International Series in
Operations Research & Management Science

Jack P.C. Kleijnen

Design and Analysis of Simulation Experiments

Second Edition

Operations Research
Management Science

Springer

International Series in Operations Research & Management Science

Volume 230

More information about this series at http://www.springer.com/series/6161

Jack P.C. Kleijnen

Design and Analysis of Simulation Experiments

Second Edition

Springer

Jack P.C. Kleijnen
Department of Management
Tilburg University
Tilburg, The Netherlands

ISSN 0884-8289　　ISSN 2214-7934 (electronic)
International Series in Operations Research & Management Science
ISBN 978-3-319-18086-1　　ISBN 978-3-319-18087-8 (eBook)
DOI 10.1007/978-3-319-18087-8

Library of Congress Control Number: 2015937759

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media
(www.springer.com)

I dedicate this book to my wife, Wilma

Preface

This book is the second version of *Design and Analysis of Simulation Experiments*, originally published in 2008. Compared with the first edition, I have made many changes; I think that only a few sentences remain unchanged. Altogether, the new version has approximately 50% more pages than the original version. I have also changed the organization of the book; i.e., I have changed the order of the various chapters. More specifically, I have moved the chapter called "Screening Designs" from the end of the first edition, so in the second edition this chapter immediately follows the two chapters on classic designs (because both screening designs and classic designs assume polynomial regression metamodels). I have also reversed the order of the two chapters called "Simulation Optimization" and "Kriging Metamodels." Now the Kriging chapter immediately follows the chapters on classic designs and screening designs (Kriging assumes a metamodel involving a Gaussian process). The optimization chapter uses either regression models or Kriging models, which are now presented in the preceding chapters. The chapters on Kriging and optimization show most changes compared with the first edition; Kriging and simulation optimization are very active fields of research. I moved the section on validation (including R^2 and cross-validation) from Chap. 2 (which assumes "white noise") to Chap. 3. To the chapter on screening, I added a section on selecting the number of replications in sequential bifurcation (SB) through Wald's sequential probability ratio test (SPRT) and a section on SB for multiple types of simulation responses. I deleted Chap. 7, which was the last chapter called Epilogue. Note that in 2010 the first edition was also translated into Chinese (Beijing: Publishing House of Electronics Industry).

In the new version, I am no longer referring to a specific publication through a number, but through the name(s) of the author(s) plus the year of publication; the latter notation is more informative. Furthermore, I have tried to keep the list of references relatively short, so I exclude older references that are listed in newer references—unless I consider the older reference to be a "classic" publication. Nevertheless, this edition contains many references. Instead of a single list of references at the end of the book, I now present a list of references at the end of each chapter so that chapters may be downloaded separately. To improve the book's readability, I list many references at the very end of a paragraph or in a separate paragraph that starts with "*Note*".

In this version, I still focus on those aspects of simulation in which I have a certain expertise. This expertise is based on more than 40 years of research in the simulation method and its application in various areas. Although most of this expertise concerns discrete-event simulations (especially queueing and inventory simulations), I do have some experience with deterministic simulation (especially engineering simulations). Furthermore, this expertise is based on a doctoral degree in business and economics—in the German, not the Anglo-Saxon tradition—specializing in mathematical methods; altogether, I am an "operations researcher", but there are different types of operations researchers.

Like the first edition, the second edition requires that the readers already have a basic knowledge of the simulation method; e.g., they know concepts such as terminating simulation and steady-state simulation. They should also have a basic understanding of mathematical statistics, including concepts such as distribution functions, averages, and variances.

Information that I consider to be redundant is displayed between parentheses; nonredundant, extra information may be placed between em dashes (or —). Abbreviations and symbols are displayed in italics. Definitions of abbreviations and symbols are repeated in various chapters, and abbreviations can also be looked up in the Subject Index at the end of the book; this redundancy enables readers to browse through the various chapters, without having to follow a particular order. I do not use any footnotes; instead some paragraphs start with the word "*Note*". To avoid misleading hyphenation of website addresses, I display each address on a separate line; a comma or a period at the end of the address is not part of the address. Sometimes I treat non-English names in a sloppy way; e.g., I write the Russian name Sobol' as Sobol, and I always write Van Beers, whereas proper Dutch sometimes requires "van Beers" and proper Dutch lists "van Beers" in the References under the letter b instead of v. I write Gaussian (not gaussian), Kriging, and Studentizing, because these words are derived from the proper names Gauss, Krige, and Student (Gosset's pseudonym). I use American English (but my native tongue is Dutch), which avoids hyphens in compounds if there is no compelling reason; e.g., I write "nonnegative" and "re-estimate".

For an update of this book, including corrections and new references, visit my website

https://sites.google.com/site/kleijnenjackpc/

I thank Fred Hillier (Stanford University), who is the former series editor, for encouraging me to write a second version of my book. Mirko Janc (INFORMS) provided numerous technical comments on a preliminary version. Wim Van Beers (University of Amsterdam) read preliminary versions of all the chapters in this book and provided me with comments and with new versions of most Figures. Ehsan Mehdad (Tilburg University) read the preliminary versions of the chapters on Kriging and optimization and provided me with some new Figures for these chapters. I also received valuable comments on preliminary versions of various chapters from the following colleagues: Bertrand Iooss (Electricité de France R & D), Tom Lucas (Naval Postgraduate School), Barry Nelson (Northwestern University), Andrea Saltelli (Joint Research Centre of the European Commission), Lee Schruben (University of California Berkeley), Wen Shi (Huazhong University of Science and Technology), and Felipe Viana (GE Global Research).

I wrote this new edition, while being an emeritus professor at Tilburg University. The university provided me with an office, a PC with appropriate software, e-mail, and library services.

Furthermore, I reproduce the following text from the back cover of the original edition:

"This is an advanced expository book on statistical methods for the *Design and Analysis of Simulation Experiments* (DASE). Though the book focuses on DASE for *discrete-event* simulation (such as queueing and inventory simulations), it also discusses DASE for *deterministic* simulation (such as engineering and physics simulations). The text presents both classic and modern statistical designs. *Classic designs* (e.g., fractional factorials) assume only a few factors with a few values per factor. The resulting input/output data of the simulation experiment are analyzed through low-order polynomials, which are linear regression (meta)models. *Modern designs* allow many more factors, possible with many values per factor. These designs include group screening (e.g., Sequential Bifurcation, SB) and space filling designs (e.g., Latin Hypercube Sampling, LHS). The data resulting from these modern designs may be analyzed through low-order polynomials for group screening, and various metamodel types (e.g., Kriging) for LHS.

In this way, the book provides relatively *simple* solutions for the problem of which scenarios to simulate and how to analyze the resulting data. The book also includes methods for computationally *expensive* simulations.

It discusses only those *tactical* issues that are closely related to strategic issues; i.e., the text briefly discusses run-length and variance reduction techniques.

The leading textbooks on discrete-event simulation pay little attention to the *strategic* issues of simulation. The author has been working on strategic issues for approximately 40 years, in various scientific disciples [the original text has a typo: "disciples" should be "disciplines"]—such as operations research, management science, industrial engineering, mathematical statistics, economics, nuclear engineering, computer science, and information systems.

The intended *audience* are researchers, graduate students, and mature practitioners in the simulation area. They are assumed to have a *basic* knowledge of simulation and mathematical statistics; nevertheless, the book summarizes these basics, for the readers' convenience."

Finally, I reproduce the following text from the Preface of the original version:

"I received valuable comments on preliminary versions of various chapters from the following colleagues: Ebru Angün (Galatasaray University, Istanbul), Russell Barton (Pennsylvania State), Victoria Chen (University of Texas at Arlington), Gabriella Dellino (Politecnico di Bari), Dick den Hertog (Tilburg University), Tony Giunta (Sandia), Yao Lin (Georgia Institute of Technology), Carlo Meloni (Politecnico di Bari), Barry Nelson (Northwestern), William Notz (Ohio State), Huda Abdullah Rasheed (al-Mustansiriyah University, Baghdad), Wim van Beers (Tilburg University), Willem van Groenendaal (Tilburg University), Jim Wilson (North Carolina State), and Bernard Zeigler (Arizona State)."

This book is summarized in Kleijnen (2015).

Reference

Kleijnen JPC (2015) Regression and Kriging metamodels with their experimental designs in simulation: review. CentER Discussion Paper 2015–035 (http://ssrn.com/abstract=2627131)

Contents

About the Author

Jack P.C. KLEIJNEN is Emeritus Professor of Simulation and Information Systems at Tilburg University, where he is still an active member of both the Department of Management and the Operations Research Group of the Center for Economic Research (CentER) in the Tilburg School of Economics and Management. His research concerns the statistical design and analysis of experiments with simulation models in many scientific disciplines including management science, operations research, and engineering. He was a consultant for several organizations in the USA and Europe, and served on many international editorial boards and scientific committees. He also spent several years in the USA, at universities and private companies. He received a number of national and international awards e.g., in 2008 the Queen of the Netherlands appointed him a Knight in the Order of the Netherlands Lion and in 2005 the INFORMS Simulation Society awarded him the Lifetime Professional Achievement Award (LPAA). While being an Emeritus Professor, he wrote the second version of *Design and Analysis of Simulation Experiments*. His e-mail address is kleijnen@tilburguniversity.edu (if this address does not work, then kleijnen.jack@gmail.com), his publication list is available on

https://pure.uvt.nl/portal/en/persons/jack-pc-kleijnen(da721b00-b03f-4c42-98fe-55e593f541a8)/publications.html

1
Introduction

This chapter is organized as follows. Section 1.1 defines various types of simulation. Section 1.2 defines *design and analysis of simulation experiments* (DASE). Section 1.3 defines DASE symbols and terms. The chapter ends with Solutions of exercises, and references.

1.1 What Is Simulation?

"Simulation" might be defined in several ways, so it includes (say) the simulation of an illness. However, we limit simulation to experimenting with quantitative models; obviously, these models are computerized nowadays. To define this type of simulation, we use the following two dichotomies:

- Deterministic versus random
- Static versus dynamic

Unlike deterministic models, random models include random or probabilistic variables. Unlike static models, dynamic models include time as a special independent variable. These two dichotomies may be combined; simple examples are:

- Deterministic and static model: a first-order polynomial with x as the independent variable and y as the dependent variable.

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_1

- Random and static model: the probability of heads or tails in the toss of a coin.

- Deterministic and dynamic model: a differential equation with time (say) t as the independent variable; e.g., the *net present value* (NPV) of a loan (also see Example 1.1 below).

- Random and dynamic model: a model of the waiting times in a single-server queueing model (also see Example 1.2).

Definition 1.1 *A simulation model is a mathematical model that is solved by means of experimentation.*

So, by definition, we ignore simulation models that are physical (instead of mathematical); e.g., a miniature airplane in a windtunnel. Mathematical models are usually converted into computer programs—also called computer codes—so simulation experiments are also called *computer experiments*. Closely related to simulation are *Monte Carlo* methods, defined as methods that use *pseudorandom numbers* (PRNs). These PRNs are generated by means of a computer program, so they are not really random, and yet they are assumed to be independently and uniformly distributed on the interval $[0, 1]$. So, Monte Carlo methods involve chance, which explains the name. Monte Carlo methods are also used to evaluate multiple integrals, which arise in mathematical statistics, physics, etc. Simulation uses experimentation to solve the mathematical model; i.e., simulation is a *numerical* method, not an analytical method. Simulation is applied in many scientific disciplines—ranging from sociology to astronomy; see the survey on the spectrum of simulation applications in the classic article Karplus (1983).

Simulation methodology is explained in many textbooks, in many scientific disciplines. Simulation methodology includes DASE, which is also known as *design of computer experiments* or *DACE*. As we mentioned in the Preface, this book on DASE is oriented towards *management science/operations research* (MS/OR). MS/OR is a discipline that includes simulation, especially random and dynamic simulation—also known as *discrete-event simulation* of *discrete-event dynamic systems*(DEDS). The most popular and most recent simulation textbook in MS/OR is Law (2015); DEDS is discussed in the classic textbook Ho and Cao (1991). A classic textbook on the theory of modeling and simulation is Zeigler et al. (2000), using the automata theory of computer science; that book influences some of the terminology in this book.

