spiking probability $p_1(t)$. Similarly, the 1 ms bin PSTH of neuron 2 estimates $p_2(t)$. An estimate of $p_{\text{joint}}(t)$ is obtained as follows: for a given trial, multiply, bin by bin, the discretized spike trains of the two neurons; the resulting *joint* spike train is a sequence of 0s and 1s, where a 1 is obtained only for the time bins that have a 1 in the spike trains of both neurons; repeat for all n pairs of spike trains and average the resulting n joint spike trains bin by bin (the result is the 1 ms bin main diagonal of the joint peri-stimulus time histogram or JPSTH; Aertsen et al. 1989). Figure 18.5 displays $\hat{p}_1(t)$, $\hat{p}_2(t)$, $\hat{p}_{\text{joint}}(t)$, and $\hat{p}_1(t)\hat{p}_2(t)$, averaged over 50 ms bins to improve clarity, for two neurons simulated from a known model. Figure 18.6 shows $t_{\text{obs}}(t)$, the observed value of the test statistic $T(t) = \hat{p}_{\text{joint}}(t) - \hat{p}_1(t)\hat{p}_2(t)$, which we smoothed to reduce its variability. The

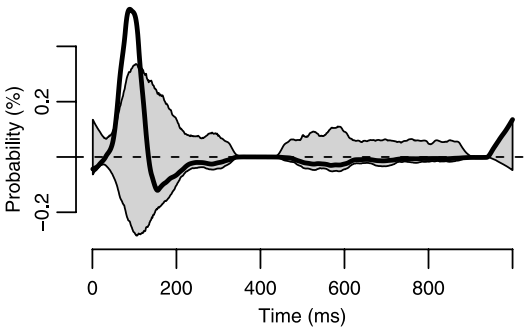

Fig. 18.6 Probable null envelope for the probability of joint spiking above chance, $T(t) = \hat{p}_{\text{joint}}(t) - \hat{p}_1(t)\hat{p}_2(t)$ with 95% pointwise coverage (*gray area*), and $t_{\text{obs}}(t)$ (*bold*), the value of $T(t)$ we observed. Excursions of $t_{\text{obs}}(t)$ outside the envelope suggest that the neurons are dependent for the corresponding times t

nominal shape of $T(t)$ under H_0 is a line with value zero for all t, so the large deviation of $t_{\text{obs}}(t)$ from zero in the [50, 150] ms range suggests that the two neurons may be dependent during that period. To be sure, we need the null distribution of $T(t)$ to assess the chance of such a deviation when the neurons are independent.

The following nonparametric bootstrap simulation is inspired by the permutation test for a correlation, which is a classic conditional test (Sect. 18.4): sample at random and with replacement the n spike trains of neuron 1, and independently, sample the n spike trains of neuron 2; put the two samples together to form n pairs of spike trains. This is a bootstrap sample. The spike trains have the same marginal distributions as the observed data, and they also satisfy H_0, since potential dependencies in pairs of spike trains were removed by sampling spike trains separately for the two neurons. Calculate $t^*(t)$, the value of $T(t)$ in that sample, following the instructions provided earlier to calculate $t_{\text{obs}}(t)$, and repeat R times. Note that if we had sampled the spike trains *without* replacement, the average of the $t^*(t)$ would be related to a (smoothed) *shuffle corrector* (Perkel et al. 1967).

Alternatively, we can perform a parametric bootstrap test: fit models to the spike trains of the two neurons, for example, Poisson or Gamma processes (see Chap. 16) or other models deemed more appropriate; sample n spike trains for neuron 1, and independently, sample n spike trains for neuron 2; put the two samples together to form n pairs of spike trains. This is a bootstrap sample. Calculate $t^*(t)$, the value of $T(t)$ in that sample, and repeat R times.

The R values of $t^*(t)$ cannot easily be displayed in a histogram to depict the null distribution of $T(t)$, since they are functions, not scalars. An alternative is to obtain a probable envelope for $T(t)$ under H_0: for each time t, define $T_{\text{low}}(t)$ and $T_{\text{up}}(t)$ to be the 2.5th and 97.5th quantiles of the R values $t^*(t)$, so that 95% of those values lie between $T_{\text{low}}(t)$ and $T_{\text{up}}(t)$ for any given t. The envelope obtained from the nonparametric bootstrap described above is plotted in Fig. 18.6. Basically, $T_{\text{low}}(t)$ and $T_{\text{up}}(t)$ are the critical values of the 5% level tests at each time t (see Sects. 18.1 and 18.2). Hence excursions of $t_{\text{obs}}(t)$ outside the envelope at t provide evidence against H_0 at t. For these times, we reject the null hypothesis of independence, with probability $\alpha = 5\%$ of making a type 1 error.

Note that the cross-correlogram (CC) (see Chaps. 5 and 6) is often used to test if neurons are dependent. Since our parametric or nonparametric bootstrap samples satisfy H_0, they can be used not only to obtain the null distribution of the JPSTH,

as we have done above, but also to obtain the null distribution of the CC or the null distribution of any test statistic we may fancy to test H_0 versus H_A.

Trial-to-trial variability effects: spike trains recorded from behaving animals display variation in spike timing that is sometimes consistent with the variation expected of Poisson or other point process models. In others, the irregularity is beyond that predicted by a common model for all trials. One aspect of extra trial-to-trial variation is the tendency for neurons to spike overall more or less for the duration of a trial, which is sometimes called spike count correlation. If such effects are shared across simultaneously recorded neurons, they will induce extra correlation in the neurons' spike trains and in turn inflate measures of dependence such as the JPSTH or the CC. This may cause one to wrongly claim that the two neurons are dependent, since apparent dependencies could be due to trial-to-trial variability effects. Figure 18.6 suggested that the neurons were dependent around $t = 100$ ms. But looking at Fig. 18.5 closely, we can see that some pairs of trials have many more spikes than others, which may be responsible for the significant result in Fig. 18.6.

Therefore we would like to assess if the neurons are correlated above what is expected from modulations in their firing rates and from trial-to-trial variability effects. We can still carry out a test based on significant excursions of $t_{\text{obs}}(t)$ outside a null envelope for $T(t) = \hat{p}_{\text{joint}}(t) - \hat{p}_1(t)\hat{p}_2(t)$, but that envelope must be obtained from bootstrap spike trains that have the same firing rates and the same trial-to-trial variability effects as the observed data, but that do not contain the short-term synchrony effects we wish to detect. The bootstrap samples used to obtain the envelope in Fig. 18.6 are not appropriate, because they do have the trial-to-trial variability effects observed in the data. The methods required to complete this test exceed the scope of this chapter. We refer to Ventura et al. (2005) for a parametric bootstrap solution and to Geman et al. (2009) for a nonparametric conditional test based on principles outlined in Sect. 18.3.

18.7 Joint Null Envelope for a Function

When we perform N *independent* tests simultaneously, we should expect to reject $\alpha\%$ of them on average when H_0 is true. Indeed, it is precisely the meaning of α. An equivalent notion is that, although a $(1-\alpha)$-coverage envelope has probability $1-\alpha$ of containing each test statistic, the chance that it captures all N of them is $(1-\alpha)^N$, which is much less than $1-\alpha$ for large N. In Fig. 18.6, we rejected 124 out of the 1,000 tests at each of the time points in [1, 1000] msec, which might seem sufficiently larger than $1{,}000\alpha = 50$ to suggest that some of these rejections are real. However that comparison is irrelevant because our tests are not independent. Indeed neurons may be similarly correlated in contiguous time bins, and we induced further correlation by smoothing $T(t) = \hat{p}_{\text{joint}}(t) - \hat{p}_1(t)\hat{p}_2(t)$.

A solution to determine the significance of the excursion of $t_{\text{obs}}(t)$ outside the envelope is to obtain a *joint* null envelope which has some suitably high probability,

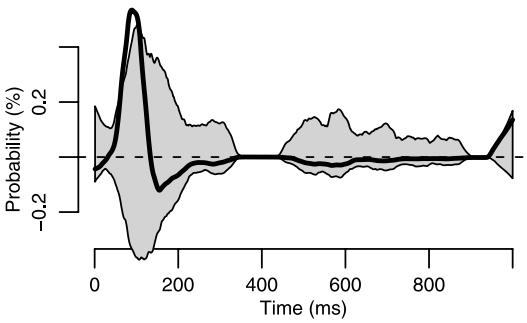

Fig. 18.7 Null envelope with joint coverage 89% for $T(t) = \hat{p}_{\text{joint}}(t) - \hat{p}_1(t)\hat{p}_2(t)$. $t_{\text{obs}}(t)$ exceeds the envelope around $t = 100$ ms, which suggests that the neurons exhibit above chance correlated activity around that time

β say, to capture an entire function $T(t)$ when H_0 is true. Then if $t_{\text{obs}}(t)$ exceeds the joint envelope, we reject the null hypothesis of independence with probability $1 - \beta$ of being wrong. A joint envelope is difficult to obtain exactly, but the previous bootstrap simulation can be used to find an approximation, as described in Davison and Hinkley (1997): given the R bootstrap functions $t^*(t)$, we set one of them aside, $t_1^*(t)$ say, calculate the $(1 - \alpha)$ pointwise envelope from the $(R - 1)$ remaining $t^*(t)$, and record if it fully covers $t_1^*(t)$. We repeat, setting aside each $t^*(t)$ in turn, and tally the proportion of $t^*(t)$'s that were fully included in their respective envelopes. This estimates β, the chance of the envelope fully covering $T(t)$ when H_0 is true. Following this procedure, we calculated that the 95% pointwise envelope in Fig. 18.6 has joint coverage $\beta = 23\%$. Hence, the chance that $t_{\text{obs}}(t)$ exits the envelope when the neurons are independent is 77%, which casts serious doubts on our earlier conclusion that the neurons are dependent. We can also calibrate α to achieve a desired β. For example, $\alpha = 1\%$ and 0.5% yield $\beta = 71\%$ and 89%, respectively. The joint 89% coverage null envelope is plotted in Fig. 18.7. It does not fully contain $t_{\text{obs}}(t)$, so we conclude with some degree of confidence that the neurons are indeed correlated above chance around $t = 100$ ms.

Note that to achieve a high β requires that α be very small. This means that the joint envelope is made of extreme quantiles in the tails of the bootstrap null distribution of $T(t)$, which are highly variable (this is related to simulation error in Sect. 18.4). Therefore we do not recommend obtaining a very high joint coverage envelope, since it would be too variable to be reliable, unless a very large simulation size R is used.

References

Aertsen A, Gerstein G, Habib M, Palm G (1989) Dynamics of neuronal firing correlation–modulation in effective connectivity. J Neurophysiol 61:900–917

Davison AC, Hinkley DV (1997) Bootstrap methods and their applications. Cambridge University Press, Cambridge

Efron B, Tibshirani R (1993) An introduction to the bootstrap. Chapman and Hall, New York

Geman S, Amarasingham A, Harrison M, Hatsopoulos N (2009) The statistical analysis of temporal resolution in the nervous system. Division of Applied Mathematics Technical Report, Brown University

Kass RE, Ventura V, Brown EN (2005) Statistical issues in the analysis of neuronal data. J Neurophysiol 94:8–25

Olson CR, Gettner SN, Ventura V, Carta R, Kass RE (2000) Neuronal activity in macaque supplementary eye field during planning of saccades in response to pattern and spatial cues. J Neurophysiol 84:1369–1384

Perkel DH, Gerstein GL, Moore GP (1967) Neuronal spike trains and stochastic point processes. II. Simultaneous spike trains. Biophys J 7:419–440

Ventura V, Cai C, Kass RE (2005) Trial-to-trial variability and its effect on time-varying dependence between two neurons. J Neurophysiol 94:2928–2939

Chapter 19
Generating Random Numbers

Hans Ekkehard Plesser

Abstract Large-scale parallel surrogate data generation and neuronal network simulation require far more pseudorandom numbers than many popular pseudorandom number generators (PRNGs) can deliver. Fortunately, random number generation has progressed from black art to science over the past two decades, providing us with reliable algorithms for generating random numbers and rigorous tests for such algorithms. In this chapter we will first review requirements for good PRNGs, before presenting the basic principles of some widely used generators, including cryptographic generators. We then discuss seeding strategies and the transformation of integer random numbers to random variates following other distributions.

Random number generation has progressed from black art to science over the past two decades (Panneton et al. 2006; Park and Miller 1988), providing us with reliable algorithms for generating random numbers and rigorous tests for such algorithms (L'Ecuyer and Simard 2007). No deterministic algorithm can generate truly random numbers. Strictly speaking, we are considering *pseudorandom number generators* (PRNGs). For any generator, there will be some application that reverberates with the correlation structure hidden in the generator's algorithm, leading to false results (Ferrenberg et al. 1992; Compagner 1995).