Example 1.1 *Consider the following NPV problem. Given are θ, the discount factor used by the decision maker; n, the length of the planning period measured in years; and x_t, the cash flow in year t with $t = 0, \ldots, n$. Then the NPV—also called the Present Value (PV)—(say) y may be computed through the following equation:*

$$y = \sum_{t=0}^{n} \frac{x_t}{(1+\theta)^t}. \tag{1.1}$$

Engineers often use an alternative formula, assuming continuous time so $\sum$ *becomes* $\int$*, etc. Eq. (1.1) may be used to compare alternative cash flow patterns. Different patterns may be caused by different loan types. One loan type may require a fixed amount paid back at the end of each year (say)* z_t *with* $t = 1, \ldots, n$ *and* $z_0 = 0$*, and interest payments determined by the interest rate* c *and the loan amount at the end of the year* t*, namely* w_t*:*

$$x_t = -[min(z_t, w_t) + c\,w_t] \quad with\ t = 1, \ldots, n \tag{1.2}$$

where the loan amount is determined by

$$w_t = w_{t-1} - z_t \quad with\ t = 1, \ldots, n \tag{1.3}$$

and

$$x_0 = w_0 \tag{1.4}$$

where w_0 *is the original loan amount, so* x_0 *is the positive cash flow at the start of the planning period, whereas* x_t *with* $t = 1, \ldots, n$ *are negative cash flows (the initial condition* $z_0 = 0$ *has already been specified). Finally, the stopping conditions of the simulation run must also be given; in this example, the simulation stops when the end of the planning period is reached.*

Obviously, Example 1.1 illustrates a deterministic dynamic model, including a first-order difference equation; namely, Eq. (1.3). Easy programming of such models is possible through *spreadsheet* software such as Excel; a recent reference on spreadsheet-based simulation is Schriber (2009).

Exercise 1.1 *Derive that* $NPV = 6.238$ *in case the original loan is* $w_0 = 100$*,* $n = 2$*,* $c = 0.10$*, and* $\theta = 0.15$ *(i.e., the loaner expects to earn a higher return on investment or ROI than the bank can offer).*

The *deterministic* financial simulation in Example 1.1 may be augmented to a *random* simulation, if (say) the discount factor θ or the cash flows x_t are unknown so their values are sampled from distribution functions. This type of simulation is called *risk analysis* (RA) or *uncertainty analysis* (UA); see again Schriber (2009). Random simulation is more complicated than deterministic simulation is, so we recommend random simulation only if a random model is necessary to obtain a valid representation of the real system so that the model serves the goals that will be discussed in Sect. 1.2.

Note: RA in chemical engineering is discussed in Saltelli et al. (2005). Some well-known textbooks on RA are Evans and Olson (1998) and Vose (2000); a recent survey article is Wu and Olson (2013). Combining risk management and robust design is discussed in Mordecai and Dori (2013). We discuss DASE aspects of RA in Sect. 5.9 and DASE aspects of robust design and robust optimization in Sect. 6.4.

Complicated realistic examples of deterministic simulation are found in *computer aided engineering* (CAE) and *computer aided design* (CAD), including models of airplanes, automobiles, television sets, chemical processes, and computer chips—developed at Boeing, General Motors, Philips, etc. Many applications use finite-elements analysis. The role of simulation in engineering is discussed by the Blue Ribbon Panel of the American "National Science Foundation (NSF)"; and reported in Oden (2006).

Another type of (primarily) deterministic simulation is *system dynamics* (SD), originally called "industrial dynamics" in Forrester (1961). SD is more than a simulation method; it is a world view. In this view, a crucial concept is *feedback*; i.e., compare an output with a norm, and react if there is an undesirable deviation. Simulation results show that this feedback often generates counterintuitive behavior. Applications include simulations of companies, industries (including supply chains), countries, and the whole globe (including the warming-up of the earth's atmosphere). A textbook with more than 1,000 pages is Sterman (2000).

Some deterministic simulation models show numerical inaccuracies, which make these models related to random simulation. These deterministic simulations are also called "noisy computer experiments" or "stochastic simulators"; see Picheny et al. (2013). We distinguish the following three types of random simulation.

- The simulation model is deterministic, but it has *numerical* noise caused by numerical approximations; see again Picheny et al. (2013) and also Forrester et al. (2008, p. 141) and Wiebenga (2014).

- The simulation model is deterministic, but the exact values of its inputs are uncertain so these values are sampled from a prior input distribution through Monte Carlo methods (e.g., Latin hypercube sampling, discussed in Sect. 5.5). This is done in RA, and is also known as *uncertainty propagation*. This uncertainty is called epistemic, subjective, or the analysts' uncertainty; see Helton and Davis (2003).

- The simulation model itself includes PRNs; examples are discrete-event simulation models, including queueing in traffic systems, telecommunications, and supply chains. These PRNs may be used to sample the occurrence of events such as the arrival of cars, telephone calls, and production orders. The times at which these events occur may be sampled from a given distribution; e.g. an exponential distribution. This sampling creates so-called aleatory, objective, or the system's inherent uncertainty; see again Helton and Davis (2003).

Only a few publications combine epistemic and aleatory uncertainties. For example, Helton and Davis (2003) discusses the simulation model of the "waste isolation pilot plant (WIPP)" that combines (i) deterministic simulation through differential equations that model chemical and physical subsystems and (ii) discrete-event simulation that models human interventions. Another example is discrete-event simulation with uncertain parameters; e.g., the parameter of the arrival distribution in the queueing simulation is uncertain. Combining aleatory and epistemic uncertainties is further discussed in Borgonovo and Plischke (2015), De Rocquigny et al. (2008), Helton et al. (2014), Kleijnen (2007), Sakallı and Baykoç (2011), and Xie et al. (2014). Besides epistemic uncertainty, Grubler et al. (2015) discusses more types of uncertainty.

The preceding discussion implies the following two definitions, based on Zeigler et al. (2000).

Definition 1.2 *A model parameter has a value that is inferred from data on the real system.*

This inference is necessary if the parameter value can not be observed directly in the real system. An example is the arrival rate of customers into a supermarket; i.e., we can observe the times between two successive arrivals, and use these observations to estimate the arrival rate.

Definition 1.3 *An input variable of a model can be directly observed in the real system.*

Returning to the supermarket example, we can simply observe the number of servers (checkout lanes).

Exercise 1.2 *Consider the following two applications involving the discount factor for a NPV calculation as in Example 1.1: (a) a student wishes to select the best NPV for several loan alternatives—each with the same interest rate, but with different amortization schemes; (b) a company wishes to select the highest NPV among several investment alternatives, such that the company maintains the ROI that it has realized during the last five years. Is the discount factor a parameter or a variable in (a) and (b)?*

Now we focus on *discrete-event* simulation. This simulation is inherently random; i.e., without randomness the problem would change completely. For example, a queueing problem is caused by the randomness of the arrival or the service times; if these times were deterministic, the problem would become a so-called scheduling problem. A popular discrete-event simulation—which may be a building block for more complicated simulations, and which is often used in this book and in other publications—is the M/M/1 model (the symbol M/M/1 is used in the so-called Kendall notation).

Definition 1.4 *An M/M/1 model is a queueing model with one server, and Markovian interarrival and service times.*

These Markovian times are exponentially distributed and “independent”; i.e., the interarrival times are independent, and so are the service times; arrival and service times are mutually independent (also see Example 1.2

below). The exponential distribution has the memoryless property; e.g., if many customers happened to arrive during last period, then this does not affect the number of customers in the next period. Furthermore, the exponential distribution implies that the number of events (e.g., arrivals) per period (e.g., per day) has a Poisson distribution. The notation M/M/1 implies that the server's priority rule is first-in-first-out (FIFO), the waiting room has infinite capacity, etc. An M/M/1 model may be simulated as follows.

Example 1.2 *Let a_{i+1} denote the interarrival time between customers i and $i+1$, s_i the service time of customer i, and r a PRN. Assume that the output of interest is w, the waiting time of a customer, and that the probability density function (PDF) of this random output is characterized by its mean that is estimated through*

$$\overline{w} = \frac{\sum_{i=1}^{n} w_i}{n} \tag{1.5}$$

where n denotes the number of customers that stops the simulation run. (This example is a terminating simulation, not a steady-state simulation; in the latter case, n would not be prefixed or would be a "very large" number; see Law (2015).) Furthermore, assume that the simulation starts in the "empty" state (no customers in the system), so the customer who arrives first does not need to wait; i.e., the initial condition of this dynamic model is $w_1 = 0$. The dynamics of the single-server system are specified by the so-called Lindley recurrence formula

$$w_{i+1} = max\ (0, w_i + s_i - a_{i+1}). \tag{1.6}$$

In this equation, the random input variables s and a are sampled such that these variables have a service rate μ and an arrival rate λ; so the mean or expected service and interarrival times are $1/\mu$ and $1/\lambda$, respectively. To sample these variables, the simulation may use the PRN r as follows:

$$s_i = \frac{-\ln r_{2i-1}}{\mu} \tag{1.7}$$

and

$$a_{i+1} = \frac{-\ln r_{2i}}{\lambda} \tag{1.8}$$

where a single PRN stream (namely, r_1, r_2, ...,r_{2n-1}, r_{2n}) is used; obviously, each of the n customers needs two PRNs—namely, one PRN for the arrival time and one PRN for the service time.

To program the simulation model in Example 1.2, the analysts can choose from many simulation software packages. In fact, Swain (2013) lists 43 products in the ninth biennial survey of simulation software for discrete-event simulation; that survey also includes information on DASE and so-called animation (kind of motion pictures).

Exercise 1.3 *Example 1.2 uses a single PRN stream (namely, r_1, r_2, ..., r_{2n-1}, r_{2n}) in Eqs. (1.7) and (1.8). What are the advantages of using two separate PRN streams for the two input processes—namely, the arrival and the service processes—when applying two well-known variance reduction techniques (VRTs)—namely, common random numbers (CRN) and antithetic random numbers (ARN)? Do these advantages change when the single-sever simulation has the last-in-first-out (LIFO) server priority rule or the service time has a uniform distribution?*

Mathematical analysis of the M/M/1 model reveals that the fundamental input parameter is the so-called *traffic rate*—also called traffic intensity or traffic load—(say) ρ defined as $\rho = \lambda/\mu$ with λ and μ defined above Eq. (1.7). In other words, the M/M/1 model has a single input parameter (namely, ρ), whereas its computer code has two parameters (λ and μ). More precisely, mathematical analysis gives the following equation for the expected value of the waiting time in the "steady-state" so Eq. (1.6) has $i \uparrow \infty$:

$$E(w_i \mid i \uparrow \infty) = \frac{\lambda}{\mu(\mu - \lambda)} = \frac{1}{\mu}\frac{\rho}{(1-\rho)}, \tag{1.9}$$

so selecting the time unit such that $\mu = 1$ (e.g. measure time in either seconds or hours) gives $E(w_i \mid i \uparrow \infty) = \rho/(1-\rho)$.

Though the M/M/1 model will often be used as an example in this book, we shall also need an example with multiple inputs. Therefore we now present another well-known building block for discrete-event simulation; namely, the so-called (s, S) model.

Definition 1.5 *An (s, S) model is a model of an inventory management system with the following properties. Its control variables s and S satisfy the condition $s < S$. One of the model inputs is (say) D, the random demand per period, so the inventory level I becomes $I - D$. This I is replenished whenever I decreases to a value smaller than or equal to the reorder level s. When I is replenished, the order quantity Q is $S - I$. Altogether the model implies*

$$Q = \begin{cases} S - I & \textit{if } I \leq s \\ 0 & \textit{if } I > s. \end{cases} \tag{1.10}$$

There are several *variations* on this basic model. For example, review of the inventory level I may be continuous instead of periodic (e.g., at the end of each day). The lead time of the order may be either a nonnegative constant or a nonnegative random variable. Demand that exceeds the inventory at hand (so $D > I$) may be either lost or backlogged. Costs may consist of inventory, ordering, and out-of-stock costs (including loss of goodwill and expediting costs). These cost components are specific mathematical functions; e.g., inventory carrying (or holding) cost may be a constant per item unit, per time unit. In practice, out-of-stock costs are hard to quantify so a service (or fill rate) constraint may be specified instead; e.g., the total stockout quantity per (say) year should be smaller than 10 % of the total sales during that same period.

Programming this inventory model is harder than programming the M/M/1 model; the latter has dynamics specified by the simple Eq. (1.6). Thorough discussions of this programming is found in simulation textbooks such as Law (2015).

Discrete-event simulation and continuous simulation may be combined into so-called *hybrid* simulation. An example is a simulation of the ejection of the pilot seat (a discrete event) during a flight that is modeled through differential equations until this event occurs. This type of simulation is also discussed in textbooks on discrete-event simulation; e.g. Law (2015). We also refer to Giambiasi and Carmona (2006).

In summary, simulation is a method that is applied in many disciplines. Simulation provides a flexible, powerful, and intuitive tool for the analysis of complicated processes. The resulting insight may be used to design better real systems.

Much more could be said about simulation. There are many more textbooks besides the ones mentioned above; e.g., Nelson (2013) and Pidd (2004); the latter textbook also discusses system dynamics. The most recent publications on discrete-event simulation can be found in the annual proceedings of the *Winter Simulation Conference*; see its web page

`http://www.wintersim.org/`.

Top journals on MS/OR including discrete-event simulation are published by INFORMS; see

`http://www.informs.org/`.

Handbooks in MS/OR also cover discrete-event simulation; an example is Henderson and Nelson (2006). Many other journals on MS/OR also publish on simulation. Perspectives on the evolution of discrete-event simulation during 50 years are provided in Nance and Sargent (2002).

Exercise 1.4 *Does the definition of "simulation" hold for (i) entertainment games such as "America's Army" (see Swain 2005), (ii) serious games such as the beer game in system dynamics (see Simchi-Levi et al. 2003), and (iii) game theory using the Nash equilibrium (see Shubik 2002)?*

1.2 What Is "Design and Analysis of Simulation Experiments" (DASE)?

This book is about the design and analysis of simulation experiments (DASE). These terms require explicit definitions—especially because simulation is a method applied in many different scientific fields with their own terminologies, as we saw above.

Simulation implies that the modelers do not solve their model through mathematical analysis; instead, the modelers try different values for the inputs and parameters of their model in order to learn what happens to the model's output. For example, for the NPV model in Example 1.1 the modelers may experiment with different values for the parameter θ (discount factor) and the input variable z (amount paid back every year); see again Eqs. (1.1) and (1.2). In the M/M/1 model in Example 1.2 the modelers may experiment with different values for the traffic rate and with different priority rules besides the implicit FIFO rule. In the (s, S) inventory model defined in Eq. (1.10) the modelers may try different combinations of the control limits s and S and the mean demand $E(D)$. The *goals* of such numerical experiments may be (see again Oden (2006), and also Kaminski (2015), Razavi and Gupta (2015), and Tan (2014)):

- Verification and validation (V & V) of the underlying simulation model
- Sensitivity analysis (SA)—either global or local—or "what if" analysis of the simulation model
- Optimization of the simulated real system (SimOpt)
- Risk analysis (RA) of the simulated real system

In practice, these goals may be ambiguous, and they may be known under other names. One example is SA, which may aim at either "gaining insight" or "prediction". Another example is RA, which may aim at estimating the set of input combinations that give an unacceptably high probability of exceeding a given threshold. Many methods for SA and RA are surveyed in Borgonovo and Plischke (2015); we shall detail specific SA and RA methods in the next chapters; see, e.g., Sect. 5.9.

These goals require that the simulation modelers pay attention to the *design* of their experiments; e.g., if the modelers keep an input of their simulation constant, then they cannot estimate the effect of that input on the output. In practice, however, many modelers keep many inputs constant, and experiment with a few remaining inputs only. Chapter 4 (on screening) shows that there are better ways to run simulation experiments with many inputs. Another example of bad practice is changing one input at a time, while keeping all other inputs fixed at their so-called base values; Chap. 2 shows that this approach is inefficient and does not enable the estimation of any interactions among inputs.

A main theme of this book is that the *design* of the experiment is intimately related to its *analysis*. For example, suppose that the modelers assume the input to have a "linear" effect on the output; i.e., they assume a first-order polynomial approximation (remember the Taylor series in mathematics) or main effects only (mathematical statistics terminology). Given this assumption, it obviously suffices to experiment with only two values of that input. Furthermore, if the modelers assume that there are (say) $k > 1$ inputs (with main effects only), then their design requires a relatively small experiment (of order k). In this example, changing only one input at a time does give unbiased estimators of all the first-order or main effects; Chap. 2, however, will show that minimizing the variances of these estimators requires a different design—with approximately the same size of the experiment as the one required by the one-at-a-time design. Note that this book uses a DASE approach that is frequentist, not Bayesian; Bayesian versus frequentist approaches are discussed in Efron (2005).

A first-order polynomial approximation is an example of a so-called metamodel, which is the term used in Kleijnen (1975); metamodels are also called response surfaces, surrogates, and emulators in different scientific disciplines.

Definition 1.6 *A metamodel is an approximation of the input/output (I/O) function that is defined by the underlying simulation model.*

We point out that a simulation model implicitly defines a mathematical function. There are different *types* of metamodels. The most popular type is a polynomial of either first order or second order (degree), which are discussed in Chaps. 2–4. A more recent metamodel type that is gaining popularity in simulation—especially deterministic simulation—is a Kriging model—also called a Gaussian process model—discussed in Chap. 5. Metamodels may be used for different goals; e.g., a low-order polynomial best serves explanation resulting in insight, whereas a Kriging model may give better predictions which may be used in optimization (see Chap. 6), real-time decision making, etc.

Note: Less popular metamodels are (in alphabetical order): classification and regression trees (CART), game-theoretic metamodels, generalized linear models (GLM), inverse distance weighting, multivariate adaptive regression splines (MARS), (artificial) neural networks, nonlinear regression models, nonparametric regression analysis, nonparametric uncertainty analysis (NPUA), radial basic functions (RBFs), rational functions, splines, stochastic polynomial interpolation (or polynomial chaos expansion), support vector regression (SVR), symbolic regression, wavelets, etc. For these alternative metamodels, Kleijnen (2008, p. 8) gives twenty-two references. Additional references are Poropudas and Virtanen (2008) for game-theoretic models, Shepard (1968) for inverse distance weighting, Dette and Pepelyshev (2010) for NPUA, Santos and Santos (2008) for nonlinear regression models, Regis (2014) for RBFs, Tan (2014) for stochastic polynomial interpolation, and Clarke et al. (2005), Rasmussen and Williams (2006, pp. 141–146), and Rieck et al. (2012) for SVR. Various metamodels are compared in Razavi et al. (2012), Can and Heavey (2012), Forrester and Keane (2009), Levy and Steinberg (2010), Storlie et al. (2009), Van Gelder et al. (2014), Viana et al. (2014), Villa-Vialaneix et al. (2012), Wang et al. (2014), and Zhu et al. (2011).

In theory, modelers may combine several types of metamodels, weighing each type with its estimated accuracy. In practice, however, such a combination is rare, because modelers are familiar with one or two types only.

Note: Combining metamodels into a so-called ensemble is further discussed in Acar and Rais-Rohani (2009), Gorissen (2010, Chapter 7), Müller and Shoemaker (2014), and Viana et al. (2014, Section IV). Furthermore, Buchholz et al. (2008) discusses the combination of several regression models, each with a different subset of inputs. Harari and Steinberg (2014) discusses the combination of several Kriging models, each with its own correlation function.

The term “response surface” is used for *local* metamodels in *response surface methodology* (RSM); the same term is used for *global* metamodels in deterministic simulation. Such a local model implies that only a small subarea of the total experimental area is considered. The limit of this “small” subarea is an area with a size that tends to zero, so partial derivatives are legitimately considered. These derivatives are the components of the gradient, which will be further discussed in Sect. 6.2 on RSM for the optimization of real or simulated systems.

The *experimental area* is called the *experimental frame* in Zeigler et al. (2000). We could also call it the “domain of admissible scenarios”, given the goals of the simulation study.

We propose the following algorithm for DASE.

Algorithm 1.1

1. Select a tentative metamodel.
2. Select a design that enables the estimation of the parameters of the selected metamodel, followed by the validation of this tentative estimated metamodel.
3. If this metamodel is rejected because this model seems not to be valid, then select a different metamodel and return to step 1; else proceed to the next step.
4. Apply the validated metamodel for one or more goals mentioned above; namely, V & V, SA, SimOpt, or RA.

Steps 1 and 2 imply that specification of the metamodel precedes selection of the design. Step 3 implies that the specified tentative metamodel of Step 1 can be rejected (so the strategy agrees with Popper’s “falsification” principle). Details of this algorithm will be given in the next chapters, assuming metamodels that are either “low order” polynomials—namely, first-order and second-order polynomials—or Kriging models.

DASE has both *strategic* and *tactical* aspects. Traditionally, researchers in discrete-event simulation have focused on tactical issues, such as the runlength of a steady-state simulation, the number of runs of a terminating simulation, and VRTs; see the classic article Conway (1963) and the more recent literature mentioned above. In deterministic simulation these tactical issues vanish, so statisticians have been attracted to strategic issues; namely, which input combinations to simulate and how to analyze the resulting output; see the textbooks Fang et al. (2006) and Santner et al. (2003). Few statisticians have studied random simulation. Few simulation analysts have focused on strategic issues. In this book, we focus on strategic issues, discussing only those tactical issues that are closely related to strategic issues; e.g., the consequences of applying CRN.

The statistical theory on *design of experiments* (DOE or DoE) was developed for real, nonsimulated experiments in agriculture in the 1920s, and has been developed in engineering, psychology, etc. since the 1950s. In these real experiments it is impractical to investigate "many" factors; ten factors seems a maximum. Moreover, it is then hard to experiment with factors that have more than "a few" values; five values per factor seems the limit. In simulated experiments, however, these restrictions do not apply. Indeed, simulation models may have thousands of factors—each with many values. Consequently, a multitude of factor combinations may be simulated. Moreover, simulation is well-suited to "sequential" designs instead of "one shot" designs, because simulation experiments are run on computers that typically produce output sequentially (apart from parallel computers, which are used only in specific application areas such as military applications and energy exploration), whereas agricultural experiments are run during a single growing season. So a change of mindset of simulation experimenters is necessary. A more detailed discussion of simulated versus real experiments is Sanchez et al. (2012).

In summary, DASE is needed to improve the efficiency and effectiveness of simulation; i.e., DASE is crucial in the overall process of simulation.

1.3 DASE Symbols and Terminology

Some DASE symbols and terms should be explicitly defined, because DASE is a combination of mathematical statistics and linear algebra that is applied to experiments with deterministic and random simulation models; these models are applied in different scientific disciplines.

Deciding on the DASE *symbols* is problematic; e.g., mathematicians use capital letters to denote matrices, whereas statisticians use capitals to denote random variables. Consistency would require denoting the error term in a regression model by (say) E and the matrix of explanatory variables by $\mathbf{x}$. Such a notation, however, would seem too orthodox. Most authors in simulation and regression analysis do not always use capitals for random variables; the readers should infer from the context whether a variable is random or not. Bold letters denote matrices and vectors. Whenever readers might be misled, we explicitly discuss the randomness of a particular variable. For example, Chap. 3 covers "generalized least squares (GLS)", which uses the covariance matrix of the simulation responses; in practice this matrix is estimated, which creates statistical problems that need explicit discussion.

Greek letters denote *parameters*; parameters were introduced in Definition 1.2. For example, the service rate μ in the M/M/1 model is estimated from the (say) n observations on the service time s so $\widehat{\mu} = 1/\overline{s}$ with $\overline{s} = \sum_{i=1}^{n} s_i/n$. An "estimator" (e.g., the sample average) is a random variable; the estimator has a specific value called an "estimate".

Unlike a parameter, a *variable* can be directly observed in the real world. For example, the input variable service time s can be measured in a straightforward way; we could say that s is the realization of the random variable S. A variable may be either an input or an output of a model; e.g., the M/M/1 model may have the input s and the output w, denoting waiting time.

Both parameters and input variables may be changed in a simulation experiment; in that case they have at least two *values* or *levels* in the experiment. Parameters and input variables together are called *factors,* in DOE. For example, a simple design in DOE is a 2^k factorial experiment; i.e., there are k factors, each with two levels; all their combinations are simulated. These combinations are often called *scenarios* in simulation and modeling. Scenarios are usually called *design points* or *runs* by statisticians, but we reserve the term "run" for a *simulation run*; a simulation run starts in the initial condition (e.g., the empty state in an M/M/1 simulation) and ends once a specific event occurs (e.g., n customers have been simulated; see the discussion below Eq. (1.5)).

Factors and responses (outputs) may be either *qualitative* or *quantitative.* In the M/M/1 example, quantitative factors are the arrival and service rates; the traffic rate is the fundamental quantitative factor. In a single-server queueing simulation, a qualitative factor may be the priority rule—which may have (say) three levels, namely FIFO, LIFO, or "shortest-processing time first" (SPT).