Before considering pseudorandom number generators in detail, let us briefly consider the number of random numbers required for a large project that keeps a 1024-core computer busy for a year, or roughly 2^{55} nanoseconds. Assuming a clock speed of 1 GHz, the project goes through $2^{65} \approx 4 \times 10^{19}$ clock cycles. If one pseudorandom number is consumed per 10^4 clock cycles, a total of 4×10^{15} numbers will be consumed (Knuth 1998, Chap. 3.6). Empirical evidence indicates

H.E. Plesser (✉)
Dept. of Mathematical Sciences and Technology, Norwegian University of Life Sciences, PO Box 5003, 1432 Aas, Norway
RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan
e-mail: hans.ekkehard.plesser@umb.no
url: http://arken.umb.no/~plesser

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_19, © Springer Science+Business Media, LLC 2010

that a PRNG used in this project should be able to generate a sequence of at least $(4 \times 10^{15})^3 = 64 \times 10^{45}$ pseudorandom numbers before repeating itself (Brent 2006; Matsumoto et al. 2006; L'Ecuyer and Simard 2007). This exceeds by far the capabilities of many widely used generators.

In this chapter we will first review requirements for good PRNGs, before presenting the basic principles of some widely used generators, including cryptographic generators. We then discuss seeding strategies and the transformation of integer random numbers to random variates following other distributions.

For details, please see the textbook by Gentle (2003), Knuth's classical treatise (1998), or the review by L'Ecuyer and Simard (2007); the latter two references include a comprehensive treatment of tests for PRNGs. Devroye (1986) is the canonical reference for nonuniform random variates.

19.1 Requirements for Pseudorandom Number Generators

A pseudorandom number generator is an algorithm providing a *drawing function* $\mathcal{D}()$ which emits a new pseudorandom number $r_j \in \mathbb{W}$ each time the drawing function is executed; $\mathbb{W}$ is the finite, discrete set from which numbers are chosen. The *period* ρ of the generator is the number of PRNs that can be drawn before the sequence of numbers repeats itself. If different initializations of the generator can lead to different periods, one considers the minimal period as the period of the generator. For most modern generators, the period is much larger than the set of values $\mathbb{W}$.

Brent (2006) lists the following requirements for good PRNGs:

Uniformity The numbers $(r_0, r_1, \ldots)$ generated should be distributed uniformly over $\mathbb{W}$.

Independence Any subsequence of the full sequence $(r_0, r_1, \ldots)$ of random numbers should be statistically independent.

Long period From a PRNG with period ρ, one should draw at most $\rho^{1/3}$ random numbers (*cube-root rule*; Brent 2006; Matsumoto et al. 2006; L'Ecuyer and Simard 2007).

Initialization Any choice of seed should yield equally random numbers.

Skip ahead Certain random generators allow one to "skip ahead", e.g., to go directly to the 2^{100}th random number in a sequence (L'Ecuyer et al. 2002; Haramoto et al. 2008).

Repeatability Especially when debugging simulations, it is crucial that the same random number sequence can be generated over and over again.

Portability A PRNG should produce exactly the same sequence of numbers for any computer on which it is run, without modifications to the source code.

Efficiency Random number generators should consume as little as possible of the total simulation time.

Unpredictability A truly random number sequence is unpredictable: no matter how long we observe the sequence, we will never be able to predict the next number. This is not the case for most PRNGs.

Uniformity and independence are ascertained by standard tests for pseudorandom number generators, while the period is usually known from the design of the generator. Initialization has received systematic attention only recently (Matsumoto et al. 2007); we will return to it in Sect. 19.4.

The ability to skip ahead in a random number stream is essential for the parallel generation of random number sequences and thus useful for generating large numbers of surrogate spike trains. It will be discussed in detail in Sect. 19.4.1.

Since pseudorandom number generators are based on deterministic algorithms, it is straightforward to reproduce their output in repeated simulations, provided that their initial state is known. Ill-designed parallel programs may still lead to irreproducible consumption of random numbers; we will not discuss such issues here.

Portability and efficiency of PRNGs are of lesser concern today than a decade ago, since several well-tested, portable PRNG libraries exist. An interesting new development are PRNGs exploiting the superior performance of vector extensions to CPUs and of graphics processors (Saito and Matsumoto 2008; Tzeng and Wei 2008).

Unpredictability of PRNGs is of little concern in simulations, while it is essential in cryptography and gaming. We will discuss unpredictable cryptographic generators in Sect. 19.3.

19.2 Recurrence-Based Generators

Most pseudorandom number generators used today are recurrence-based generators. These generators are defined by three mathematical functions operating upon a state variable Σ. The random generator is initialized by a *seed* value S according to a *seeding rule*

$$\Sigma_0 = \mathcal{S}(S). \tag{19.1}$$

Each time a new random number is required, the state of the generator is updated by an *iteration rule*

$$\Sigma_j = \mathcal{I}(\Sigma_{j-1}) \quad (j > 0), \tag{19.2}$$

before the jth random number is drawn from the state variable according to a *drawing rule*

$$r_j = \mathcal{D}(\Sigma_j). \tag{19.3}$$

The size of the state variable Σ sets an upper limit for the period of the generator. If Σ is n bits long, then the generator is at all times in one of 2^n states. Since (19.2) dictates that each state has exactly one successor state, it follows that the longest possible sequence of states without repetition has 2^n states, limiting the period to $\rho \leq 2^n$. PRNGs with 32-bit integers as state variables thus cannot have periods of more than $\rho = 2^{32} \approx 10^9$, which is far too short to be of practical use.

The random numbers r_j that can be drawn from a generator are typically 32-bit integers, although some generators provide floating point numbers directly. The

range of values that can be generated stretches from 0, 1, or 2 to some large integer $M < 2^n$. Note that the period ρ of a generator can be much larger than M.

19.2.1 Linear Congruential Generators

Additive linear congruential generators (LCGs) have been popular due to their simplicity and efficiency and are infamous for their weaknesses (Knuth 1998, Chap. 3.2.1). Seeding and drawing functions are trivial for these generators, while the iteration function is straightforward modulo arithmetic:

$$\mathcal{S}(s) = s \quad (0 < s < m), \tag{19.4}$$

$$\mathcal{D}(\Sigma) = \Sigma, \tag{19.5}$$

$$\mathcal{I}(\Sigma) = (a\Sigma + c) \bmod m, \tag{19.6}$$

where a is the *multiplier*, c the *increment*, and m the *modulus*. Lewis et al. (1969) proved that choosing $a = 7^5 = 16807$, $c = 0$, and $m = 2^{31} - 1 = 2{,}147{,}483{,}647$ results in a generator with maximum period, i.e., period $\rho = 2^{31} - 2$. Care must be taken to avoid integer overflow when implementing this algorithm using 32-bit integers (Park and Miller 1988).

Even though this generator has been considered a "minimal standard" for a long time (Park and Miller 1988; Press et al. 1992, Chap. 7), it is no longer a viable choice for large-scale simulations today due to its short period: strict application of the cube-root rule would allow the use of only 1290 random numbers, and even a common PC will zip through the entire period within a minute.

19.2.2 Lagged Fibonacci Generators

Lagged Fibonacci generators (LFG) were first proposed by Mitchell and Moore some 50 years ago and have long been considered reliable (Knuth 1998, Chap. 3.2.2). Their state vector $\mathbf{\Sigma}$ consists of the last l 32-bit random numbers drawn according to the generalized Fibonacci recurrence relation

$$r_n = (r_{n-s} \pm r_{n-l}) \bmod m \quad (0 < s < l \leq n). \tag{19.7}$$

Several valid pairs (l, s) of long and short lags are given by Knut (1998, Chap. 3.2.2, Table 1). Mitchell and Moore originally proposed lags (55, 24), while Knuth recommends (100, 37) in his Chap. 3.6. For moduli of the form $m = 2^k$, the LFG generator has a period of $\rho = 2^{k-1}(2^l - 1)$. For the parameters given by Knuth, one thus obtains a period $\rho \approx 2^{129}$.

Some weaknesses in these generators can be overcome by *decimation*, e.g., using only the first 100 of every 1009 numbers generated (Lüscher 1994). This gives the

canonical implementation of the LFG. Source Code is freely available from Donald Knuth's website.[1]

Properly initializing the state vector $\mathbf{\Sigma}$, which contains l 32-bit integers, based on a single 32-bit seed S is not trivial; Knuth has revised the initialization routine for the LFG as recently as in 2002.[2] The present initialization scheme appears robust under stringent tests (Matsumoto et al. 2007).

L'Ecuyer (2004) advises strongly against the use of LFGs, since the triplets (X_n, X_{n-s}, X_{n-l}) have strong correlations between themselves, so that simulators which should happen to combine random numbers in these intervals in unfortunate ways may suffer.

Marsaglia and Tsang (2004) suggest to combine a lagged Fibonacci generator with lags 97 and 33 with an arithmetic sequence of the form $c - jd \bmod p$ for suitable constants c, d, and p, to obtain a generator with period $\rho \approx 2^{202}$. This generator does not suffer from the shortcomings of "plain" LFGs, and thus one does not need to decimate the random number stream.

19.2.3 Combined Multiple Recursive Generators

Multiple recursive generators (MRG) generalize the lagged Fibonacci generators by introducing more terms and arbitrary coefficients into the recurrence (L'Ecuyer 2004)

$$X_n = (a_1 X_{n-1} + a_2 X_{n-2} + \cdots + a_k X_{n-k}) \bmod m. \tag{19.8}$$

LCGs are a special case of an MRG for $k = 1$, while LFGs are MRGs for which two $a_j = \pm 1$ while all other coefficients vanish. Combining two such generators with carefully chosen parameters results in a combined multiple recursive generator (CMRG) such as `MRG32k3a` with the following equations describing the recurrence and drawing (L'Ecuyer 1999):

$$\Sigma_{1,n} = (1{,}403{,}580 \times \Sigma_{1,n-2} - 810{,}728 \times \Sigma_{1,n-3}) \bmod m_1, \tag{19.9}$$

$$\Sigma_{2,n} = (527{,}612 \times \Sigma_{2,n-1} - 1{,}370{,}589 \times \Sigma_{2,n-3}) \bmod m_2, \tag{19.10}$$

$$r_n = (\Sigma_{1,n} - \Sigma_{2,n}) \bmod m_1, \tag{19.11}$$

$$m_1 = 2^{32} - 209 = 4{,}294{,}967{,}087, \tag{19.12}$$

$$m_2 = 2^{32} - 22{,}853 = 4{,}294{,}944{,}443, \tag{19.13}$$

where Σ_1 and Σ_2 are stored as doubles with 53-bit mantissa. This combined generator has a period $\rho = (m_1^3 - 1)(m_2^3 - 1)/2 \approx 2^{191} \approx 10^{57}$ and passes all tests in the TestU01 suite (L'Ecuyer and Simard 2007). An implementation in C is available from L'Ecuyer.[3] It is included in Matlab 7.7 and later. The `cmrg` generator in

[1] http://www-cs-faculty.stanford.edu/~knuth/programs/rng.c.

[2] As of the 9th printing of Knuth (1998).

[3] http://www.iro.umontreal.ca/~lecuyer/myftp/papers/combmrg2.c.

the GNU Scientific Library is a similar CMRG with a period $\rho \approx 2^{185}$, which also passes all TestU01 tests but shows some weakness in initialization tests (Galassi et al. 2001; Matsumoto et al. 2007). We will return to `MRG32k3a` in Sect. 19.4.1.

19.2.4 Mersenne Twister and Related Generators

Mersenne Twister (MT) random number generators were introduced by Matsumoto and Nishimura (1998) based on a class of generators known as *feedback shift register* generators. These generators do not operate on integer or double numbers, but on vectors of individual bits. This permits fast code based on bit-shift and bit-wise logical operations, thorough mathematical analysis, and efficient skipping ahead. Mersenne Twisters have thus become very popular in the ten years since their invention. For an introduction to the principles behind Mersenne Twisters and related generators, see L'Ecuyer (2004), on which the following is based.

For MT generators, all arithmetic is done modulo 2, i.e., in the finite field $\mathbb{F}_2$ with the two elements 0 and 1. The state variable $\mathbf{\Sigma}$ is a vector of k 0s and 1s, and iteration and drawing rules are given by

$$\mathbf{\Sigma}_j = \mathbf{A}\mathbf{\Sigma}_{j-1}, \tag{19.14}$$

$$\mathbf{r}_j = \mathbf{B}\mathbf{\Sigma}_j, \tag{19.15}$$

$$u_j = \sum_{l=1}^{w} r_{j,l} 2^{-l}, \tag{19.16}$$

with the $k \times k$ transition matrix $\mathbf{A}$ and the $w \times k$ output transformation matrix $\mathbf{B}$, both with elements from $\mathbb{F}_2$. $u_j \in [0, 1)$ is the output of the algorithm given as a double. Since doubles have only a 53-bit mantissa, one usually has $w \leq 53$, while $k \gg w$. The matrix multiplications in the equations above can be implemented efficiently through bit-shift and bit-wise logical operations.