Simulation inputs and outputs may be measured on the following five types of *scales*:

1. *Nominal*: This is the only scale that applies to a qualitative (or categorical) factor. One example was the priority rule with its three nominal values (FIFO, LIFO, SPT). Another example is a simulation with two types of customers, namely A (emergencies) and B (regular). Interpolation or extrapolation makes no sense (so regression analysis must be applied with care; see Chap. 2).

2. *Ordinal*: This scale ranks the values of the input or output. For example, this scale sorts (say) n observed output values from lowest to highest, and assigns them ranks from 1 to n. *Order statistics* uses

such a scale; see the textbooks on nonparametric (distribution-free) statistics, Conover (1999) and the more recent Sprent and Smeeton (2007); order statistics will be used in later chapters. Another example is a survey that assigns ranks from 1 to 5 in order to measure how strongly the respondent agrees with a statement; namely, completely agree, agree, neutral, disagree, and strongly disagree.

3. *Interval*: This scale assigns numbers that are unique except for a linear transformation; i.e., this scale has an arbitrary zero point. An example is temperature measured in Celsius or Fahrenheit degrees. Analysts should prefer mathematical and statistical methods that are not sensitive to the scale that is used to quantify inputs or outputs. For example, Sect. 6.2.3 covers a scale-independent alternative for the steepest ascent method; the latter method is standard in RSM.

4. *Ratio*: This scale has a unique zero, so "$2x$" means "twice as much as x". Examples are length measured in centimeters or inches, and cash flow measured in euros or US dollars. Other examples are the arrival and the service rates, which depend on the time unit (e.g., seconds). Like the interval scale, the ratio scale should not change "the" conclusions of mathematical and statistical analyses.

5. *Absolute*: No transformation applies. An example is the number of customers arriving during the simulation run of an M/M/1 model; this is a discrete (not a continuous) variable.

A more detailed discussion of types of variables and measurement scales is given in Kleijnen (1987, pp. 135–142).

Exercise 1.5 *Mathematical statistics often uses Student's t-statistic. This statistic has several forms, but the simplest and best-known form is*

$$t_{m-1} = \frac{\overline{x} - \mu_x}{s_{\overline{x}}}$$

with

$$\overline{x} = \frac{\sum_{r=1}^{m} x_r}{m}$$

where $x_r \sim NIID(\mu_x, \sigma_x^2)$ with NIID standing for "normally, independently, and identically distributed", $\mu_x = E(x)$, and $\sigma_x^2 = Var(x)$; furthermore

$$s_{\overline{x}} = \frac{s_x}{\sqrt{m}}$$

with

$$s_x = \sqrt{\frac{\sum_{r=1}^{m}(x_r - \overline{x})^2}{m-1}}.$$

Obviously, the m *outputs of a simulation model with constant parameters (e.g., an M/M/1 simulation model with a constant traffic rate) using nonoverlapping PRN streams are IID. These outputs are normally (Gaussian) distributed if the output is (e.g.) the average waiting time (even though the individual waiting times are autocorrelated; see the "functional central limit theorem" in Sect. 3.3). The null-hypothesis may be that* μ_x *is given by the steady-state formula for the M/M/1 queueing system given in Eq. (1.9). This hypothesis is rejected if the* $1-\alpha$ *confidence interval* $\overline{x} \pm t_{m-1;1-\alpha/2}$ *does not cover the hypothesized value. Run your experiment (say) 100 times; i.e., generate 100 macroreplications with nonoverlapping PRNs and* $\alpha = 0.10$*; check whether you indeed reject the null-hypothesis in approximately* 10 $(= 100 \times 0.10)$ *macroreplications.*

Exercise 1.6 *Because "simulation" involves experimenting with a computer model, you should program the M/M/1 defined in Example 1.2 using any software you like (e.g., Arena or C++). Select your "favorite" performance measure; e.g., average waiting time. Next you should experiment with your simulation model; some suggestions follow.*

1. *Change the run-length (symbol* n *in Example 1.2) from (say)* $n = 10$ *(terminating simulation) to* n *large enough to reach the steady state; try these two* n *values for a "low" and a "high" traffic rate. Run "several" macroreplications; e.g.,* $m = 10$ *replications. Ensure that these replications are identically and independently distributed (IID); i.e., use nonoverlapping PRN streams. Use either a single PRN stream for service and arrival times or use two separate streams for the arrival and service times, respectively. Compare your simulation estimate with the analytical steady-state mean; use graphical plots and mathematical statistics such as discussed in Exercise 1.5.*

2. *To estimate the I/O function, change the traffic load* $(\rho = \lambda/\mu)$*. Apply either the same or different PRN seeds when comparing traffic loads: do CRN give better results?*

3. *Replace the exponential distribution for service times by a different distribution; e.g., a uniform distribution with the same mean, keeping the traffic load constant when changing the distribution. Select some fixed value for the traffic rate, the number of customers per run, and the number of macroreplications, respectively; e.g., select one of the values used above. Does the change in distributions change the selected performance measure significantly?*

Solutions of Exercises

Solution 1.1

t	*payback*	*interest*	*NPV*
0	*100*	*0*	*100*
1	*−50*	*−10*	$\frac{-(50+10)}{1+0.15} = -52.174$
2	*−50*	*−5*	$\frac{-(50+5)}{(1+0.15)^2} = -41.588$
			$100 - 52.174 - 41.588 = 6.238$

Solution 1.2 *(a) For the student the discount factor is a variable, quoted by the bank; (b) for the company it is a parameter to be estimated from its investments during the last five years.*

Solution 1.3 *Separate PRN streams improve the performance of CRN and ARN; see any textbook on discrete-event simulation. This improvement also holds for LIFO or uniformly distributed service times.*

Solution 1.4 *Both entertainment games and serious games are simulation models; gaming theory uses analytical solutions so it is no simulation.*

Solution 1.5 *Program and run your Monte Carlo experiment.*

Solution 1.6 *Many answers are possible; compare your results with the results that you will obtain, once you will have read some of the next chapters.*

References

Acar E, Rais-Rohani M (2009) Ensemble of metamodels with optimized weight factors. Struct Multidiscip Optim 37(3):279–294

Buchholz A, Holländer N, Sauerbrei W (2008) On properties of predictors derived with a two-step bootstrap model averaging approach—a simulation study in the linear regression model. Comput Stat Data Anal 52:2778–2793

Borgonovo E, Plischke E (2015) Sensitivity analysis: a review of recent advances. Eur J Oper Res (in press)

Can B, Heavey C (2012) A comparison of genetic programming and artificial neural networks in metamodeling of discrete-event simulation models. Comput Oper Res 39(2):424–436

Clarke SM, Griebsch JH, Simpson TW (2005) Analysis of support vector regression for approximation of complex engineering analyses. ASME J Mech Des 127(6):1077–1087

Conover WJ (1999) Practical nonparametric statistics, 3rd edn. Wiley, New York

Conway RW (1963) Some tactical problems in digital simulation. Manag Sci 10(1):47–61

De Rocquigny E, Devictor N, Tarantola S (2008) Uncertainty settings and natures of uncertainty. In: de Rocquigny E, Devictor N (eds) Tarantola suncertainty in industrial practice. Wiley, Chichester

Dette H, Pepelyshev A (2010) NPUA: a new approach for the analysis of computer experiments. Chemom Intell Lab Syst 104(2):333–340

Efron B (2005) Bayesians, frequentists, and scientists. J Am Stat Assoc 100(469):1–5

Evans JR, Olson DL (1998) Introduction to simulation and risk analysis. Prentice-Hall, Upper Saddle River

Fang K-T, Li R, Sudjianto A (2006) Design and modeling for computer experiments. Chapman & Hall/CRC, London

Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1–3):50–79

Forrester A, Sóbester A, Keane A (2008) Engineering design via surrogate modelling: a practical guide. Wiley, Chichester

Forrester JW (1961) Industrial dynamics. MIT, Cambridge

Giambiasi N, Carmona JC (2006) Generalized discrete event abstraction of continuous systems: GDEVS formalism. Simul Model Pract Theory 14(1):47–70

Gorissen D (2010) Grid-enabled adaptive surrogate modeling for computer aided engineering. Ph.D. dissertation, Ghent University, Ghent

Grubler, A., Y. Ermoliev, and A. Kryazhimskiy (2015), Coping with uncertainties-examples of modeling approaches at IIASA. *Technological Forecasting and Social Change*, in press

Harari O, Steinberg DM (2014) Convex combination of Gaussian processes for Bayesian analysis of deterministic computer experiments. Technometrics 56(4):443–454

Helton JC, Davis FJ (2003) Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. Reliab Eng Syst Saf 81:23–69

Helton JC, Hansen CW, Swift PN (2014) Performance assessment for the proposed high-level radioactive waste repository at Yucca mountain, Nevada. Reliab Eng Syst Saf 122:1–6

Henderson SG, Nelson BL (eds) (2006) Handbooks in operations research and management science, vol 13. North-Holland, Amsterdam

Ho Y, Cao X (1991) Perturbation analysis of discrete event dynamic systems. Kluwer, Dordrecht

Kamiński, B. (2015) Interval metamodels for the analysis of simulation input-output relations. Simulation Modelling Practice and Theory, 54:86–100

Karplus WJ (1983) The spectrum of mathematical models. Perspect Comput 3(2):4–13

Kleijnen JPC (1975) A comment on Blanning's metamodel for sensitivity analysis: the regression metamodel in simulation. Interfaces 5(3):21–23

Kleijnen JPC (1987) Statistical tools for simulation practitioners. Marcel Dekker, New York

Kleijnen JPC (2007) Risk analysis: frequentist and Bayesians unite! In: Yücesan E (ed) Proceedings of the 2007 INFORMS Simulation Society Research Workshop, Fontainebleau, pp 61–65

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Levy S, Steinberg DM (2010) Computer experiments: a review. AStA Adv Stat Anal 94(4):311–324

Mordecai Y, Dori D (2013) Model-based risk-oriented robust systems design with object-process methodology. Int J Strateg Eng Asset Manag 1(4):331–354

Müller J, Shoemaker CA (2014) Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems. J Glob Optim 60:123–144

Nance RE, Sargent RG (2002) Perspectives on the evolution of simulation. Oper Res 50(1):161–172

Nelson BL (2013) Foundations and methods of stochastic simulation; a first course. Springer, New York

Oden JT (2006) Revolutionizing engineering science through simulation. National Science Foundation (NSF) Blue Ribbon Panel on Simulation-based Engineering Science. National Science Foundation, Arlington

Picheny V, Ginsbourger D, Richet Y, Caplin G (2013) Quantile-based optimization of noisy computer experiments with tunable precision, including comments and rejoinder. Technometrics 55(1):1–36

Pidd M (2004)Computer simulation in management science, 5th edn. Wiley, Chichester

Poropudas J, Virtanen K (2008) Game theoretic approach to air combat simulation analysis. Systems Analysis Laboratory, Helsinki University of Technology

Rasmussen CE, Williams C (2006) Gaussian processes for machine learning. MIT, Cambridge

Razavi S, Gupta HV (2015) What do we mean by sensitivity analysis? The need for comprehensive characterization of "global" sensitivity in earth and environmental systems models. Water Resour Res 51 (in press)

Razavi S, Tolson BA, Burn DH (2012) Review of surrogate modeling in water resources. Water Resour Res 48, W07401:1–322

Regis RG (2014) Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. Eng Optim 46(2):218–243

Rieck K et al (2012) Support vector machines. In: Gentle JE, Haerdle W, Mori Y (eds) Handbook of computational statistics, concepts and fundamentals, vol 1, 2nd edn. Springer, Heidelberg, pp 883–926

Sakallı ÜS, Baykoç ÖF (2011) An optimization approach for brass casting blending problem under aletory and epistemic uncertainties. Int J Prod Econ 133(2):708–718

Saltelli, Ratto M, Tarantola S, Campolongo F (2005) Sensitivity analysis of chemical models. Chem Rev 105(7):2811–2827

Sanchez SM, Lucas TW, Sanchez PJ, Nannini CJ, Wan H (2012) Chapter 12: designs for large-scale simulation experiments, with applications to defense and homeland security. In: Hinkelmann K (ed) Design and analysis of experiments, volume 3, special designs and applications. Wiley, New York, pp 413–442

Santner TJ, Williams BJ, Notz WI (2003) The design and analysis of computer experiments. Springer, New York

Santos MI, Santos PM (2008) Sequential experimental designs for nonlinear regression metamodels in simulation. Simul Model Pract Theory 16(9):1365–1378

Schriber TJ (2009) Simulation for the masses: spreadsheet-based Monte Carlo simulation. In: Rossetti MD, Hill RR, Johansson B, Dunkin A, Ingalls RG (eds) Proceedings of the 2009 Winter Simulation Conference, Austin, pp 1–11

Shepard D (1968) A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 ACM National Conference, New York, pp 517–524. doi:10.1145/800186.810616

Shubik M (2002) Game theory and operations research: some musings 50 years later. Oper Res 50(1):192–196

Simchi-Levi D, Kaminsky P, Simchi-Levi E (2003) Designing and managing the supply chain: concepts, strategies, and case studies, 2nd edn. Irwin/McGraw-Hill, Boston

Sprent P, Smeeton NC (2007) Applied nonparametric statistical methods, 4th edn. Chapman & Hall/CRC, Atlanta

Sterman JD (2000) Business dynamics: systems thinking and modeling for a complex world. McGraw-Hill, Homewood

Storlie C, Swiler L, Helton J, Sallaberry C (2009) Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models, SAND report 2008-6570. Sandia, Albuquerque

Swain JJ (2005) “Gaming” reality. OR/MS Today 32(6):44–55

Swain JJ (2013) Simulation software: a better reality? OR/MS Today 40(5):48–59

Tan MHY (2014) Stochastic polynomial interpolation for uncertainty quantification with computer experiments. Technometrics (in press)

Van Gelder L, Das P, Janssen H, Roels S (2014) Comparative study of metamodelling techniques in building energy simulation: guidelines for practitioners. Simul Model Pract Theory 49:245–257

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690

Villa-Vialaneix N, Follador M, Ratto M, Leip A (2012) A comparison of eight metamodeling techniques for the simulation of N 2O fluxes and N leaching from corn crops. Environ Model Softw 34:51–66

Vose D (2000) Risk analysis; a quantitative guide, 2nd edn. Wiley, Chichester

Wang C, Duan Q, Gong W, Ye A, Di Z, Miao C (2014) An evaluation of adaptive surrogate modeling based optimization with two benchmark problems. Environ Model Softw 60:167–179

Wiebenga JH (2014) Robust design and optimization of forming processes. Ph.D. thesis, University of Twente, Enschede

Wu DD, Olson DL (2013) Computational simulation and risk analysis: an introduction of state of the art research. Math Comput Model 58(9–10):1581–1587

Xie W, Nelson BL, Barton RR (2014) A Bayesian framework for quantifying uncertainty in stochastic simulation. Oper Res 62(6):1439–1452

Zeigler BP, Praehofer H, Kim TG (2000) Theory of modeling and simulation, 2nd edn. Academic, San Diego

Zhu P, Zhang Y, Chen G (2011) Metamodeling development for reliability-based design optimization of automotive body structure. Comput Ind 62(7):729–741

2
Classic Regression Metamodels and Their Designs

This chapter discusses the basics of low-order polynomial regression metamodels and their designs. This chapter is organized as follows. Section 2.1 discusses black-box versus white-box approaches in the *design of simulation experiments* (DASE). Section 2.2 covers the basics of linear regression analysis. Section 2.3 focuses on first-order polynomial regression. Section 2.4 presents designs for estimating such first-order polynomials; namely, so-called *resolution-III* (R-III) designs. Section 2.5 augments the first-order polynomial with interactions (cross-products). Section 2.6 discusses *resolution-IV* (R-IV) designs, which give unbiased estimators of the first-order effects—even if there are two-factor interactions. Section 2.7 presents *resolution-V* (R-V) designs, which also enable the estimation of all the individual two-factor interactions. Section 2.8 extends the first-order polynomials to second-order polynomials. Section 2.9 presents designs for second-degree polynomials, focussing on *central composite designs* (CCDs). Section 2.10 briefly examines "optimal" designs and other designs. Section 2.11 summarizes the major conclusions of this chapter. The chapter ends with appendixes, solutions for the exercises, and references.

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_2

2.1 Introduction

In Chap. 1 we introduced the statistical theory on DASE. This theory views the simulation model as a black box—not as a white box.

Definition 2.1 *A black-box view of a simulation model observes the inputs and outputs of this simulation model, but not the internal variables and specific functions implied by the simulation's computer modules.*

To explain the difference between the black-box view and the white-box view, let us return to the M/M/1 example (with its Markovian arrival and service times and a single queue) in Chap. 1. Now we slightly reformulate this example; e.g., we replace the symbol n by c because n is a reserved symbol for another quantity in the current chapter.

Example 2.1 *Let the symbol a_{i+1} denote the interarrival time between customers i and $i+1$, s_i the service time of customer i, and w_i the waiting time of customer i. The output of interest is the average*

$$\overline{w} = \frac{\sum_{i=1}^{c} w_i}{c} \tag{2.1}$$

where c denotes the number of customers that stops the simulation run. The simulation starts in the empty state; i.e., the initial condition is $w_1 = 0$. The dynamics of a single-server system are specified by Lindley's recurrence formula

$$w_{i+1} = max\ (0, w_i + s_i - a_{i+1}). \tag{2.2}$$

The input variables s and a are sampled such that s has the service rate μ and a has the arrival rate λ, so the mean service and interarrival times are $1/\mu$ and $1/\lambda$. To sample these variables, the simulation may use the inverse of the exponential distribution function and a single PRN stream with $2c$ PRNs $r_1,\ r_2,\ \ldots,\ r_{2c-1},\ r_{2c}$:

$$s_i = \frac{-\ln r_{2i-1}}{\mu} \quad \textit{and} \quad a_{i+1} = \frac{-\ln r_{2i}}{\lambda} \tag{2.3}$$

Note that—instead of the average defined in Eq. (2.1)—the output of interest might have been the estimated 90 % quantile (also called percentile) of the waiting times; the estimator may then be the quantile estimator $w_{(\lceil .90c \rceil)}$ where $w_{(i)}$ $(i = 1, \ldots, c)$ denotes the order statistics—so $w_{(1)} \leq w_{(2)} \leq \ldots \leq w_{(c-1)} \leq w_{(c)}$—and the so-called ceiling function $\lceil 0.90c \rceil$ means that $0.90c$ is rounded upwards to the next integer. Another output of interest may be the estimated variance of the waiting time in the steady state, denoted by $s^2(w_i | i \uparrow \infty)$ or briefly $s^2(w)$—not to be confused with $s^2(\overline{w})$, which quantifies the accuracy of the estimator defined in Eq. (2.1).

Note: Example 2.1 illustrates a white-box view. Such a view is used by *perturbation analysis* (PA) and the *score function* (SF) or *likelihood ratio*

(LR) method. These methods estimate the gradient for local—not global—sensitivity analysis and for optimization; see the classic textbooks Ho and Cao (1991), Rubinstein and Shapiro (1993), and Spall (2003). Gradient estimation will be further discussed later on, in this chapter and in Chap. 6 on simulation optimization.

DASE does not view the simulation model as a white box, but as a *black box*. Such a black-box view is also used by *design of experiments* (DOE) for real-world experiments—see the classic textbook Montgomery (2009) and also Mee (2009)—and by *design and analysis of computer experiments* (DACE) for deterministic simulation experiments—see the classic textbooks Fang et al. (2006) and Santner et al. (2003).

Now we consider an example of such a black-box view of any *single-server* simulation model—not only the M/M/1 model. This model has as output w, which may denote the average waiting time (so a more traditional symbol would be $\overline{w}$), the estimated 90 % quantile, the estimated variance, etc. Suppose this simulation model has as inputs the arrival rate λ, the service rate μ, and the queueing priority rule, denoted by (say) QPR. Obviously this QPR is a qualitative input (various scales were discussed in Sect. 1.3). Suppose that QPR has three nominal values; namely, first-in-first-out (FIFO), last-in-first-out (LIFO), and shortest-processing-time-first (SPT). Note that the priority rule is implicitly fixed to be FIFO when using the notation M/M/1. In this example we furthermore assume that the single-server model has a fixed waiting room capacity, etc. A special input are the PRNs; e.g., the PRNs are generated through the popular *linear congruential method*

$$r_{i+1} = \frac{(an_i + b) \bmod m}{m} \quad (i = 0, 1, \ldots) \tag{2.4}$$

with the nonnegative integers a, b, and m; the symbol mod denotes the mathematical modulo operation; the *seed* of the PRNs is the nonnegative integer n_0 so the n_i are also nonnegative; we are running out of symbols, so n_i and m have nothing to do with n and m elsewhere in this chapter. A proper selection of the parameters a, b, and m should make the PRN stream r_1, r_2, … appear to behave like independent samples from the uniform distribution on the interval $[0, 1)$.

The default of PRN generators makes the computer select the PRN *seed* r_0; e.g., the computer uses its internal clock to select the value of the (micro)second measured at the start of the simulation experiment. Instead of using this default, we ourselves may select a seed. If multiple runs are made, then we should guarantee that the seeds of these runs do not create PRN streams that may overlap; such an overlap would imply that the replications are not IID. We might select the same seed for n runs with the simulation model where n denotes the number of input combinations; such a selection implies common random numbers (CRN).

Because Example 2.1 uses a single PRN stream in Eq. (2.3), the black-box view of this M/M/1 simulation model is

$$w = f_{\mathrm{M/M/1}}(\lambda, \mu, QPR, r_0) \tag{2.5}$$

where $f_{\mathrm{M/M/1}}(.)$ denotes the mathematical function defined by the computer program that implements the equations in Example 2.1; namely, Eq. (2.1) through Eq. (2.3). Obviously, $f_{\mathrm{M/M/1}}(.)$ is indeed a mathematical function; i.e., $f_{\mathrm{M/M/1}}(.)$ is a relation between a set of inputs and a set of outputs such that each input combination gives exactly one output value.

The black box in Eq. (2.5) changes if Example 2.1 uses two separate PRN streams; namely, one for the interarrival times a and one for the service times s. Then "the" seed r_0 in Eq. (2.5) must be replaced by the *vector of seeds* (say) $\mathbf{r}_0 = (r_{0;a}, r_{0;s})$ where $r_{0;a}$ and $r_{0;s}$ denote the seed of the (inter) arrival times a and the service times s, respectively. Because these a and s are statistically independent in the M/M/1 model, these seeds must be selected such that the two PRN streams do not overlap. Modern discrete-event simulation software makes the selection of seeds straightforward, even if the linear congruential generator specified in Eq. (2.4) is replaced by a more complicated generator. Details on PRNs can be found in Kelton et al. (2007) and Law (2015).

Examples that are more complicated than the single server in Example 2.1, are networks of servers; e.g., customers can choose among a number of parallel servers (as in a supermarket) or customers must proceed from one server to the next server (as in a hospital). Each server may have its own service rate. The priority rule may be more complicated (e.g., supermarket customers with no more than ten items may choose a special server). The computer implementation of such server networks may assign separate seeds to the arrival process and to each of the (say) e servers, so the seed r_0 is replaced by the seed vector $\mathbf{r}_0 = (r_{0;1}, \ldots, r_{0;e+1})'$.

A more *general* black-box equation than Eq. (2.5) is

$$\mathbf{w} = f_{\mathrm{sim}}(d_1, \ldots, d_k, \mathbf{r}_0) = f_{\mathrm{sim}}(\mathbf{d}, \mathbf{r}_0) \tag{2.6}$$

where $\mathbf{w}$ denotes the vector of simulation outputs; $f_{\mathrm{sim}}(.)$ denotes the mathematical function defined by the simulation computer code implementing the given simulation model; d_j $(j = 1, \ldots, k)$ is the jth input of the computer code; in deterministic simulation the seed vector $\mathbf{r}_0$ vanishes; the k inputs are collected in the vector $\mathbf{d} = (d_1, \ldots, d_k)'$.

The *design matrix* for the simulation experiment is $\mathbf{D} = (d_{i;j})$ with $i = 1, \ldots, n$ where n denotes the number of input combinations in that experiment. Usually, this $\mathbf{D}$ is *standardized* such that $-1 \leq d_{ij} \leq 1$; sometimes $\mathbf{D}$ is standardized such that $0 \leq d_{ij} \leq 1$. For example, a two-level design usually has elements that are either -1 or $+1$; a space-filing design usually has elements such that $0 \leq d_{ij} \leq 1$.