The most commonly used Mersenne Twister today is MT19937 published in 1998, with an improved initialization routine published in 2002; source code is available from Matsumoto and others[4] and is available in the GNU Scientific Library as `mt19937`[5]; it is the default PRNG in NumPy and Matlab (since version 7.4). MT19937 has $k = 19{,}937$, i.e., the transition matrix $\mathbf{A}$ is a very sparse matrix of $19{,}937 \times 19{,}937$ bits. The state variable $\mathbf{\Sigma}$ correspondingly contains 19,937 bits, stored in 624 32-bit words. The period of the generator is $\rho = 2^{19{,}937} - 1 \approx 10^{6{,}000}$, so that even after application of the cube-root rule we are left with a mindboggling number of usable random numbers. MT19937 is fast in spite of its huge state vector and has passed all tests in the TestU01 testsuite except two, which all feedback shift register generators fail. On this failure, L'Ecuyer and Panneton (2005) comment that

[4] http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html.

[5] GSL v. 1.2 and later; earlier versions have the defective pre-2002 initialization routine.

"this is very unlikely to cause a problem in practice, unless the system we simulate has a lot to do with linear dependencies among bits". Doubts about the MT19937 recently raised by Kim et al. (2008) were based on a faulty analysis (Plesser and Jahnsen 2010).

19.2.5 Nonlinear Random Number Generators

All PRNGs presented so far are linear. Nonlinear generators, such as inversive congruential generators (ICG) proposed first by Eichenauer and Lehn (1986), are based on recurrences of the type

$$X_{j+1} = (a\overline{X}_j + b) \bmod m,$$

where the bar denotes the inverse with respect to multiplication modulo m, and m is a large prime. Integer division modulo a large prime is slow, whence ICGs are about 50 times slower than MT19937 (L'Ecuyer and Simard 2007). Due to their short period (typically $\rho \approx 2^{31}$), ICGs also fail the TestU01 tests. They may still be of interest in special situations, as they produce random numbers in a quite different fashion than all linear generators. If you doubt your simulation results, it may be an idea to redo a few simulations using an ICG or other nonlinear generator. The PRNG library by Otmar Lendl provides several such generators.[6]

19.3 Cryptographically Strong Random Number Generators

Cryptography, as well as gaming and gambling, have very different demands on random number generators than simulation-based science. For secure cryptography, it is crucial that a random number generator is *unpredictable*, i.e., that no efficient algorithm exists that permits an observer to predict future random numbers based on numbers produced by the generator in the past.

Reasonably fast cryptographic random number generators are based on encryption algorithms such as the Advanced Encryption Standard (AES) (Hellekalek and Wegenkittl 2003) and hash functions (Barker and Kelsey 2007). Encryption functions take a key K and a plaintext P and return a ciphertext

$$C = E(K, P). \tag{19.17}$$

They can be used as random number generators in one of three ways (Hellekalek and Wegenkittl 2003; Dworkin 2001):

- In *output feedback mode* (OFB) one chooses a fixed key K and some plaintext S to generate $C_0 = E(K, S)$. Random numbers are then generated by iterating the encryption on the ciphertext $C_j = E(K, C_{j-1})$.

[6] http://statistik.wu-wien.ac.at/software/prng.

- In *counter mode* (CTR) one chooses a fixed key K and obtains a sequence of ciphertexts by encrypting a counter that is increased by one for each number drawn, $C_j = E(K, S + j)$.
- *Key counter mode* (KTR) is complementary to counter mode: subsequent numbers are used as keys to encrypt a fixed plaintext, $C_j = E(S + j, P)$.

Random numbers are obtained from the ciphertext using a suitable drawing function $r_j = \mathcal{D}(C_j)$. The period of a generator based on (key) counter mode can obviously not be larger than the range of values the counter can assume. Therefore, long counters with hundreds of bits should be used. Several cryptographic RNGs have passed the TestU01 suite (L'Ecuyer and Simard 2007); the testsuite also contains implementations of random generators based on AES and SHA-1 encryption with periods of 2^{130} and 2^{440}, respectively.

Cryptographic RNGs in a (key) counter mode have no internal state variable: each random number is generated independently of all previous and subsequent numbers. This has two interesting advantages: First, skipping ahead is trivial. Second, many random numbers can be generated in parallel exploiting single-instruction, multiple-data (SIMD) coprocessors such as SSE and AltiVec extensions in modern CPUs or the massive power of current graphics cards (Tzeng and Wei 2008).

19.4 Seeding Random Number Generators

The choice of seeds for random number generators is subject of much folklore and little systematic investigation. Deficiencies in the seeding process of the Mersenne Twister and Knuth's LFG were discovered as late as 2002, and Matsumoto et al. (2007) recently reported that most PRNGs in the GNU Scientific Library have weak seeding strategies. Marsaglia and Tsang (2004) provide an interesting perspective on seeding of random number generators for legal or gambling purposes.

We assume here that surrogate data generation requires one to generate streams of random numbers to generate spike trains. All these streams must be statistically independent, no matter how they are generated and consumed: sequentially (serial algorithm runs many times in succession), simultaneously (many instances of a serial algorithm run independently on a cluster), or in parallel (parallel algorithm consuming several PRN streams).

Unfortunately no generator can provide *provably* statistically independent streams. The best a good PRNG can offer is to guarantee, or at least make it highly likely, that different seeds lead to nonoverlapping random number sequences of a certain length. A good PRNG should thus come with an explicit seeding algorithm which ensures reliable results for any admissible seed. Use it, instead of relying on traded advice on the choice of seeds such as choosing seeds with a roughly equal number of 0 and 1 bits.

The good news is that if the PRNG is well behaved for *any* choice of seed, then we can choose seeds systematically throughout our project. Assuming 32-bit num-

bers as seeds, we could, for example, compute the seed for any given stream according to

$$S_{\text{stream}} = n_{\text{coll}} \times 2^{26} + n_{\text{ex}} \times 2^{16} + n_{\text{tr}} \tag{19.18}$$

with collaborator number n_{coll}(1–63), experiment number n_{ex}(0–1,023), and trial number n_{tr}(0–65,535), specifying the stream. This scheme is easily extended to a wider range of components by using seeds with more than 32 bits. Such a scheme provides a systematic and clear way of enumerating random number streams, ensures that no two collaborators or simultaneous runs will use identical seeds, and leaves the difficult task of actually initializing the PRNG to the expert developers of generators.

That said, it is still useful to have a minimal understanding of how seeding actually works. For most PRNGs, the state vector $\mathbf{\Sigma}$ will take on any possible value at some point during the period ρ of the generator. Seeding simply initializes the state vector to some value, based on the value of the seed. Good seeding algorithms spread starting points evenly across the period ρ. When seeding the Mersenne Twister with $\rho = 2^{19,937} - 1$ with 32-bit integer seeds, starting points should be separated by roughly $2^{19,905}$ random numbers, provided that the seeding algorithm is good. Naive initialization of the state vector, on the other hand, may lead to initial states separated by only very few numbers and thus overlapping, highly correlated streams.

19.4.1 Parallel Streams of Random Numbers

We shall now briefly discuss the *block splitting* technique, also known as *cycle division* used to initialize PRNGs in a way that ensures nonoverlapping random number streams (L'Ecuyer et al. 2002; Bauke and Mertens 2007). *Parameterization* of PRNGs is another technique for the same purpose, see Mascagni and Srinivasan (2000, 2004).

Block splitting exploits that some PRNGs allow us to jump ahead by a certain number B of random numbers: given state $\mathbf{\Sigma}_0$, we can quickly find any $\mathbf{\Sigma}_{nB}$. Using n as a stream index, we can thus choose between ρ/B *provably* nonoverlapping random number streams of length B. The `MRG32k3a` generator (cf. Sect. 19.2.3) has a full period of $\rho \approx 2^{191}$, which can be split into 2^{64} streams of length $B = 2^{127}$ (L'Ecuyer et al. 2002). Similar techniques are available for lagged Fibonacci generators (Mascagni and Srinivasan 2004; Knuth 1998, Chap. 3.6, Ex. 3.6-9). Haramoto et al. (2008) developed a jump-ahead algorithm for the Mersenne Twister and related PRNGs, but no implementations appear to be publicly available at this time. Matlab v. 7.7 provides nonoverlapping streams of random numbers using block splitting for some generators; unfortunately, neither the Python `random.jumpahead()` (v. 2.6.2) nor the GSL `gsl_rng_set()` provide any similar guarantees.

Selecting random streams using block splitting does not mix well with changing the seed of the underlying PRNG, as both select starting points for random number

sequences from the same overall sequence. When using generators with multiple streams, you should choose one fixed seed for your entire study and then use (19.18) to select streams. For parallel simulations, simply extend the equation to include MPI rank or thread number.

19.5 Transforming Random Numbers

Most random number generators draw "raw" numbers r_j uniformly from the set $\mathbb{W} = \{0, 1, \ldots, M\}$ with $M \approx 2^{32}$. These raw numbers need to be transformed for use in simulations. We will consider here transformation to the unit interval, to a given range of integers, the exponential distribution, and some more general methods; for more information, see Gentle (2003), Knuth (1998), or Devroye (1986). If your software package or library provides methods to generate random deviates with the required distribution, you should use those methods.

Raw numbers r_j are usually converted to floating-point random numbers in the unit interval $[0, 1)$ using $u_j = r_j/(M + 1)$. While most scientific software today uses floating point variables with a 53-bit mantissa, yielding a machine resolution of $\mathcal{O}(10^{-16})$, 32-bit integer random numbers will yield u_j with a resolution of only $\mathcal{O}(10^{-9})$. This need not be a problem in practice, but you should be aware of this limitation. Random numbers with 53-bit mantissa can be obtained by suitable combination of two subsequent raw numbers.

The most common approach concerning the endpoints of the unit interval appears to be to draw from $[0, 1)$. Including 0 in the range of values can lead to problems in algorithms taking the logarithm of or dividing by random numbers. The most common strategies to exclude 0 are to redraw if 0 is drawn, or to substitute a very small value for 0. Doornik (2007) recently proposed an efficient way of obtaining floating point random numbers on the open interval $(0, 1)$ that are strictly symmetric about $\frac{1}{2}$. To our knowledge, his algorithm has not yet been incorporated in standard random number libraries.

Uniformly distributed integers i_j from a range $\{1, 2, \ldots, n\}$ are most easily computed as $i_j = 1 + \lfloor nu_j \rfloor$, although some libraries transform the r_j into i_j directly. You should not use $1 + (r_j \bmod n)$ to obtain the i_j, since this exploits only the lowest bits of the generated random numbers.

Random deviates following any other probability density $p(x)$ with pertaining cumulative distribution function $P(x) = \int_{-\infty}^{x} p(s)\,ds$ can be obtained in a number of ways. If $P(x)$ can be inverted analytically, then $x_j = P^{-1}(u_j)$ will be distributed according to $p(x)$. Exponentially distributed numbers with $p(x) = e^{-x/\mu}/\mu$ are thus obtained as $x_j = -\mu \ln(1 - u_j)$.

If $P^{-1}(x)$ is not available analytically or is computationally expensive, rejection methods are often useful. Let $q(x)$ be another probability density with an easily computable inverse $Q^{-1}(u)$, $p(x) \leq \alpha q(x)$ for all x, and $\alpha \geq 1$. Obtain a candidate $z = Q^{-1}(u_j)$; if $\alpha q(z)u_{j+1} < p(z)$, accept $x = z$, otherwise obtain a new candidate z. To be efficient, this method requires that the majorizing function $\alpha q(x)$ is a tight bound on the actual distribution $p(x)$. This can often be achieved by composing $q(x)$ from suitable functions for parts of the support of $p(x)$.

19.6 Recommendations

Donald Knuth (1998, Chap. 3.6) reminds us that

> "... the history of the subject warns us to be cautious. The most prudent policy ... to follow is to run each Monte Carlo program at least twice using quite different sources of random numbers, before taking the answers of the program seriously; this will not only give an indication of the stability of the results, it will also guard against the danger of trusting in a generator with hidden deficiencies. (Every random number generator will fail in at least one application.)"

Many a lesson has been learned the hard way about the interference between random number generators and Monte Carlo simulations in physics (Ferrenberg et al. 1992; Bauke and Mertens 2004). No comparable problems caused by random number generators have been reported in the computational neuroscience literature—yet. This does by no means mean that such problems do not exist: they might just have gone undetected so far, for want of analytical solutions to model equations providing gold standards. Spurious observations due to numerical inaccuracies in neuronal simulations should serve as a warning (Hansel et al. 1998).

With this in mind, we conclude with four recommendations:

1. Do not tie yourself to a single random number generator but implement a flexible interface to existing random number libraries.
2. Do not implement random number generators yourself but use existing, well-tested libraries such as the GNU Scientific Library.
3. Test your scheme for drawing random numbers against reference data provided by the RNG developers or against established test resources, such as L'Ecuyer's TestU01 testsuite, to make sure that random numbers are correctly handled by your interface to the random generator libraries. Include such tests in the test-suite for your software, so you can validate the interface when porting to a new architecture.
4. Consult *recent* tests of random number generators, at the time of writing the TestU01 test report (L'Ecuyer and Simard 2007).

And finally, before you submit your groundbreaking results for publication, repeat at least some simulation runs with at least one different random number generator.

References

Barker E, Kelsey J (2007) Recommendation for random number generation using deterministic random bit generators (revised). Technical Report NIST Special Publication 800-90. National Institute of Standards and Technology. http://csrc.nist.gov/publications/nistpubs/800-90/SP800-90revised_March2007.pdf

Bauke H, Mertens S (2004) Pseudo random coins show more heads than tails. J Stat Phys 114:1149–1169

Bauke H, Mertens S (2007) Random numbers for large-scale distributed Monte Carlo simulations. Phys Rev E (Statist Nonlin Soft Matter Phys) 75:066701. doi:10.1103/PhysRevE.75.066701. http://link.aps.org/abstract/PRE/v75/e066701

Brent RP (2006) Fast and reliable random number generators for scientific computing. In: Proceedings of the PARA'04 workshop on the state-of-the-art in scientific computing. Lect notes comput sci, vol 3732. Springer, Berlin, pp 1–10. http://wwwmaths.anu.edu.au/~brent/pub/pub217.html

Compagner A (1995) Operational conditions for random-number generation. Phys Rev E 52(5):5634–5645. doi:10.1103/PhysRevE.52.5634

Devroye L (1986) Non-uniform random variate generation. Springer, New York. Out of print. Available at http://cg.scs.carleton.ca/~luc/rnbookindex.html

Doornik JA (2007) Conversion of high-period random numbers to floating point. ACM Trans Model Comput Simul 17:3. http://doi.acm.org/10.1145/1189756.1189759

Dworkin M (2001) Recommendation for block cipher modes of operation: methods and techniques. Technical Report NIST Special Publication 800-38A. National Institute of Standards and Technology. http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

Eichenauer J, Lehn J (1986) A nonlinear congruential pseudorandom number generator. Stat Hefte 27:315–326

Ferrenberg AM, Landau DP, Wong YJ (1992) Monte Carlo simulations: Hidden errors from "good" random number generators. Phys Rev Lett 69(23):3382–3384. doi:10.1103/PhysRevLett.69.3382

Galassi M, Davies J, Theiler J, Gough B, Jungman G, Booth M, Rossi F (2001) GNU scientific library reference manual. Network Theory, Bristol. http://sources.redhat.com/gsl

Gentle JE (2003) Random number generation and Monte Carlo methods, 2nd edn. Springer Science + Business Media, New York

Hansel D, Mato G, Meunier C, Neltner L (1998) On numerical simulations of integrate-and-fire neural networks. Neural Comput 10:467–483

Haramoto H, Matsumoto M, Nishimura T, Panneton F, L'Ecuyer P (2008) Efficient jump ahead for $\mathbb{F}_2$-linear random number generators. INFORMS J Comput 20(3):385–390. http://www.iro.umontreal.ca/~lecuyer/myftp/papers/jumpf2.pdf

Hellekalek P, Wegenkittl S (2003) Empirical evidence concerning AES. ACM Trans Model Comput Simul 13:322–333

Kim C, Choe GH, Kim DH (2008) Test of randomness by the gambler's ruin algorithm. Appl Math Comput 199:195–210. doi:10.1016/j.amc.2007.09.060

Knuth DE (1998) The art of computer programming, vol 2, 3rd edn. Addison-Wesley, Reading

L'Ecuyer P (1999) Good parameters and implementations for combined multiple recursive random number generators. Oper Res 47:159–164

L'Ecuyer P (2004) Random number generation. In: Gentle JE, Haerdle W, Mori Y (eds) Handbook of computational statistics. Springer, Berlin, pp 35–70. http://www.iro.umontreal.ca/~lecuyer/myftp/papers/handstat.pdf

L'Ecuyer P, Panneton F (2005) Fast random number generators based on linear recurrences modulo 2: overview and comparison. In: Kuhl ME, Steiger NM, Armstrong FB, Jones JA (eds) Proceedings of the 2005 winter simulation conference, pp 110–119

L'Ecuyer P, Simard R (2007) TestU01: A C library for empirical testing of random number generators. ACM Trans Math Softw 33:22. Article 22, 40 pages. doi:10.1145/1268776.1268777. http://www.iro.umontreal.ca/~simardr/testu01/tu01.html

L'Ecuyer P, Simard R, Chen EJ, Kelton WD (2002) An object-oriented random-number package with many long streams and substreams. Oper Res 50:1073–1075

Lewis PAW, Goodman AS, Miller JM (1969) A pseudo-random number generator for the System/360. IBM Syst J 8:136–146

Lüscher M (1994) A portable high-quality random number generator for lattice field theory simulations. Comput Phys Commun 79:100–110

Marsaglia G, Tsang WW (2004) The 64-bit universal RNG. Statist Probab Lett 66:183–187

Mascagni M, Srinivasan A (2000) Algorithm 806: SPRNG: a scalable library for pseudorandom number generation. ACM Trans Math Softw 26(3):436–461. http://doi.acm.org/10.1145/358407.358427

Mascagni M, Srinivasan A (2004) Parameterizing parallel multiplicative lagged-Fibonacci generators. Parallel Comput 30:899–916

Matsumoto M, Nishimura T (1998) Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Trans Model Comput Simul 8:3–30

Matsumoto M, Saito M, Haramoto H, Nishimura T (2006) Pseudorandom number generation: impossibility and compromise. J Univers Comput Sci 12:672–690

Matsumoto M, Wada I, Kuramoto A, Ashihara H (2007) Common defects in initialization of pseudorandom number generators. ACM Trans Model Comput Simul 17:15. http://doi.acm.org/10.1145/1276927.1276928

Panneton F, L'Ecuyer P, Matsumoto M (2006) Improved long-period generators based on linear recurrences module 2. ACM Trans Math Softw 32:1–16

Park SK, Miller KW (1988) Random number generators: good ones are hard to find. Commun ACM 31:1192–1201

Plesser HE, Jahnsen AG (2010) Re-seeding invalidates tests of random number generators. Appl Math Comput 217:339–346. doi:10.1016/j.amc.2010.05.066

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical recipes in C, 2nd edn. Cambridge University Press, Cambridge

Saito M, Matsumoto M (2008) SIMD-oriented fast Mersenne twister: a 128-bit pseudorandom number generator. In: Keller A, Heinrich S, Niederreiter H (eds) Monte Carlo and quasi-Monte Carlo methods 2006. Springer, Berlin, pp 607–622

Tzeng S, Wei LY (2008) Parallel white noise generation on a GPU via cryptographic hash. In: I3D '08: Proceedings of the 2008 symposium on interactive 3D graphics and games. Microsoft Technical Report TR-2007-141. http://research.microsoft.com/research/pubs/view.aspx?tr_id=1384

Chapter 20
Practically Trivial Parallel Data Processing in a Neuroscience Laboratory

Michael Denker, Bernd Wiebelt, Denny Fliegner, Markus Diesmann, and Abigail Morrison

Abstract The complexity of modern data analysis techniques and the increasing amounts of data gushing out from neuroscientific experiments place new demands on the computing infrastructure required for data processing. The needs exceed the speed and memory constraints of a classical serial program design and require scientists to parallelize their analysis processes on distributed computer systems. In this chapter we explore, step by step, how to transform a typical data analysis program into a parallelized application. On the conceptual level, we demonstrate how to identify those parts of a serial program best suited for parallel execution. On the level of the practical implementation, we introduce four methods that assist in managing and distributing the parallelized code. By combining high-level scientific programming languages with modern techniques for job control and metaprogramming, no knowledge of system-level parallelization and the hardware architecture is required. We describe the solutions in a general fashion to facilitate the transfer of insights to the specific software and computer system environment of a particular laboratory.

The last two decades have seen a rapid gain in interest for increasingly complex methods employed in the analysis of neuronal spike train data, many of which are highlighted in this book. Partly, this trend may be attributed to a new quality of interaction between researchers in theoretical and experimental fields. However, equally important, data analysis with such techniques could not have been implemented without the steady increase in computation speed and memory storage of modern computers. There are three primary challenges in spike train analysis where increased computing capabilities are direly needed. First, many ad-

M. Denker (✉)
Laboratory for Statistical Neuroscience, RIKEN Brain Science Institute, 2-1 Hirosawa, Wakoshi, Saitama 351-0198, Japan
e-mail: mdenker@brain.riken.jp
url: http://www.cnpsn.brain.riken.jp

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0_20, © Springer Science+Business Media, LLC 2010

vanced algorithms are computationally expensive. Second, neuronal data sets are typically large, so that a terabyte of data is not a rare sight. In past times, data was often downsampled and reduced in order to accommodate the technology of the time, at the price of discarding the raw data forever. Thus, many questions that first arose during the analysis could not be answered. Modern storage technology allows the electrophysiologist to retain the complete recorded data set for reference. In addition to this increased amount of data coming from each electrode, the number of electrodes has also steadily increased, and techniques such as tetrodes or laminar electrodes provide multiple data streams per electrode. The third challenge stems from the observation that neuronal data typically exhibits nonstationary behavior, which complicates the task of finding the correct null hypothesis to assess the significance of an analysis. With modern computer power, it is possible to utilize a data-based approach to tackle significance estimation: surrogate techniques (see also Chap. 17). In this framework the original data is modified in a specific way so as to keep some aspects of the data (e.g., the nonstationary nature of the data), while deliberately destroying others (i.e., those described by the test parameter). Repeating this procedure many times estimates the expected distribution of the test parameter under the null hypothesis. The caveat lies in the repetition: N surrogates will increase the calculation time by a factor of about N.

As we have seen, the increased complexity of analysis, the large amount of recorded data, and the need for surrogate techniques increases demand for both speed and memory. Although the memory capacity of computers can still be expected to show gradual improvement, the same can no longer be said for the speed of the computer processors since the clock-speed of the basic processing unit is unlikely to become significantly higher in the near future. However, a different trend compensates for the stalling processor speeds: instead of having just one processor in a given machine, the computer possesses several. These are either realized as an array of CPU chips, or several subprocessors, called cores, within one physical chip (e.g., dual-core CPUs). In the following, we will use the terms *cores* or *processors* synonymously to denote a computer's processing units independent of their physical realization, e.g., a computer with 2 dual-core CPUs has 4 cores (processors). In this chapter we will not cover how distinguishing between the two can be utilized for optimization purposes. The design of having several cores share a computer's components (e.g., memory) is cost-effective and enables a substantially more efficient use of resources than a single core can realize. Programs started on the computer request a time share on the cores, and both the hardware and the operating system take care to balance the load and optimize performance of programs competing for the different processor resources.

Although this design intrinsically confers advantages for typical office computer usage (where the user will have several distinct applications running at the same time), the programming neuroscientist is faced with a severe problem: without further work, an analysis program will typically only run on one core and hence will not profit from the additional number of cores. Whereas some high-level languages, such as *Python* or *Matlab*, provide the option to have their internal func-

tions use several cores in parallel automatically (using a technique called *multithreading*, Butenhof 1997), the increase in speed that can be achieved by relying on just a few parallelized built-in methods is severely limited by the *serial* (sequential) design of the original program (Amdahl's law, see, e.g., Miller and Boxer 2005; Wilkinson and Allen 2004). Therefore, in order to accommodate the future demands of data analysis, neuroscientists need to become acquainted with the programming concept of *trivial parallelization*: splitting up the analysis into independent parts that can be computed in parallel without communication (Bader 2008). Applications for which this decomposition into independent processes is possible are known as *embarrassingly parallel* (Foster 1995). A major advantage that comes as an almost free bonus of a parallel program is the possibility of distributing it not merely over the cores of a single computer, but over all the cores of a set of independent *nodes* (computers) belonging to a *cluster* (or a *grid*, see Foster and Kesselman 2004). With the increased availability of cluster systems, the effort of parallelization provides high rewards in terms of speed gain and opens up the possibility of tackling the large amounts of data that will likely become commonplace in electrophysiology in years to come.