The simulation output $\mathbf{w}$ in Eq. (2.6) is a *multivariate* random variable that is meant to estimate (say) $\mathbf{\Theta}$, which denotes the vector with the relevant characteristics of the output distribution; e.g., the simulation's average output $\overline{w}$ estimates μ_w, which denotes the mean of the distribution of the simulation output w, and the simulation's order statistic $w_{(\lceil 0.90c \rceil)}$ estimates the 90 % quantile of that same distribution. In *deterministic* simulation, $\mathbf{r}_0$ vanishes so $\mathbf{w}$ becomes a *vector* with n outputs that is meant to estimate $\mathbf{\Theta}$, which now denotes the vector of relevant output characteristics such as the mean and the maximum of the simulation output in the experimental domain. In practice, many simulation models have indeed multiple outputs; examples are given in Kleijnen and Mehdad (2014) and Shi et al. (2014).

Let us consider a possible metamodel for the black-box model in Eq. (2.5) representing a single-server simulation model; for simplicity, we assume a fixed queueing discipline (say, FIFO). This metamodel may be a *first-order polynomial* in the arrival rate λ and the service rate μ, augmented with the additive error term e:

$$y = \beta_0 + \beta_1 \lambda + \beta_2 \mu + e \tag{2.7}$$

where y denotes the output of the metamodel for the average simulation output $\overline{w}$; β_0, β_1, and β_2 are the parameters of this metamodel; e is the residual or noise. This e includes both *lack-of-fit* of the metamodel—because this metamodel is a Taylor series approximation cutoff after the first-order effects—and *intrinsic noise*—caused by the PRNs. (In deterministic simulation, e does not include intrinsic noise.)

There are alternatives for Eq. (2.7); e.g., a simpler metamodel is

$$y = \beta_0 + \beta_1 x + e \tag{2.8}$$

where x denotes the traffic rate—in queueing theory usually denoted by ρ—so

$$x = \rho = \frac{\lambda}{\mu}. \tag{2.9}$$

We observe that statisticians often use ρ to denote a correlation coefficient; in this book, the context should clarify what the symbol ρ means. Obviously, Eq. (2.9) combines the two original inputs λ and μ in Eq. (2.7) into a single input ρ, inspired by queueing theory (and "common sense"?).

Equation (2.9) illustrates the use of *transformations*. Another useful transformation replaces y, λ, and μ in Eq. (2.7) by $\log(y)$, $\log(\lambda)$, and $\log(\mu)$; this logarithmic transformation makes the first-order polynomial approximate relative changes; i.e., the regression parameters collected in the vector $\beta = (\beta_0, \beta_1, \beta_2)'$ become "elasticity coefficients", which measure percentage changes.

Definition 2.2 *The elasticity coefficient of (say) y with respect to x is the relative change in y caused by a relative change in x: $(\partial y / \partial x)(x/y)$.*

Exercise 2.1 *Prove that the elasticity coefficient of y with respect to λ in Eq. (2.7) is β_1 if y is replaced by $\log(y)$ and λ by $\log(\lambda)$.*

Elasticity coefficients are popular in econometrics; e.g. Kleijnen and Van Schaik (2011) applies the logarithmic transformation to some—but not all—inputs, using data obtained through passive observation of a real system (a mussel auction in the Netherlands) instead of active simulation experimentation. The use of transformations illustrates that simulation analysts should be guided by knowledge of the real system and—if available—corresponding analytical models.

2.2 Linear Regression

First we discuss basic linear regression analysis. Next we discuss slightly advanced linear regression analysis, which uses several F-statistics. These F-statistics are known to be sensitive to the classic regression assumptions; namely, the outputs are independently and normally distributed with a common variance. In practice this advanced analysis may be replaced by the analysis presented in the next chapter. Hence, some readers may wish to skip this advanced analysis, and proceed to Sect. 2.3.

2.2.1 Basic Linear Regression Analysis

We apply the following general matrix representation for *linear regression* models with multiple inputs and a single output:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e} \tag{2.10}$$

where $\mathbf{y} = (y_1, \ldots, y_n)'$ denotes the n-dimensional vector with the dependent variable and n denotes the number of simulated input combinations (runs, observations); $\mathbf{X} = (\mathbf{x}_{i;j})$ denotes the $n \times q$ matrix of independent (explanatory) regression variables with $\mathbf{x}_{i;j}$ denoting the value of independent variable j in combination i ($i = 1, \ldots, n$; $j = 1, \ldots, q$); $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)'$ denotes the q-dimensional vector with regression parameters; and $\mathbf{e} = (e_1, \ldots, e_n)'$ denotes the n-dimensional vector with the residuals in the n combinations. For example, Eq. (2.7) has $q = 3$ parameters and Eq. (2.8) has $q = 2$ parameters; both equations include the *dummy* independent variable $x_{i;0} = 1$, which remains constant for all i values and corresponds with β_0, the effect of the dummy. If the general regression model specified in Eq. (2.10) includes a dummy, then β_1 in the vector $\boldsymbol{\beta}$ denotes the intercept, whereas β_0 denoted the intercept in the regression model specified in Eqs. (2.7) and (2.8). Initially we assume that no input combination is replicated; obviously, this assumption always holds in deterministic simulation.

When estimating the parameters $\boldsymbol{\beta}$ in the linear regression model specified in Eq. (2.10), the most popular criterion is so-called *least squares* (LS)—also called ordinary LS or OLS (generalized LS will be discussed in the next chapter). By definition, this criterion computes the estimator $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_q)'$ such that $\widehat{\boldsymbol{\beta}}$ minimizes the *sum of squared residuals*, SSR:

$$\underset{\widehat{\boldsymbol{\beta}}}{min}\ SSR = \sum_{i=1}^{n} (\widehat{e}_i)^2 = \sum_{i=1}^{n} (\widehat{y}_i - w_i)^2 = (\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w}) \tag{2.11}$$

where $\widehat{e}_i = \widehat{y}_i - w_i$ is the estimated residual for input combination i, $\widehat{y}_i$ is the regression predictor defined by

$$\widehat{y}_i = \sum_{j=1}^{q} x_{i;j} \widehat{\beta}_j = \mathbf{x}_i' \widehat{\boldsymbol{\beta}}, \tag{2.12}$$

and w_i denotes the simulation output of run i (e.g., the average waiting time of that run in discrete-event simulation, or the maximum output during the run in deterministic simulation). The solution of the minimization problem defined in Eq. (2.11) can be derived to be

$$\hat{\boldsymbol{\beta}} \ = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}. \tag{2.13}$$

Obviously, this $\hat{\boldsymbol{\beta}}$ exists only if the matrix $\mathbf{X}$ is not *collinear*; i.e., $\hat{\boldsymbol{\beta}}$ exists only if the inverse $(\mathbf{X}'\mathbf{X})^{-1}$ exists or this inverse remains stable in its numerical computation. For example, $\mathbf{X}$ is collinear in Eq. (2.7) if the two inputs λ and μ change simultaneously by the same amount; $\mathbf{X}$ is collinear in Eq. (2.8) if the input ρ is kept constant. The selection of a "good" $\mathbf{X}$ is the focus of the next sections, which discuss various designs.

Actually, the computation of $\widehat{\boldsymbol{\beta}}$ does not need to use Eq. (2.13); i.e., better numerical accuracy may result when solving the set of *normal equations*

$$\mathbf{X}'\mathbf{w} = \mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}}, \tag{2.14}$$

which follows from Eq. (2.10); also see Press et al. (2007). However, the next sections provide such good design matrixes that the computation of the LS estimates becomes trivial and numerical problems are negligible.

We emphasize that the LS criterion is a mathematical—not a statistical—criterion, which is also known as the L_2 *norm*. Other popular mathematical criteria are the L_1 and the L_∞ norms; see Cooper (2009), Narula and Wellington (2007), and Viana et al. (2014, Figure 4).

However, adding statistical assumptions about the output implies that the LS estimator has interesting statistical properties. We therefore examine the following definition.

Definition 2.3 *White noise (say) u is normally, independently, and identically distributed (NIID) with zero mean and some variance σ_u^2: $u \sim$ NIID$(0, \sigma_u^2)$.*

This definition leads us to the following comments:

- The concept of "white noise" is used in many disciplines. Not all these disciplines use the same definition of white noise; e.g., some definitions do not require normality.

- The linear regression model defined in Eq. (2.10) implies $\sigma_y^2 = \sigma_e^2$. The noise e is assumed to be white noise, provided the metamodel is a valid approximation. If the metamodel is indeed valid, then the dependent variable y in this metamodel may be replaced by the simulation output w. This w may indeed be *normally* distributed if it is an average computed from a "long" time series of individual simulation outputs. These individual outputs are autocorrelated (serially correlated), so the classic *central limit theorem* (CLT) does not apply. Yet it can be proven that—under specific conditions—this average tends to be normally distributed. A counterexample is a simulation with the estimated 90 % quantile $w_{(\lceil 0.90c \rceil)}$ as its output; nonnormality may be assumed for such an estimated quantile, unless the simulation run c is very long. We also refer to our discussion of the normality assumption in Sect. 3.3.1.

- Obviously, *deterministic* simulation violates the white noise assumption, so the statistical properties of the LS estimator $\widehat{\boldsymbol{\beta}}$ do not hold; also see Chap. 3 on classic assumptions versus simulation practice.

- The simulation outputs w_i and $w_{i'}$ with $i \neq i'$ are indeed *independent* if they use PRN streams that do not overlap. *CRN* violate this assumption, as we shall detail in Sect. 3.5.

- If a random variable is "identically" distributed, then it has a constant variance; see σ_u^2 in Definition 2.3. However, we may expect that the simulation outputs w_i do not have the same variance when the input combinations change; i.e., we expect that the variances σ_w^2 are heterogeneous (heteroscedastic, heteroskedastic) instead of homogeneous (homoscedastic, homoskedastic). For example, it is well-known that the variance of the steady-state waiting time in the M/M/1 model increases as the traffic rate increases; actually, this variance increases much more than the steady-state mean (this mean was displayed in Eq. (1.9)). This issue will be discussed Sect. 3.4.

In this chapter we assume that the simulation outputs w_i $(i = 1, \ldots, n)$ are indeed normally and independently distributed with the same variance (say) σ_w^2; obviously, these w_i may have different means in different input combinations i. Let us initially—until the discussion after Eq. (2.23)—assume that the linear regression model defined in Eq. (2.10) is a "valid" metamodel, which is defined as follows.

Definition 2.4 *A metamodel is valid if and only if its residual has zero mean: $E(e) = 0$.*

If $E(e) \neq 0$, then the metamodel is *biased*; i.e., the metamodel may either overestimate or underestimate the expected simulation output $E(w)$. The following definition is related to Definition 2.4.

Definition 2.5 *A metamodel fits "perfectly" if and only if all its estimated residuals are zero: $\widehat{e}_i = 0$ $(i = 1, \ldots, n)$.*

A *perfectly* fitting metamodel is "too good to be true"; i.e., n (number of simulation runs) is too small. Such a perfect fit implies that the well-known coefficient of determination R^2 has the ideal value one; see Sect. 3.6.1.

If the regression residual e is white noise, then LS gives the *best linear unbiased estimator* (BLUE). The condition is not "if and only if"; see the Gauss-Markov theorem discussed in Tian and Wiens (2006). Obviously, the LS estimator is indeed a *linear* transformation of the simulation response $\mathbf{w}$:

$$\hat{\boldsymbol{\beta}} = \mathbf{L}\mathbf{w} \tag{2.15}$$

where $\mathbf{L} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ because of Eq. (2.13); $\mathbf{L}$ is not random, whereas $\mathbf{w}$ is random in random simulation. Obviously, this linear estimator has the expected value

$$E(\hat{\boldsymbol{\beta}}) \;= \mathbf{L}[E(\mathbf{w})] \tag{2.16}$$

and the covariance matrix

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}} = \mathbf{L}\boldsymbol{\Sigma}_{\mathbf{w}}\mathbf{L}' \tag{2.17}$$

where $\boldsymbol{\Sigma}_{\mathbf{w}}$ denotes the covariance matrix of $\mathbf{w}$ (if the "white noise" assumption holds, then $\boldsymbol{\Sigma}_{\mathbf{w}} = \sigma_w^2\mathbf{I}$).