Our text summarizes the experience we have gained since we started to use trivial parallelization (Fliegner et al. 2003) and is inspired by the seminal work of Gerstein et al. (1983) on the design of a laboratory for multineuron studies. In the following sections, we provide a hands-on tutorial using a simple example task to help get the reader started with the idea of trivially parallel programming. The details of implementation naturally depend on a variety of factors, in particular, the computer hardware and software installation available to the reader. In particular, if you consider using a computer cluster shared by other parties, you will probably not have any influence on the methods available to you depending on the setup of the system. Therefore, we present the following approaches in a general style that emphasizes the common concepts of adapting an existing serial program to a parallelization scheme, rather than focusing on the technical details of a particular hardware and software setup. We describe parallelization on four levels of increasing complexity: manual and automated parallelization on the single multicore computer, queuing systems typically found on clusters of computers, and high-level interfaces to coordinate parallelized jobs that are available in some programming languages. These methods should be easy to adapt to most parallel computing environments and provide a starting point when designing and installing a high-performance computing infrastructure from scratch. All examples assume a Linux-based system, as scientifically used clusters are commonly UNIX-based. However, similar tools and approaches are also available for other operating systems. The code examples in this chapter are expressed in the *Python* programming language (http://www.python.org), which is presently gaining importance in scientific computing (Langtangen 2006) and neuroscience (Kötter 2009). The $ symbol indicates commands run from the Linux (bash) shell prompt (Newham and Rosenblatt 1995). All code presented in this chapter, in addition to further programs not reproduced here, is available at the web site associated with this book (http://www.apst.spiketrain-analysis.org).

Listing 20.1 The original serial program.

```
import pickle
from numpy import *

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#read spike data
f = open('./spikes.dat', 'r')
spikes = pickle.load(f)
f.close()

#preallocate result variables
num_ccs     = (num_neurons**2 - num_neurons)/2
cc_orig     = zeros((num_ccs,2*maxlag+1))
cc_surrs    = zeros((num_ccs,2*maxlag+1,num_surrs))
idxrange    = range(num_bins-maxlag,num_bins+maxlag+1)
row = 0

#for all pairs ni,nj such that nj > ni
for ni in range(num_neurons-1):
    for nj in range(ni+1,num_neurons):
        cc_orig[row,:] = correlate(spikes[ni,:],spikes[nj,:],"full")[idxrange]

        num_spikes_i = sum(spikes[ni,:])
        num_spikes_j = sum(spikes[nj,:])
        for surrogate in range(num_surrs):
            surr_i = zeros(num_bins)
            surr_i[random.random_integers(0,num_bins-1,num_spikes_i)] = 1
            surr_j = zeros(num_bins)
            surr_j[random.random_integers(0,num_bins-1,num_spikes_j)] = 1
            cc_surrs[row,:,surrogate] = correlate(surr_i,surr_j,"full")[
                idxrange]
        row = row +1

#save results
f = open('./result_cc_originals.dat','w')
pickle.dump(cc_orig,f)
f.close()
f = open('./result_cc_surrogates.dat','w')
pickle.dump(cc_surrs,f)
f.close()
```

20.1 Introducing a Simple Serial Program

Let us first introduce a serial program of the type one encounters in the analysis of neuronal data. Assume that we have recorded spike times from $num\_neurons = 100$ neurons in parallel over a duration of $num\_secs = 10$ s each. The spike data is provided as a binary vector with millisecond binning, i.e., $num\_bins = 10{,}000$ bins. We are interested in the mutual cross-correlation functions between all pairs, retaining a lag of up to $maxlag = 200$ ms. In order to assess the significance of all pairwise correlations, we calculate the cross-correlations for $num\_surrs = 1{,}000$ surrogate data sets for each neuron pair. For the sake of simplicity, each surrogate is generated by randomly redistributing the times of the original spikes. A simple program to accomplish this task is given in Listing 20.1.

First, parameters describing the experiment (*num_neurons*, *num_secs*, *num_bins*) and parameters of the subsequent analysis (*maxlag*, *num_surrs*) are imported from

two separate files, which contain nothing but the assignment of the values given above to the respective variables. Later, when we separate the program into individual parts during the parallelization process, it will pay off to have put parameters into such extra files rather than hard-wiring them into the code, as this way we are certain that different parts of the program use the same parameter values. In a second step, the data is loaded from file into the variable *spikes*. Next, two `for`-loops with loop control variables *ni* and *nj* realize all combinations of cell pairs. As the cross-correlation function is symmetrical, only those pairs are considered where the ID of the first neuron (*ni*) is lower than that of the second (*nj*). In practice this is realized by letting the inner *nj*-loop start at $ni + 1$, i.e., dependent on the value *ni* of the outer loop. In Python, array indexing starts from zero, so it is convenient to use neuron IDs between 0 and 99 for the 100 neurons. Thus, in this example, *ni* runs from 0 to 98, and *nj* from $ni + 1$ to 99 (for the particulars of the corresponding Python syntax, see Langtangen 2006). The enclosing block of these `for`-loops can be seen as two parts. First, the cross-correlation of both neurons is calculated. Second, each iteration of an additional `for`-loop running from 1 to *num_surrs* creates one surrogate spike train pair and calculates its cross-correlation. The original and all 1,000 surrogate cross-correlations are then inserted into the structures *cc_orig* and *cc_surrs*, respectively. The last lines, outside the `for`-loops, save the final results to disk in two separate files.

As simple as the program seems, it is not without two serious problems. First, the computation time is rather long. The block enclosed by the nj-loop (calculating 1,001 cross correlations) takes on the order of two minutes on a modern desktop computer. Therefore, all 4,950 combinations of neurons would take about 150 hours. Although this time span is still bearable, if the computations performed in the inner loop become more involved, for example, if we use a more sophisticated method to create the surrogate, the computation time might easily get into the range of weeks. A second problem is the memory requirement of the result variable: 4,950 combinations $\times$ 1,001 cross-correlations $\times$ 401 ms lag time $\times$ 8 bytes to store each value per lag time bin amounts to roughly 15 GB, which exceeds the typical memory capacity of present-day workstations. In this example, we would have no choice other than to rewrite the code so that it continuously frees memory by writing smaller data segments to disk during the calculation, as opposed to saving the whole block of results at the end. For instance, a natural choice would be to save the data separately for each neuron combination (just before the end of the *nj*-loop) in a file whose name clearly identifies the corresponding pair of neuron IDs *ni* and *nj* for later reference. In short, even simple programs may be unexpectedly demanding in terms of computing power and memory consumption.

20.2 The Idea of Trivial Parallelization

20.2.1 Theoretical Considerations

The principal challenge posed by the trivial parallelization process is to identify those parts of a program that are repeatedly executed (i.e., occur within a loop), and where each individual execution is independent of the results of any other execution. Such independent parts of the program can be executed in parallel, instead of sequentially. In the following, we term each execution of an independent part a *job*. In contrast, a nontrivial parallelization involves heterogeneous parts of the program running in parallel and/or communication between the parallel execution streams. Fortunately, in typical applications of data analysis in electrophysiology (and many other disciplines), this identification process is a relatively straightforward task, as virtually all analyses are carried out in some sort of looped structure. In the example Listing 20.1 we loop over individual neuron pairs and the individual surrogates. Other common loops iterate animals, recording sessions, stimuli, time windows, or technical parameters of the analysis (e.g., a defined bin width). Typically, the analysis performed in each loop iteration is independent of every other, for example, the spike data in rat A is independent of that in rat B, or a PSTH with a large bin width can be computed independently of a PSTH with a smaller one. Therefore, the individual iterations of these types of loops are the natural choice for defining independent jobs.

In many cases, the loop that is best suitable for parallelization is simply the outermost loop (in our example, the *ni*-loop). However, sometimes it is worth considering a loop that is more deeply nested. There is no rule to say which loop is best, but here are some points to consider:

- Optimization of run time: a job's run time should be neither too short nor too long. Having a large number of jobs that each run for a very short time (on the order of seconds) wastes a lot of time on *overhead*: the time required to start, initialize, and finalize each job. Moreover, such a strategy makes it difficult to monitor all the jobs and results in an inconveniently large number of output files (at least one per job). Conversely, having a few long jobs (on the order of days) is not optimal either, since they might not fully exploit the number of available cores, and any tendency for differing run times will be exacerbated (unbalanced jobs, see below). On a practical note, the longer a single job runs, the longer it may take to spot programming errors in the code. As a rule-of-thumb, a run time on the order of minutes to hours is a good choice between the two extremes. In particular, one should take care that the overhead required is small compared to the actual computation within the loop.
- Ease of implementation: inner loops often depend on variables that are set in previous iterations or by outer loops, therefore they can be more difficult to isolate. In this case, these variables must be passed appropriately to each job.
- Equalization of run time: an optimal parallelization has balanced jobs that take equal amounts of time; this is also known as *load balancing*. Assume the case

where each job analyzes a specific recording session of an experiment: if the sessions differ in length, e.g., session A takes 5 min and session B 200 min, the complete analysis will still take 200 min. If the run times are unpredictable, this can be compensated by creating more jobs than the number of processors available; a processor that finishes a job quickly can start on the next, while another one is still working on a long running job.

Once we have selected a loop to transform into a job, we must think about parameterizing the loop. This will later help to easily differentiate between individual instances of the job. In some cases this is trivial; if a loop extends over recording sessions labeled by sequential integer IDs, then these already uniquely identify each loop iteration. However, sometimes a loop runs over more complicated values, like floating point numbers, strings, or even complex data types. In this case, we recommend introducing a *job index* k as an integer variable to replace the former loop variable, and which can be mapped back to the original parameter via a predefined look-up table $Z(k)$. Thus, each job is identified by a unique index k, and no round-off errors or string conversions can confuse the association between a particular job and its parameter. For example, assume that a loop runs over different stimulus intensities given as $Z = [1.4325, 4.3214, 10.5423]$ or different animals given as $Z = [\text{'monkeyA'}, \text{'monkeyB'}]$. In this case a practical approach is to include the look-up table Z in the analysis parameter file (`analysis_params.py` in our example), use $k = 0, 1, \ldots$ as the job index, and have each job extract its actual parameter value as $Z[k]$. A similar procedure can be employed when $n > 1$ parameters should be varied: Z stores the n-tuple of all parameter configurations, and each value of k indexes a specific configuration.

The next step is to isolate the parameterized loop and transform it into a standalone job. Additionally, a *distributor* program may be required that takes care of executing the individual jobs. As outlined above, the job is derived from a loop, whereas the distributor is typically derived from the rest of the program and ensures that the required number of jobs are executed with the correct parameterization. Several mechanisms to achieve this are presented in Sect. 20.3, Sect. 20.4, and Sect. 20.6. Depending on the circumstances, the distributor may also serve as a *collector* of results or perform further analysis steps once all jobs have finished. This topic will be covered in more detail in Sect. 20.5.

Three principal changes must be carried out to convert a loop into a job. First, all data necessary for the loop to run must be loaded when the job starts. Second, the variable used to control the loop must be converted to the job index k, possibly using a look-up table Z. Third, results that used to be calculated in one iteration of the loop and are now calculated by the job must be saved to disk before the job terminates. The name of this results file should contain the index k for later reference.

Note that for most parallelization mechanisms (here, Sect. 20.3 and Sect. 20.4), the job must be a program that is executable from the command line of the computer that is running it. For some interpreted languages, such as *Matlab*, it can be advisable to compile the job script into a standalone application that can be run independently of whether the interpreter is installed on a specific machine. A further situation where compiling is advisable arises when the required number of parallel instances

of the job is prohibited by the specific licensing model of the interpreter application. Compiled code is often not subject to limitations regarding the number of instances executed in parallel.

20.2.2 Basic Parallelization of the Example Program

Let us now apply the concepts introduced in Sect. 20.2.1 to our example program (Listing 20.1). Each iteration of the *ni*-loop is independent and runs for between two minutes and about three hours, depending on the value of *ni*. Although this is not optimal load balancing, we start with the parallelization of this loop as it is particularly easy to extract and results in a manageable number of 99 jobs (the loop ranges from 0 to 98). In Sect. 20.7 we discuss better balanced alternatives. As *ni* is a plain integer value that denotes the unique ID of the first neuron of a pair, we can incorporate it into our parallel job as the job index k without the need of a look-up table Z.

Let us now transform the *ni*-loop into a job for parallel distribution. In the subsequent sections, for simplicity, we assume that this standalone program is called `job`. The result of the three-step recipe is shown in Listing 20.2. The first lines are the typical lines that initialize the program. Note that we first need to load necessary data and parameters. This is part of the overhead, i.e., the set of instructions that need to be performed by each individual call to the job. Recall that in order for parallel programs to be efficient, the time required for the overhead should be small compared to the actual calculations. In our example this constraint is fulfilled, as the data loads in a matter of seconds. On the following lines, we see that the variable *ni* that is used to control the loop in Listing 20.1 is now passed to the program in string form as the job index k, whereupon it is converted into an integer. A call to the program with the argument k performs the kth iteration of the former loop. For example, to calculate the 16th iteration of the original loop that calculates only the correlations of neuron 15 with neurons 16 to 99, we would call

```
$ job 15
```

The main part of the program is unchanged from the body of the original *ni*-loop in Listing 20.1, except that on the last lines we save the results to disk, using the job index variable k to create a uniquely identifiable file name for each iteration of the job, in this case `result_cc_originals_`k`.dat` and `result_cc_surrogates_`k`.dat`. Note that we obtain 198 different output files, which may at first seem bothersome. However, in Sect. 20.1 we discussed the fact that a single output file containing all the data would be extremely large, so splitting the results into several files may well be desirable.

In the following sections we demonstrate how to run the new program in parallel. Several methods are available, and the design of the distributor program will depend on the method chosen. In Sect. 20.3 we present two simple techniques that are restricted to running parallel jobs on a computer with multiple cores. In Sect. 20.4 and Sect. 20.5 we show how to utilize more complex systems to distribute parallel jobs on different computers and how to deal with dependencies between jobs. In

Listing 20.2 The generic job created from the example program.

```
import sys
import pickle
from numpy import *

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#read neuron id from the command line
k_str = sys.argv[1]
k = int(k_str)

#read spike data
f = open('./spikes.dat', 'r')
spikes = pickle.load(f)
f.close()

#preallocate result variables
num_ccs  = num_neurons - k - 1
cc_orig  = zeros((num_ccs,2*maxlag+1))
cc_surrs = zeros((num_ccs,2*maxlag+1,num_surrs))
idxrange = range(num_bins-maxlag,num_bins+maxlag+1)
row = 0

#for all pairs neuron,nj such that nj > neuron
for nj in range(k+1,num_neurons):
    cc_orig[row,:] = correlate(spikes[k,:],spikes[nj,:],"full")[idxrange]

    num_spikes_i = sum(spikes[k,:])
    num_spikes_j = sum(spikes[nj,:])
    for surrogate in range(num_surrs):
        surr_i = zeros(num_bins)
        surr_i[random.random_integers(0,num_bins-1,num_spikes_i)] = 1
        surr_j = zeros(num_bins)
        surr_j[random.random_integers(0,num_bins-1,num_spikes_j)] = 1
        cc_surrs[row,:,surrogate] = correlate(surr_i,surr_j,"full")[idxrange]
    row = row +1

#save results
f=open('./result_cc_originals_'+k_str+'.dat', 'w')
pickle.dump(cc_orig,f)
f.close()
f=open('./result_cc_surrogates_'+k_str+'.dat', 'w')
pickle.dump(cc_surrs,f)
f.close()
```

Sect. 20.6 we present a different approach to parallelization that allows jobs to be defined and executed in a flexible fashion using the high-level parallelization interfaces that have been implemented in some programming languages.

20.3 Starting a Parallel Job on a Single Multicore Computer

20.3.1 A Manual Solution: Screen

As an instructive first step, let us discuss the most simple way to distribute jobs to the different cores of a single computer: by hand. To this end, we simply start all 99

instances of the program in such a way that they are executed in the background and independent of the calling terminal:

```
$ nohup job 0 &
$ nohup job 1 &
...
$ nohup job 98 &
```

On each line the program `nohup` (a contraction of "no hang up") calls `job` so that the process is detached from the shell (i.e., closing the console will not kill the job) and returns control immediately to the shell via the & directive. On each call of `job` we pass as an argument the job index k. It is converted into the variable k in the job program Listing 20.2.

A more interactive way to manually start a number of jobs that can be detached from the current terminal is to use the program `screen`. Calling

```
$ screen
```

provides a virtual console, where any of the jobs may be started, e.g.,

```
$ job 0
```

Using the keyboard short-cut `Ctrl` − `a` followed by `c`, a new console is created in which another job can be started. The shortcuts `Ctrl` − `a` `n` (next) and `Ctrl` − `a` `p` (previous) may be used to cycle between these virtual consoles. On each console, the job outputs may be monitored. Most importantly, pressing `Ctrl` − `a` `d` detaches the virtual consoles from your shell and continues execution of all running jobs in the background. The console may be reattached to your terminal at any later point by calling

```
$ screen -R
```

which allows you to monitor the progress of your jobs. See the installed manual page for screen (`man screen`) for a complete list of commands.

However, in addition to the laborious task of having to start each job by hand, there is a more fundamental problem with this approach: the above procedure only works if the computer has at least 99 cores available. A typical serial job requires one core (unless it starts multiple *threads*, see Sect. 20.7). If fewer cores are present than running jobs, individual jobs compete for time, creating additional overhead for the operating system and resulting in an overall slower execution. In addition, all 99 jobs must fit into the memory of the machine. Given our earlier estimate of 15 GB for the example program, we see that memory can be an equally important constraint. Assuming that the total amount of memory of the computer divided by the number of cores is sufficient for any one job, a better solution would be a smart mechanism that automatically starts jobs from a given list of jobs in such a way that only one job per core is running at any given point in time.

20.3.2 An Automated Solution: Make

A straightforward way to accomplish this is the program `make` (Oram and Talbott 1991). Although this program was originally intended to aid the process of compiling programs, its features are ideally suited to provide an easy and flexible way of parallelizing a program on a personal workstation. Central to the idea of `make` is a script known as a *makefile* that lists *targets*, the commands to execute for each target, and target dependencies defining the order in which targets are executed. A simple makefile to run our jobs might look like this (note that all indentations are essential):

```
all: dep0 dep1 dep2 dep3 ... dep98
        echo Done

dep0:
        job 0
dep1:
        job 1
...
dep98:
        job 98
```

The first line specifies a target "all", which depends on 99 other targets: "dep0" up to "dep98". Therefore, the commands of target "all" will execute (i.e., printing the message "Done" on the screen) only if all these 99 dependent targets have finished executing. On the following lines the individual targets "dep*X*" are defined: they have no dependency by themselves, and each executes exactly one job with a unique value for the job index k.

Assuming the makefile is stored in the file `job.make`, a call of

```
$ make job.make
```

does not confer any advantage: make will first sequentially execute the command(s) of the first target without dependencies ("dep0"), then of the second ("dep1"), and so on. Finally, when "dep98" has been executed, it will have fulfilled all required targets to execute the `echo` command of last remaining target "all". The trick is to call `make` with a parameter that indicates that targets which do not depend on one another should be executed in parallel:

```
$ make -j P job.make
```

where P is the number of parallel process that may be started in parallel (i.e., on a quad-core processor, you would specify 4). In this example, `make` would simultaneously execute the commands that belong to "dep0" to "dep3", and start "dep4" as soon as one of the four finishes.

Listing 20.3 The generic distributor program for using `make` to parallelize.

```
import os
import glob

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#number of simultaneous jobs
ncores=4;

#filename of makefile
make_name='./job.make'

#write makefile
f = open(make_name, 'w')
f.write('all:␣');
for k in range(num_neurons-1):
    f.write('dep'+str(k)+'␣');
f.write('\n\n')
for k in range(num_neurons-1):
    f.write('dep'+str(k)+':\n');
    f.write('\tjob␣'+str(k)+'\n');
f.close()

#start make
os.system('make␣-j␣'+str(ncores)+'␣-f␣'+make_name);
```

Writing a makefile like the one above can be a tiresome job. In addition, the analysis parameters must be entered explicitly. In our example we need one entry per neuron; the makefile would need to be adapted in order to perform an identical analysis on a different number of neurons. As previously indicated (see Sect. 20.2.1), a more elegant and error-safe way is to write a distributor program. The task of this program is to create all necessary files needed for parallelization (here the makefile) and then start the actual execution. A distributor for our example is given in Listing 20.3. The program first loads the variables describing the data and the analysis, allowing the distributor to automatically detect how many neurons (*num_neurons*) were recorded and therefore how many jobs should be run. The main body of the program employs metaprogramming (Czarnecki and Eisenecker 2000): it creates the makefile `job.make` discussed above by opening a text file and entering the corresponding lines in an automated fashion. Lastly, a system call to `make` executes the newly created makefile using *ncores* cores in parallel (here 4).

20.4 Using a Queuing System to Distribute Jobs

As we have seen in the previous section, the program `make` provides an intuitive and easy way to execute a parallelized program on the different cores of a single computer. However, there are two important scenarios in which `make` turns out to be too simplistic. First, a user may have access to several different computers. However, `make` cannot exploit this by distributing the jobs over multiple computers. Therefore, the maximum number of parallel jobs is restricted to the maximum

number of cores on a single computer. Second, large computer systems are typically shared by several users. However, `make` does not have knowledge about how much of the computers' resources are occupied by other users running their jobs on the same machine. Therefore, their jobs would compete for resources. In particular, if jobs demand more memory than is available, the machine will eventually crash.

To solve these problems, computing infrastructures of medium to large size are often equipped with a set of programs termed a *queuing system*. The description that follows is based on the syntax of the freely available TORQUE/MAUI system (http://www.clusterresources.com), also known by the acronym PBS. Another popular queuing system is the Sun Grid Engine (http://gridengine.sunsource.net). On the whole, most queuing systems work on similar principles to those described here. The same holds true for *grid computing* environments (Foster and Kesselman 2004; Backofen et al. 2006), which provide a more complex version of the queuing infrastructure that combines the power of computing resources at distant sites.

Let us first outline the standard procedure for running a single job via a queuing system. A user who wishes to run a program first writes a small script, termed a *job description file* (jdf). The purpose of the jdf is similar to the makefile described in Sect. 20.3. It contains the name of the program(s) to run and some additional information on various parameters for the queuing system. In particular, it should list the resources required to run the program, such as the number of cores or the amount of memory. In a subsequent step, the jdf is submitted to the queuing system by a command typically named `qsub`. This command parses the requirements stated in the jdf file and enters a corresponding request in the queue, assigning it a unique, numerical job ID. An independent program, the queue server, periodically communicates with a set of computers (nodes) that are all assigned to devote their resources to the queue. As soon as any of these nodes reports a vacant slot, the queue server queries a scheduler program to select a jdf from the queue that fits the free resources of this node. The job is then executed on the available node and marked as "running" within the queue. Upon completion of the job, it is removed from the queue, and the freed resources will be assigned to the next appropriate jdf in the queue. When selecting a job, the scheduler considers not only the resources required versus resources available, but also tries to balance jobs from individual users, so that users receive a fair share of the processing power. The extent to which this balancing is implemented in different queuing systems varies widely, but for small setups (in terms of machines and users), a simple solution just balances the number of jobs per user at any point in time.

Knowing how to submit a single job to the queuing system, we now extend this concept to a set of parallel jobs in a straightforward way. For each execution of the job, a separate jdf file is created that calls the job with its corresponding job index k. In a second step, all jdfs are submitted to the queuing system. From then on, the queuing system takes care that the jobs are executed in parallel on all available nodes. The following jdf file could be used to submit our example program with job index $k = 10$:

```
#!/bin/bash
#PBS -q my_cluster
#PBS -l nodes=1:ppn=1,mem=2gb,walltime=24:00:00
#PBS -N CC
#PBS -d /home/user/my_cc_project

./job 10
```

The first line is a stereotypical line that indicates that all following commands should be executed using the `bash` shell as interpreter. In principle, however, the jdf could be written in any script language, even in Python directly. The following four lines all start with `#PBS` and provide information to the queue in terms of options of `qsub` (which will be explained in more depth in the following paragraphs). Before the queue server submits and executes a jdf file, it preprocesses these lines and extracts the directives contained within. All of the options following the `#PBS` statements could also have been directly passed to `qsub` on the command line, but by placing them explicitly in the jdf, they remain clearly visible for future reference. In the case of conflicting arguments, options given on the command line override those given in the jdf. The hash mark preceding these lines identifies the line as commentary for most shell interpreters. Therefore in our example the `bash` interpreter will ignore these queue directives when the jdf is actually executed. Note that `qsub` only interprets `#PBS` statements that are directly located at the top of the file. Furthermore, the format of these lines will differ for queuing systems that are not implemented by TORQUE. Finally, the last line of the jdf calls our job with $k = 10$.