Exercise 2.2 *Prove that the LS estimator $\hat{\boldsymbol{\beta}}$ defined in Eq. (2.15) is an unbiased estimator of $\boldsymbol{\beta}$ if $E(e) = 0$.*

Equation (2.17) together with the white-noise assumption implies that the LS estimator has the following covariance matrix:

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2. \tag{2.18}$$

Like any covariance matrix, this $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}}$ must be symmetric and positive semidefinite. Equation (2.18) does not assume that the noise is normally distributed.

Exercise 2.3 *Prove that the LS estimator $\hat{\boldsymbol{\beta}}$ defined in Eq. (2.15) has the covariance matrix defined in Eq. (2.18) in case of white noise. (Hint: $(\mathbf{X}'\mathbf{X})^{-1}$ is symmetric.)*

Exercise 2.4 *Use Eq. (2.17) to prove that the variance of the average waiting time of a simulation run with c customers—defined in Eq. (2.1)—would be σ^2/c if and only if the individual waiting times were IID with variance σ^2 (actually, these waiting times have different variances and are autocorrelated).*

Given the white noise assumption, it can be proven that—among all linear unbiased estimators—the LS estimator is *best*, i.e., this estimator has the minimum variance. Obviously, the variances of the individual regression estimators $\widehat{\beta}_j$ are given by the main diagonal elements of Eq. (2.18); their covariances are given by the off-diagonal elements of the symmetric matrix. The matrix $(\mathbf{X}'\mathbf{X})$ is also known as the *information matrix*.

Instead of deriving an unbiased estimator, some statisticians minimize the *mean squared error* (MSE); i.e., they accept possible bias. In regression analysis, the MSE criterion leads to *ridge regression*. We do not know any application of ridge regression in simulation, so we do not further discuss this type of regression.

The *linear* LS estimator $\widehat{\boldsymbol{\beta}}$ has another interesting property if the simulation outputs $\mathbf{w}$ are *normally* distributed; i.e., $\widehat{\boldsymbol{\beta}}$ is then normally distributed too. Combining this property with the mean following from Eq. (2.16) and the covariance given in Eq. (2.18) gives

$$\widehat{\boldsymbol{\beta}} \sim N[\boldsymbol{\beta}, (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2].$$

Consequently, the individual estimated regression parameters $\widehat{\beta}_j$ may be tested through the *Student t-statistic with $n - q$ degrees of freedom*:

$$t_{n-q} = \frac{\widehat{\beta}_j - \beta_j}{s(\widehat{\beta}_j)} \quad \text{with} \quad j = 1, \ldots, q \tag{2.19}$$

where $s(\widehat{\beta}_j)$ is the square root of the jth element on the main diagonal of the covariance matrix for $\widehat{\boldsymbol{\beta}}$ given in Eq. (2.18) with σ_w^2 estimated through the *mean squared residuals* (MSR):

$$\text{MSR} = \frac{\text{SSR}}{n-q} = \frac{(\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w})}{n-q} \tag{2.20}$$

where SSR was given in Eq. (2.11). This MSR assumes that degrees of freedom are left over, after fitting the regression model: $n - q > 0$.

The t-statistic defined in Eq. (2.19) may be used to test whether an individual regression parameter β_j has a specific value such as the value zero:

$$H_0 : \beta_j = 0. \tag{2.21}$$

This *null-hypothesis* H_0 is rejected if the computed t-value is *significant*: $|t_{n-q}| > t_{n-q;1-\alpha/2}$ where $t_{n-q;1-\alpha/2}$ denotes the $1 - \alpha/2$ quantile of the (symmetric) distribution of t_{n-q}; this $t_{n-q;1-\alpha/2}$ is also called the upper $\alpha/2$ critical point of the t-distribution (obviously, $t_{n-q;1-\alpha/2} = -t_{n-q;\alpha/2}$). Only if we have strong evidence against H_0, we reject this hypothesis and accept the alternative hypothesis H_1; e.g., H_0 in Eq. (2.21) implies the alternative hypothesis H_1: $\beta_j \neq 0$.

To avoid the selection of a specific value for α in $t_{n-q;1-\alpha/2}$, we may present the so-called *p-value* which is the probability of obtaining a test statistic at least as extreme as the one that was actually observed, assuming that the null hypothesis is true. (We might say that the p-value is the α-value that changes the observed value of the statistic from nonsignificant to significant, but a search of the Internet suggests that the correct interpretation of the p-value is controversial.)

We point out that the *nonsignificant* inputs are usually removed from the fitted metamodel. However, we should keep in mind that the BLUE is still $\widehat{\beta}_j$, so we must have good (nonstatistical) reasons to replace $\widehat{\beta}_j$ by zero. One such reason may be that in sensitivity analysis we may apply the principle of *parsimony*, which we may colloquially call "keep it simple, stupid (KISS)". In optimization, however, we may keep the nonsignificant first-order effects because they may become important when we fit a first-order polynomial in another experimental area (when searching for the optimum applying RSM). For example, Dengiz et al. (2006) keeps two nonsignificant first-order effects in the regression metamodel because these two effects correspond with two decision variables in a simulated decision support system (DSS) that is to be optimized. Furthermore, we emphasize that an input may turn out to be *significant*, but this input may still be *unimportant*. For example, Breukers (2006) uses $m = 500$ replications so all inputs turn out to be significant; replications will be further discussed below (see the discussion around Eq. (2.24)). Significance versus importance is also discussed outside simulation; e.g., Lin et al. (2013) points out that a large sample (with, say, 10,000 observations) may lead to a very significant statistic (with a corresponding small p-value) and yet the practical importance may be small. We shall discuss the selection of the number of replications, in the next chapters.

2.2.2 Advanced Linear Regression Analysis

Instead of formulating a hypothesis involving a single parameter, we may formulate a *composite* or *joint* hypothesis involving several parameters; e.g., instead of H_0 in Eq. (2.21) we may define

$$H_0 : \beta_{j'} = \ldots = \beta_q = 0 \tag{2.22}$$

where—for simplicity of presentation—the q parameters are arranged such that the last $q - j' + 1$ parameters are hypothesized to be zero. To test this hypothesis, we may use an F-statistic; see, e.g., the general regression textbook Searle (1971). This test proceeds as follows.

1. Compute the SSR without H_0; this is called the SSR of the *unrestricted* or *full* regression model: SSR_{full}.
2. Compute the SSR under H_0, called the SSR of the *restricted* or *reduced* regression model: $\text{SSR}_{\text{reduced}}$. Obviously $\text{SSR}_{\text{reduced}} \geq \text{SSR}_{\text{full}}$ because imposing the constraint specified by H_0 in Eq. (2.22) increases the minimum value of SSR.
3. Compute
$$F_{q-j'+1;n-q} = \frac{SSR_{\text{reduced}} - SSR_{\text{full}}}{SSR_{\text{full}}}. \tag{2.23}$$

The H_0 in Eq. (2.22) is rejected if $F_{q-j'+1;n-q}$ exceeds the $1-\alpha$ quantile of the $F_{q-j'+1;n-q}$ distribution; that quantile may be denoted by $F_{q-j'+1;n-q;1-\alpha}$. Note that this H_0 uses a one-sided F test, whereas the H_0 in Eq.(2.21) uses a two-sided t test (obviously, such a two-sided t test should be used to test $H_0 : \beta_j = 0$).

Actually, before testing the individual inputs in H_0 defined in either Eqs. (2.19) or (2.22)—using either Eqs. (2.19) or (2.23)—we should test whether the *metamodel as-a-whole* is valid. Because classic regression analysis and DOE assume white noise, we use the so-called *lack-of-fit F-statistic*. In addition to white noise, this F-statistic assumes that at least one input combination is replicated. (In the next chapter we shall drop the white noise assumption and present alternative validation statistics based on R^2 and cross-validation.) This F-statistic compares $\widehat{y}$ (metamodel predictor) with $\overline{w}$ (average output of the underlying simulation model). Obviously, the probability of a significant difference between $\widehat{y}$ and $\overline{w}$ increases, as m_i (number of replications) increases. The increase of this probability is desirable if the metamodel is indeed inadequate; i.e., the power of the test should increase as m_i increases. Whether the lack-of-fit is *important* is determined by the goals of the metamodel and the simulation model. An extensive discussion of the role of these goals in the validation of metamodels is Kleijnen and Sargent (2000).

Denoting the number of replications of input combination i by m_i, we give the following definition.

Definition 2.6 *A replication of the input combination $\mathbf{d}_i$ of the simulation model implies that this $\mathbf{d}_i$ is simulated more than once, so $m_i > 1$.*

Below Eq. (2.10) we mentioned that we initially assume that no input combination is replicated. This assumption is realistic in *passive* observation of real systems, as in econometrics. In such passive observation, the independent variables are not controlled so they are actually random and the probability of multiple realizations of the same combination $\mathbf{x}_i = (x_{i1}, \ldots, x_{iq})'$ $(i = 1, \ldots, n)$ is negligible. However, in *active* experimentation with either real systems or random simulation models of real systems, we do control the input combinations $\mathbf{d}$ defined in Eq. (2.6);

i.e., more than once we may observe at least one combination of the independent variables $\mathbf{x}_i$ in Eq. (2.10). In *deterministic* simulation, however, it makes no sense to repeat the simulation with the same input combination, because it gives the same simulation output. In the rest of this section we focus on random simulation with replications.

The classic assumption is that replications are IID. In discrete-event simulation, this assumption is guaranteed if the replications use PRN streams that do not overlap. If the simulation output is the response of a steady-state simulation, then this IID assumption is guaranteed if the whole "long" run is replicated. The assumption is also satisfied if subruns are used and these subrun outputs have negligible autocorrelations. If the subruns are actually renewal (regenerative) cycles, then the IID assumption is satisfied by definition. Obtaining IID outputs in steady-state simulation is extensively discussed in the discrete-event simulation literature; e.g. Law (2015).

Replication implies that the matrix of independent variables $\mathbf{X}$ has at least one combination $\mathbf{x}$ repeated; e.g., if the first combination of λ and μ in Eq. (2.7) is replicated three times ($m_1 = 3$) and these values are 0.5 and 1.0, respectively, then the first three rows of $\mathbf{X}$ are

$$\begin{bmatrix} 1 & 0.5 & 1.0 \\ 1 & 0.5 & 1.0 \\ 1 & 0.5 & 1.0 \end{bmatrix}.$$

In general, replication increases the number of rows of $\mathbf{X}$ from n to (say) N defined as follows:

$$N = \sum_{i=1}^{n} m_i \tag{2.24}$$

with m_i *identical* rows $\mathbf{x}'_i$ if combination i is simulated m_i times. Consequently, MSR defined in Eq. (2.20) now has more degrees of freedom; namely, $N - q$ instead of $n - q$, as we shall see. Obviously, Eq. (2.24) also holds in the special case $m_i = 1$ for some i or all i.

Besides Definition 2.6 we use the following definition, throughout this book.

Definition 2.7 *A macroreplication of an experiment with a simulation model means that the whole simulation experiment defined by the* $N \times k$ *design matrix* $\mathbf{D}$ *is repeated such that only the seed vector* $\mathbf{r}_0$ *is changed.*

It is possible to keep the number of rows in $\mathbf{X}$ limited to the n *different* combinations. The output corresponding with $\mathbf{x}_i$ then becomes the output averaged over the m_i replications. So we should distinguish the following two situations:

- The number of replications is the same in all n simulated input combinations: $m_i = m$. The LS estimate may then be computed from the n simulation output averages, $\overline{w}_i$ ($i = 1, \ldots n$). The MSR can still be computed analogously to Eq. (2.20), replacing $\mathbf{w}$ by $\overline{\mathbf{w}} = (\overline{w}_1, \ldots \overline{w}_n)'$:

$$\mathrm{MSR}_{\overline{w}} = \frac{\mathrm{SSR}_{\overline{w}}}{n-q} = \frac{(\widehat{\mathbf{y}} - \overline{\mathbf{w}})'(\widehat{\mathbf{y}} - \overline{\mathbf{w}})}{n-q}, \tag{2.25}$$

which has expected value $\mathrm{Var}(\overline{w}) = \mathrm{Var}(w)/m$ instead of $\mathrm{Var}(w)$.