A number of options to `qsub` control how jobs are sent to the queue. Although in many cases the system default values of the following options will work fine, it is still good practice to develop a habit of specifying them in order to accurately describe your job. First and foremost, `-q` *queue* selects the queue in which to run the job. In general, a queue server may provide several queues, which typically differ in the identity of the machines that are used to process jobs submitted to that queue. For example, you might have two server systems that run under a different architecture or operating system, or maybe a laboratory has a private cluster system and one that is shared with another laboratory. In such a case, one might use different queues to control on which category of machine a specific job is executed.

A further option `-l` *req* informs the queue about the requirements of the submitted job in terms of resources. Here, *req* is a comma separated list of those requirements that deviate from the default values of a particular queue. The number of nodes X and the number of processors per node Y for the single job submitted in the file (not including any other jobs you may submit independently of the current job) is passed using `-l nodes=`X`:ppn=`Y. In our single-threaded example, each job requires only one node and one core, and hence $X = Y = 1$ (which is typically the default). The memory requirement of our job is specified using `-l mem=`R, where is R is a human readable memory requirement, such as `2gb`. This value should be chosen with some care since different jobs compete for the shared memory of a single node: too low, and jobs might run out of memory; too high, and

the queue will run fewer jobs in parallel than it actually could. To illustrate the latter point, assume 16 jobs that each require 2 GB are submitted to a computer with 32 GB and 16 processors. Obviously, there is no problem in running all 16 jobs in parallel. However, if we were to generously specify `-l mem=16gb`, only 2 jobs would be executed at the same time, each one reserving the requested amount of memory. In our example program, we estimate that 2 GB is a reasonable number. Lastly, the maximum running time in hours, minutes, and seconds is controlled by `-l walltime=`$HH:MM:SS$, after which the job is mercilessly killed. This mechanism is intended to automatically remove hung jobs in larger queuing systems.

The remaining parameters are less crucial but useful nonetheless. The parameter `-d` specifies the working directory in which the job is started. The parameter `-N` assigns a user-defined internal name to the job that is used within the queuing system. By default, this name is the program's file name. A useful feature of most queuing systems is that they retain the output (stdout) and error output (stderr) created by the program during execution. These text streams are written to files that start with the job's name and end in `.o`$XXXX$ for output and `.e`$XXXX$ for error streams, where $XXXX$ indicates the unique job ID assigned to the job at time of submission to the queue. Use the option `-j oe` to merge the two output and error files into one, or specify `-o /dev/null` and `-e /dev/null` to disable the output and error files, respectively.

As in the case of `make`, the task of creating all the individual jdf files can be easily automated by a distributor program. For our simple example, such a program is provided in Listing 20.4. Again, the data is first loaded in order to extract the number of neurons, which determines the total number of jobs we need to submit. The variable q is assigned the name of the queue to which we want to submit the jobs. Then, a `for`-loop over the job index k creates a unique jdf file for each neuron with a unique file name parameterized by k. The contents of each jdf file resembles our example above, except that in each jdf the job is called with a different parameter k. The last line issues a system call to `qsub` to submit the newly created jdf files to the queue. Thus, this program creates and submits 99 uniquely parameterized jdf files for execution.

This approach is very general and should work without problems on any queueing system. However, creating a separate jdf for each job is neither particularly elegant nor efficient. Fortunately, advanced queuing systems provide the useful concept of *job arrays* which significantly simplify the process. The underlying idea is to include an option that instructs the queuing system to insert the job into the queue not once, but several times, each instance with a specific, user-defined integer value. This integer may then be exploited as the job index. Job arrays are initialized by supplying the additional parameter `-t` *range*, where *range* specifies the integer values that should be assigned to individual jobs. Valid examples for range would be `1,2,3` (execute the script three times with integer identifiers 1–3), and `10-15,1000` (execute the script seven times with integer identifiers 10, 11, 12, 13, 14, 15, and 1,000). This integer can be queried from within the jdf file—or the job program itself—via the system environment variable `$PBS_ARRAYID`. Consider the following modified jdf for our example:

Listing 20.4 A distributor program for the queuing system.

```
import os
import glob

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#queue to use
q='my_cluster'

#create one jdf file per job
for k in range(num_neurons-1):
    #filename of jdf file
    jdf_name='./job'+str(k)+'.jdf'
    f = open(jdf_name, 'w')
    f.write('#!/bin/bash\n')
    f.write('#PBS␣-q␣'+q+'\n')
    #...add more PBS options as required
    f.write('./job␣'+str(k)+'\n')
    f.close()

    #submit job
    os.system('qsub␣'+jdf_name);
```

```
#!/bin/bash
#PBS -q my_cluster
#PBS -l nodes=1:ppn=1,mem=2gb,walltime=24:00:00
#PBS -N CC
#PBS -d /home/user/my_cc_project
#PBS -t 0-98

./job $PBS_ARRAYID
```

Note that the option `-t` is used and that our job is called with the variable placeholder `$PBS_ARRAYID` instead of a fixed value. When `qsub` interprets this modified jdf, it will infer from the `-t` option that it should submit a total of 99 executions of this jdf script. Furthermore, it will set the `$PBS_ARRAYID` to 0 for the first of these calls, to 1 for the second, and so on.

The job IDs of a job array share an identical part, similar to the single submitted job, appended by a minus sign and the corresponding value of `$PBS_ARRAYID`. Assuming that the initial job submission was assigned the job ID 1234, the individual jobs would be assigned the IDs $1234-0,\ 1234-1,\ 1234-2, \ldots, 1234-98$. In conclusion, the concept of job arrays allows us to parallelize our job efficiently using an integer parameter and a single jdf. A modified distributor program for this type of job submission is listed in Listing 20.5. An analysis of the remaining differences between this distributor and that given in Listing 20.4 is left as an easy exercise for the reader.

After `qsub` has entered the jobs into the queue, it returns control to the user; the details of distributing jobs to available nodes is performed in the background. The

Listing 20.5 A distributor program for the queuing system using the job array concept.

```
import os
import glob

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#queue to use
q='my_cluster'

#filename of jdf file
jdf_name='./job.jdf'

f = open(jdf_name, 'w')
f.write('#!/bin/bash\n')
f.write('#PBS -q '+q+'\n')
f.write('#PBS -t 0-'+str(num_neurons-2)+'\n')
#...add more PBS options as required
f.write('./job $PBS_ARRAYID\n')
f.close()

#submit job
os.system('qsub '+jdf_name);
```

program `qstat` is used to gain an overview over the jobs in the queue, including several statistics such as their run time, and their status, which is in most situations coded by the letters Q (queued, not yet assigned to a computer), R (running), or C (complete). After completion, jobs are automatically removed from this list. Useful options to `qstat` are `-n1` for a more detailed output and `-u` *username* to restrict the list to jobs of a specified user. Furthermore, calling `qdel` *jobid* removes (and kills) the specified job or job array from the queuing system.

20.5 Introducing Job Dependencies

Often data analysis involves a sequence of several processing steps. In our example, we might be interested in merely collecting the generated data, or we may want to analyze how many of the cross-correlation functions have significant peaks. An example of such a program is provided with the online resources. Whatever the nature of the *post-processing*, the simplest approach is to wait for all jobs to finish and then to call the program that performs the subsequent analysis step manually. However, it is more efficient if the post-processing starts automatically once all computations are complete, especially if the structure of the analysis is complex. In this section, we introduce methods to realize such dependencies between jobs for each of the parallelization mechanisms described above.

Using `make` to parallelize a program intrinsically provides the possibility of including dependencies: in the makefile introduced in Sect. 20.3.2, we can just include the call to the post-processing program (in the following called `post-job` for simplicity) as a command that will be executed once the dependency "all" is fulfilled:

```
all: dep0 dep1 dep2 dep3 ... dep98
        post-job

dep0:
        job 0
dep1:
        job 1
...
dep98:
        job 98
```

Of course, there is no reason why the post-processing program should not itself be a distributor program for another set of parallel jobs.

If using a queuing system, things are slightly more complicated as we do not receive direct feedback when a job is finished. A general solution to the problem is to have each job write a specific file at the end of its completion, e.g., `job`*k*`.done`, where again *k* denotes the job index. The post-processing program is submitted to the queue immediately along with the parallel jobs and waits for all `*.done` files to be created before starting its analysis. However, the drawback of this solution is that the post-processing job blocks a slot in the queue without doing anything useful. Also, generating myriad `*.done` files is somewhat unsightly. A more practical method available on some queuing systems is to specify a job dependency directly as an option to the queue using the syntax:

```
#PBS -w depend=ok=JOBID[,JOBID[,...]]
```

A job submitted using this directive will not be run until all specified jobs have completed execution successfully and so does not block a slot unnecessarily. To this end, we need to retain the job IDs of the initial parallel jobs. They are provided as the output of `qsub`, possibly modified by an identifier of the used queue (e.g., `.my-cluster.edu`). If the job initiates subjobs using a job array (option `-t`), only the primary job ID is returned, the sub ID corresponding to the job index *k* must be added manually. To clarify this point, calling `qsub` with the option `-t 1-3` might return a string like `1234.my-cluster.edu`, but not explicitly the job IDs of the three single jobs making up the job array, `1234-1`, `1234-2`, and `1234-3`. Listing 20.6 shows a modified distributor script (using job arrays) for our example program that submits the post-processing job `post-job` to the queue with a dependency on completion of the original 98 jobs. To this end, the primary job ID is extracted from the first call to `qsub`, and by attaching the job indices *k* from 0 to 98, a dependency list is generated as a string. Finally, the post-processing job is submitted using a new jdf file that includes this dependency list. A similar approach can be taken to modify the corresponding distributor Listing 20.4 that does not utilize the job array functionality.

Listing 20.6 The modified distributor in a queue environment for calling a post-processing job after completion of the parallelized program.

```
import os
import glob
import subprocess

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#queue to use
q='my_cluster'

#filename of jdf file
jdf_name='./job.jdf'

f = open(jdf_name, 'w')
f.write('#!/bin/bash\n')
f.write('#PBS␣-q␣'+q+'\n')
f.write('#PBS␣-t␣0-'+str(num_neurons-2)+'\n')
#...add more PBS options as required
f.write('./job␣$PBS_ARRAYID\n')
f.close()

#modified statement to submit original jobs and save primary job ID
p = subprocess.Popen('qsub␣'+jdf_name, shell='true',stdout=subprocess.PIPE)
sts = os.waitpid(p.pid, 0)
jobid_str=p.stdout.read()

#add : and - to primary job ID
jobid_str=':'+jobid_str.split('.')[0]+'-'
#duplicate string, each time with a different secondary ID (job index)
#result is a dependency list of the form XXX-AA:XXX-BB:XXX-CC...
#where XXX is the primary job ID, and AA, BB,... are the job indices
jobdeplist=(jobid_str+jobid_str.join([str(i) for i in range(num_neurons-1)]))[
    1:]

#filename of jdf file for post processing
jdf_post_name='./post-job.jdf'

f = open(jdf_post_name, 'w')
f.write('#!/bin/bash\n')
f.write('#PBS␣-q␣'+q+'\n')
f.write('#PBS␣-w␣dependok='+jobdeplist+'\n')
#...add more PBS options as required
f.write('./post-job\n')
f.close()
os.system('qsub␣'+jdf_post_name)
```

20.6 Using Parallelization Libraries

The previous sections have dealt with various methods for managing the distribution of serial jobs that can be characterized by a job index k. However, many languages have libraries that permit a researcher to write programs that exploit parallelization from the outset. The advantages of this approach are that the individual jobs can communicate with each other, if necessary, and that the administration aspects, such as monitoring when individual jobs have finished, are generally taken care of behind the scenes. For C and Fortran, the most commonly

used libraries are based on the Message Passing Interface (MPI) standard (http://www.mcs.anl.gov/research/projects/mpi). For Python, several such libraries are available. Listing 20.7 is based on the Parallel Python module (http://www.parallelpython.com). Here we show a program which is intended for a shared-memory architecture; a version of the code adapted to cluster computing is available in the online repository.

The script consists of two parts: a function definition (`cc_surrogate_range`), which is analogous to the serial jobs defined in the previous sections, and a script section, which submits jobs and collates the results. First of all, the script initializes the job server with either the number of processors stated in the command line or through automatic detection of the number of cores on the local machine. Next, the script demonstrates a different approach to load balancing. Instead of dividing the work by parallelizing a loop, we consider the total number of operations to be performed and divide those in an equal fashion. Concretely, the script divides the total number of neuron pairs to be correlated by the number of cores to obtain the number of neuron pairs each job should process. This can easily be converted into start and end indices of a range of neuron pairs, assuming that each job increments a counter (here `idx`) over the double `for`-loop (`for ni ... for nj ...`). Note that, as a consequence, the serial function `cc_surrogate_range` contains all three nested `for`-loops of the original program (see Listing 20.1) and a fair distribution of loads is ensured.

The serial function can return any data type or collection of data types to the main program. To illustrate the flexibility of this approach without requiring huge amounts of memory, the serial function does not write the correlation functions to file. Instead, having calculated the correlation function for a neuron pair and all the corresponding surrogate trains, the surrogate correlation functions are sorted. The return value of the function consists of one 2D-array containing the correlation function for each neuron pair considered and one 3D-array containing the 5% and 95% values for the sorted surrogate correlations.

The first `for`-loop in the script section defines each job and sends it off using the routine `submit()`. In our case each job is very simple and consists of a function to be called (`cc_surrogate_range`), the parameters to be passed as arguments to the function (`params`), and the modules required by the function (`depmods`). The parameters to be passed to the function in this case include the start and end of the index range the job should process. Having submitted all the jobs, the second `for`-loop iterates through the list of submitted jobs and collates their results. In contrast to the methods introduced in Sect. 20.5, no explicit steps must be taken to wait for completion of the parallel jobs. The issue of job dependencies is taken care of behind the scenes, as the () operator used in the command `result = jobs[worker]()` automatically waits until the job is complete.

Such a framework allows an entire parallelized program including job definition, submission, results collation, and post-processing to be defined flexibly and compactly. Parallelization based on language-specific libraries can of course be combined with the parallelization based on queuing systems (see Sect. 20.4); the technical details of this depend on the systems and architectures involved.

Listing 20.7 The example program parallelized using the capabilities of the parallel Python module.

```
import pp
import sys
import pickle
import numpy

#load experiment and analysis parameters
from experiment_params import num_neurons,num_secs,num_bins
from analysis_params import maxlag,num_surrs

#cc_surrogate range calcates cc and surrogate cc for a given range of indices
def cc_surrogate_range(start_idx, end_idx, seed):
    #load experiment and analysis parameters
    from experiment_params import num_neurons,num_secs,num_bins
    from analysis_params import maxlag,num_surrs
    f = open('./spikes.dat', 'r')
    spikes = pickle.load(f)
    idx = 0
    row = 0
    my_cc_original = numpy.zeros((end_idx-start_idx,2*maxlag+1))
    my_cc_surrs = numpy.zeros((end_idx-start_idx,2*maxlag+1,2))
    idxrange = range(num_bins-maxlag,num_bins+maxlag+1)
    surrs_ij = numpy.zeros((num_surrs,2*maxlag+1))
    numpy.random.seed(seed)
    for ni in range(num_neurons-1):
        for nj in range(ni+1,num_neurons):
            #get to first index of relevant range
            if (idx < start_idx):
                idx = idx + 1
                continue
            #calculate cc and surrogate ccs for all indices in relevant range
            elif (idx < end_idx):
                my_cc_original[row,:] = numpy.correlate(spikes[ni,:],spikes[nj
                    ,:],"full")[idxrange]
                num_spikes_i = numpy.sum(spikes[ni,:])
                num_spikes_j = numpy.sum(spikes[nj,:])
                for surrogate in range(num_surrs):
                    surr_i = numpy.zeros(num_bins)
                    surr_i[numpy.random.random_integers(0,num_bins-1,
                        num_spikes_i)] = 1
                    surr_j = numpy.zeros(num_bins)
                    surr_j[numpy.random.random_integers(0,num_bins-1,
                        num_spikes_j)] = 1
                    surrs_ij[surrogate,:] = numpy.correlate(surr_i,surr_j,"full
                        ")[idxrange]
                #save point-wise 5% and 95% values of sorted surrogate ccs
                surrs_ij_sorted = numpy.sort(surrs_ij,axis=0)
                my_cc_surrs[row,:,0] = surrs_ij_sorted[round(num_surrs*0.95),:]
                my_cc_surrs[row,:,1] = surrs_ij_sorted[round(num_surrs*0.05),:]
                idx = idx + 1
                row = row + 1
            #reached end of relevant range; return results
            else:
                return [my_cc_original, my_cc_surrs]
    return [my_cc_original, my_cc_surrs]
```

Listing 20.7 (continued)

```
#script section starts here!
#tuple of all parallel python servers to connect with
ppservers = ()

if len(sys.argv) > 1:
    ncpus = int(sys.argv[1])
    #creates jobserver with ncpus workers
    job_server = pp.Server(ncpus, ppservers=ppservers)
else:
    #creates jobserver with automatically detected number of workers
    job_server = pp.Server(ppservers=ppservers)

nworkers = job_server.get_ncpus()
num_ccs = (num_neurons**2 - num_neurons)/2

#calculate number of pairs each worker should process
step = numpy.ceil(float(num_ccs)/nworkers)
start_idx = 0
end_idx = 0
starts = numpy.zeros((nworkers+1,))
starts[-1] = num_ccs

seed = 2398645
delta = 1782324
jobs = []

#send out jobs
for worker in range(nworkers):
    start_idx = end_idx
    end_idx = int(min((worker+1)*step,num_ccs))
    print start_idx, "␣->␣", end_idx - 1
    starts[worker] = start_idx
    params = start_idx, end_idx, seed,
    depmods = "numpy","pickle",
    jobs.append(job_server.submit(cc_surrogate_range,params,modules=depmods))
    seed = seed + delta

print "submitted␣all␣jobs"

#collect results from workers
cc_original = numpy.zeros((num_ccs,2*maxlag+1))
cc_surrs = numpy.zeros((num_ccs,2*maxlag+1,2))
for worker in numpy.arange(nworkers):
    start = starts[worker]
    end = starts[worker + 1]
    result = jobs[worker]()
    cc_original[start:end,:] = result[0]
    cc_surrs[start:end,:,:] = result[1]

#save results
f = open('./result_cc_originals.dat','w')
pickle.dump(cc_original,f)
f.close()
f = open('./result_cc_surrogates_conf.dat','w')
pickle.dump(cc_surrs,f)
f.close()
```

20.7 Concluding Remarks

A critical issue that has been mentioned on several occasions in this chapter concerns the load balancing between individual jobs. As we have outlined in Sect. 20.2.2, our original implementation of the job (Listing 20.2) is not optimal, as the runtime of the job depends on the job index k. A better load balance might be achieved by choosing a different loop to parallelize. For instance, we could have opted to rearrange the order of the loops and submit 1,000 jobs that each calculate exactly one surrogate cross-correlation for all combinations of neurons (see the online code repository for an implementation). In this case, all jobs would calculate an equal number of 4,950 cross-correlations. However, a word of caution: rearranging loops must be done with care, and even if the resulting code is executable, it may lead to a conceptually different analysis. As an alternative approach, we showed in Sect. 20.6 that the load may be balanced by maintaining the complete nested loop structure but assigning an equal number of neuron pairs to each of the jobs.

A matter of utmost importance—especially in the context of surrogates—is the issue of correctly initializing the random number generator in each call to the job. For the sake of simplicity only, we have ignored this issue in the first examples in this chapter. However, Listing 20.7 shows a very basic way of ensuring that each job at least receives a different random number seed for initialization. To this end, each job is assigned a unique seed according to the linear relation $seed + k \cdot delta$, where k is the job index, and *seed* and *delta* are arbitrarily defined at the beginning of the script. This seed is passed to the function `cc_surrogate_range` as a further parameter. Note that this strategy does not guarantee independence of the parallel random number streams. Please consult Chap. 19 for a thorough exploration of the issue of random number generation and the additional constraints of parallel generators, including appropriate initialization strategies (e.g., using a high-level interface, such as the Python function `random.jumpahead(n)`).

Lastly, let us mention again that even if it appears that a job is single-threaded and will therefore only occupy one core, some applications may try to automatically use more than one core by means of multithreading its built-in functions. The Matlab interpreter is one such example. The increased demand for processors of the single job must be reflected in the parallelization procedure in two steps. First, it is typically possible to instruct the application to use up to a specified maximum number n of threads only (in Matlab, e.g., using the function `maxNumCompThreads(n)`). Second, if $n > 1$, this maximum number must either be passed to `make` (dividing the value passed via the `-j` option by n) or to the queue (using the `-l nodes=1:ppn=`n option to request one node with n cores per job). This guarantees that each job will have enough cores to run on without competition from other jobs. It is worth considering turning off multithreading ($n = 1$) altogether, since custom-tailored parallelization schemes such as those presented throughout this chapter are likely to be superior in efficiently utilizing the processor power of the available cores.

Ten years after we introduced parallel computing in our laboratory to simulate large-scale models of neuronal networks, we finally also routinely use these techniques in our data analysis projects. This is despite the fact that neuronal network

simulations require nontrivial parallelization (Morrison et al. 2005), whereas, as argued above, a large class of data analysis problems can be solved by trivial parallelization. The main reason for this seemingly paradoxical delay is probably that of exigency: large-scale simulations cannot be performed without harnessing the collective working memory of a computer cluster, whereas even very complex data analysis could still be done sequentially on a single computer. An additional reason is cultural: researchers working on network models typically have a higher affinity to computer science problems than those working on experimental data. However, with the burgeoning amount of available data, parallelizing analysis is becoming less a matter of convenience and more one of necessity.

Acknowledgements Partially funded by DIP F1.2, Helmholtz Alliance on Systems Biology (Germany), Next-Generation Supercomputer Project of MEXT (Japan), BMBF Grant 01GQ0420 to BCCN Freiburg, and EU Grant 15879 (FACETS).

References

Backofen R, Borrmann H-G, Deck W, Dedner A, De Raedt L, Desch K, Diesmann M, Geier M, Greiner A, Hess W, Honerkamp J, Jankowski S, Krossing I, Liehr A, Karwath A, Kloefkorn R, Pesche R, Potjans T, Roettger M, Schmiedt-Thieme L, Schneider G, Voss B, Wiebelt B, Wienemann P, Winterer V-H (2006) A bottom–up approach to grid-computing at a university: the black-forest-grid initiative. PIK 29(2):81–87

Bader DA (2008) Petascale computing: algorithms and applications. Chapman & Hall/CRC, Boca Raton

Butenhof DR (1997) Programming with POSIX threads. Addison-Wesley, Boston

Czarnecki K, Eisenecker U (2000) Generative programming: methods, tools, and applications. Addison-Wesley, Reading (illustrated ed)

Fliegner D, Grün S, Messer P, Diesmann M, Geisel M (2003). Distributed computing for neuroscience-data analysis. In: Proc. 5th Meeting German Neuroscience Society, pp 659–660

Foster I (1995) Designing and building parallel programs. Addison-Wesley, Reading

Foster I, Kesselman C (2004). The Grid: Blueprint for a New Computing Infrastructure, 2nd edn. Morgan-Kaufman, San Francisco

Gerstein G, Bloom MJ, Espinosa IE, Evanczuk S (1983) Design of a laboratory for multiunit studies. IEEE Trans Sys M Cybern 5:668–676

Kötter R (ed) (2009) Frontiers in Neuroscience special issue: Python in Neuroscience

Langtangen HP (2006) Python scripting for computational neuroscience, 2nd edn. Springer, Berlin

Miller R, Boxer L (2005) Algorithms sequential and parallel: a unified approach, 2nd edn. Charles River Media, Hingham

Morrison A, Mehring C, Geisel T, Aertsen A, Diesmann M (2005) Advancing the boundaries of high connectivity network simulation with distributed computing. Neural Comput 17(8):1776–1801

Newham C, Rosenblatt B (1995) Learning the bash shell, 1st edn. O'Reilly & Associates, Inc., Sebastopol

Oram A, Talbott S (1991) Managing projects with make, 2nd edn. O'Reilly & Associates, Inc., Sebastopol

Wilkinson B, Allen M (2004) Parallel programming: techniques and applications using networked workstations and parallel computers, 2nd edn. Prentice Hall, New York

ERRATUM

Erratum to: Population Coding

**Stefano Panzeri, Fernando Montani,
Giuseppe Notaro, Cesare Magri,
and Rasmus S. Petersen**

**Erratum to: S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains*,
Springer Series in Computational Neuroscience 7, pp. 303–319
DOI 10.1007/978-1-4419-5675-0_14,**
© Springer Science+Business Media, LLC 2010

An error in production led to the misspelling of author Rasmus S. Petersen's name. The author's name is correctly spelled here. The Publisher regrets the error.

The online version of the original chapter can be found under
doi: 10.1007/978-1-4419-5675-0_14.

S. Panzeri (✉)
Department of Robotics, Brain and Cognitive Sciences, Italian Institute of Technology, Via Morego 30, 16163 Genoa, Italy
e-mail: stefano.panzeri@iit.it
url: http://www.iit.it

Index

S. Grün, S. Rotter (eds.), *Analysis of Parallel Spike Trains,*
Springer Series in Computational Neuroscience 7,
DOI 10.1007/978-1-4419-5675-0, © Springer Science+Business Media, LLC 2010