- The number of replications is not constant: $m_i \neq m$. The MSR can then be computed from the averages $\overline{w}_i$ $(i = 1, \ldots n)$ weighted by m_i:

$$\mathrm{MSR}_{\overline{w}}(m_i) = \frac{\sum_{i=1}^{n} m_i(\widehat{y}_i - \overline{w}_i)^2}{(\sum_{i=1}^{n} m_i) - q}. \tag{2.26}$$

If $\mathbf{x}_i$ is replicated $m_i > 1$ times, then an alternative for the MSR estimator is the *classic* variance estimator:

$$s^2(w_i) = \frac{\sum_{r=1}^{m_i}(w_{i;r} - \overline{w}_i)^2}{m_i - 1} \;(i = 1, \ldots, n) \tag{2.27}$$

with

$$\overline{w}_i = \frac{\sum_{r=1}^{m_i} w_{i;r}}{m_i}. \tag{2.28}$$

We provide the following comments on Eq. (2.27):

- The *average* in Eq. (2.28) is computed from the m_i replications; this average should not be confused with the average computed from the autocorrelated individual waiting times in a single simulation run; see Eq. (2.1).

- The average in Eq. (2.28) and the sample variance in Eq. (2.27) are statistically *independent* if the simulation outputs $w_{i;r}$ are NIID, as any basic statistics textbook mentions.

- The variance estimator in Eq. (2.27) is a *chi-square* variable with $m_i - 1$ degrees of freedom; see again any statistics textbook.

- The denominator $m_i - 1$ in Eq. (2.27) makes the estimator unbiased; the *maximum likelihood estimator* (MLE) can be proven to use the denominator m_i. (The LS estimator $\widehat{\boldsymbol{\beta}}$ is also the MLE, given the white noise assumption.)

Because of the common variance assumption implied by the white noise assumption (see Definition 2.3 above), the n variance estimators in Eq. (2.27) may be *pooled* using their degrees of freedom as weights:

$$s^2(w) = \frac{\sum_{i=1}^{n}(m_i - 1)s_i^2}{\sum_{i=1}^{n}(m_i - 1)}. \tag{2.29}$$

Altogether—if there are replications—we have the following two variance estimators:

- $\text{MSR}_{\overline{w}}$ defined in Eq. (2.25) for an equal number of replications per input combination ($m_i = m > 1$), and $\text{MSR}_{\overline{w}}(m_i)$ defined in Eq. (2.26) for $m_i > 1$. Obviously, these two estimators use the fitted regression model; if this regression model is not valid, then they overestimate the true variance. Therefore we put this estimator in the numerator of the lack-of-fit F-statistic—discussed next—and use a one-sided test.

- The *pooled* variance estimator in Eq. (2.29), which uses $m_i > 1$ replications. This estimator does not use a fitted regression model, so this estimator is unbiased—assuming the simulation outputs for a replicated combination are IID. Note that these outputs do not need to be NIID; however, the F-statistic does assume NIID.

These two estimators may be compared through the so-called *lack-of-fit F-statistic*. We point out that an F-statistic assumes that its numerator and denominator are independent. Actually, the lack-of-fit F-statistic has a numerator that uses MSR, which depends on $\overline{w}_i$ and $\widehat{y}_i$; this $\widehat{y}_i$ uses $\overline{w}_i$. The denominator depends on the pooled variance estimator, which uses s_i^2; it is well known that s_i^2 is independent of $\overline{w}_i$ if the responses w are normally distributed.

We again distinguish between an equal and an unequal number of replications, as we did in Eqs. (2.25) and (2.26).

- If each input combination i is replicated a constant number of times so $m_i = m$, then the lack-of-fit F-statistic is

$$F_{n-q;n(m-1)} = \frac{m}{n-q} \frac{(\overline{\mathbf{w}} - \widehat{\mathbf{y}})'(\overline{\mathbf{w}} - \widehat{\mathbf{y}})}{\sum_{i=1}^{n} s^2(w_i)/n} \tag{2.30}$$

  where $s^2(w_i)$ was defined in Eq. (2.27), and $(\sum_{i=1}^{n} s^2(w_i)/n)/m$ is an unbiased estimator of $\text{Var}(\overline{w}) = \text{Var}(w)/m$; however, $(\overline{\mathbf{w}} - \widehat{\mathbf{y}})'$ $(\overline{\mathbf{w}} - \widehat{\mathbf{y}})/(n-q)$ is an unbiased estimator of the same quantity, only if the regression model is a valid approximation.

- If the number of replications per combination is not constant, then this statistic becomes (see Montgomery 2009, p. 413 or any other textbook on DOE):

$$F_{n-q;N-n} = \frac{\sum_{i=1}^{n} m_i(\overline{w}_i - \widehat{y}_i)^2/(n-q)}{\sum_{i=1}^{n} \sum_{r=1}^{m_i} (w_{i;r} - \overline{w}_i)^2/(N-n)}. \tag{2.31}$$

  The numerator uses $\text{MSR}_{\overline{w}}(m_i)$ defined in Eq. (2.26) so it is computed from the *average* simulation output per combination; at least one combination is replicated (usually, the center of the experimental area is replicated when applying classic DOE to simulation).

Obviously, we reject the regression model if the F-statistic defined in either Eq. (2.30) or Eq. (2.31) is significantly high.

Note: Alternative tests for the validation of the fitted metamodel will be presented in Sect. 3.6.2, including the popular statistic R^2 and cross-validation statistics such as PRESS. Those tests do not assume white noise, so they may also be applied to deterministic simulation. Moreover, they may be applied to other metamodel types, such as Kriging models.

The lack-of-fit F-statistic becomes statistically significant whenever the estimated variance of the underlying simulation output w_i becomes "small". For example, if w represents the waiting time averaged over the simulation run-length (say) T so $w = \sum_{t=1}^{T} w_t/T$, then $\text{Var}(w)$ goes to zero as T goes to infinity. So any deviation between the observed simulation response w and the regression predictor is declared significant. In practice, however, these deviations may be unimportant. (R^2 does not have this characteristic.)

So we may use either the *individual* simulation outputs $w_{i;r}$ or the *averages* $\overline{w}_i$. Both outputs give identical $\widehat{\boldsymbol{\beta}}$ but different MSE. For example, suppose that one individual output increases with the constant c, while another individual output for the same input combination decreases with that same constant. The average $\overline{w}_i$ then remains the same, so $\widehat{\boldsymbol{\beta}}$ and MSE computed from the averages remain the same. However, MSE computed from the individual outputs increases. The best estimator is the latter one, because it has more degrees of freedom; namely, $N - q$ instead of $n - q$ where $N = \sum_{i=1}^{n} m_i$.

We conclude this section (on basic regression analysis) with a long exercise that covers many issues discussed in this section.

Exercise 2.5 *Because experiments with simulation models do not satisfy the assumptions of basic regression analysis (e.g., M/M/1 simulation models do not have constant response variances), you may perform the following experiments with Monte Carlo models. Suppose that the simulation model has the I/O function (also see Eq. (2.5))*

$$w = \beta_0 + \beta_1 z + \beta_2 z^2 + u \tag{2.32}$$

where $u \sim$*NIID*$(0, \sigma^2)$ *so* $\sigma^2 = \sigma_w^2 = \sigma_u^2$. *More specifically, suppose*

$$w = 100 + 5z + z^2 + u \quad \text{if} \quad 1 \leq z \leq 10 \tag{2.33}$$

where $u \sim$*NIID*$(0, 4)$. *Following Algorithm 1.1 (in Chap. 1), you start with the first-order polynomial metamodel*

$$y = \gamma_0 + \gamma_1 z + e \quad \text{with} \quad 1 \leq z \leq 10. \tag{2.34}$$

To fit this metamodel and validate it, you select n *values for* z *in the global experimental domain* $1 \leq z \leq 10$*; e.g., you select* $n = 5$ *equispaced values* *in this domain so* $z_1 = 1$, $z_2 = 3.25$, $z_3 = 5.50$, $z_4 = 7.75$, *and* $z_5 = 10$. *Furthermore, you sample* m_i $(i = 1, \ldots, 5)$ *replications for input value* z_i; *e.g.,* $m_i = 4$ *so* $\sigma_{\overline{z}} = \sigma_z/\sqrt{m} = 2/2 = 1$.

(a) *Compute the LS estimate* $\widehat{\boldsymbol{\gamma}} = (\widehat{\gamma}_0, \widehat{\gamma}_1)'$ *using Eq. (2.13).*

(b) *Compute the LS estimate* $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_0, \widehat{\boldsymbol{\beta}}_1)$ *using a standardized input* x *such that* $0 \leq x \leq 1$.

(c) *To validate the metamodel fitted in (a) or (b), use the lack-of-fit* F*-test defined in Eq. (2.30). (Hint:* $\widehat{y} = \mathbf{x}_i'\widehat{\boldsymbol{\beta}} = \mathbf{z}_i'\widehat{\boldsymbol{\gamma}}$ *so this* F*-statistic is scale-free.)*

(d) *Let us assume that (c) gives a significant* F*-statistic for a type-I error rate* α *with the value (say) 0.10; we make this assumption because a first-order polynomial metamodel is fitted, whereas the simulation has a second-order polynomial I/O function including intrinsic noise* σ^2 *that seems not too big. Following Algorithm 1.1, you next select an alternative metamodel; namely, a second-order polynomial*

$$y = \gamma_0 + \gamma_1 z + \gamma_1 z^2 + e \quad \textit{with} \quad 1 \leq z \leq 10. \tag{2.35}$$

Obviously, you have enough I/O combinations to fit this model with its three parameters: $n = 5 > q = 3$. *You should again validate this metamodel, using the* F*-test in Eq. (2.30). We expect that now you find a nonsignificant* F*-statistic when using* $\alpha = 0.10$, *because you fit a second-order polynomial and the simulation has indeed a second-order polynomial I/O function.*

(e) *Next, consider the following alternative for a second-order polynomial metamodel; namely, a first-order polynomial restricted to a local area that is concentrated around the middle of the experimental domain:*

$$y = \gamma_0 + \gamma_1 z + e \quad \textit{with} \quad 5 \leq z \leq 6. \tag{2.36}$$

To fit and validate this metamodel, you select n *values for* z *in the local experimental domain* $5 \leq z \leq 6$; *e.g., select* $n = 3$ *equispaced values in this domain so* $z_1 = 5$, $z_2 = 5.5$, *and* $z_3 = 6$. *You still sample* $m_i = 4$ *replications. You may use this local metamodel if the goal of your experiment is to estimate the gradient for simulation optimization. Does your (BLUE) estimate* $\widehat{\gamma}_1$ *point you in the right direction when you wish to maximize the simulation output; i.e., is* $\widehat{\gamma}_1$ *positive?*

(f) *Next we pretend that the simulation is so expensive that the number of new simulation runs should be minimized. Therefore you assume a first-order polynomial instead of a second-order polynomial. You fit this first-order polynomial for the following three old input values:*

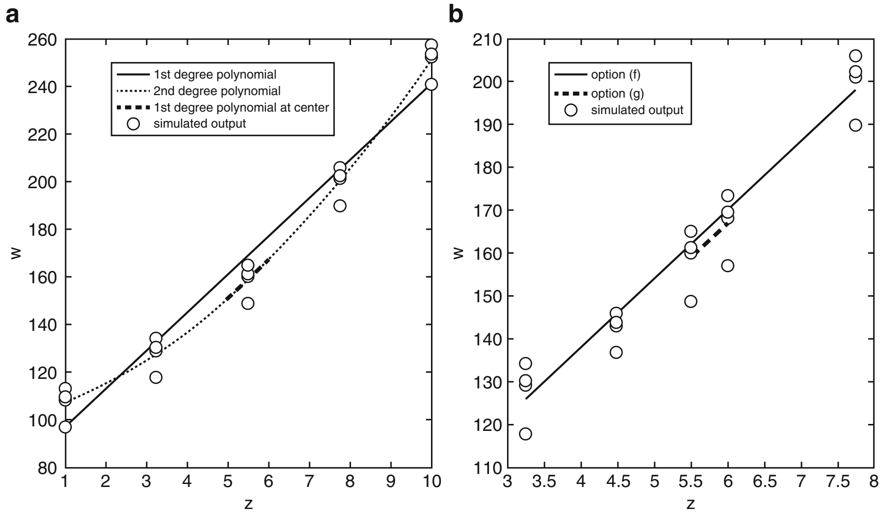


FIGURE 2.1. Scatterplot for Monte Carlo experiment defined in Exercise 2.5

$z_2 = 3.25$, $z_3 = 5.50$, and $z_4 = 7.75$; i.e., you ignore the extreme inputs $z_1 = 1$ and $z_5 = 10$. Use the $m = 4$ replications for each of these three old input values. Validate this fitted metamodel.

(g) *Instead of using the approach in (f), you obtain $m = 4$ replications—assuming that fewer replications would make the observed average simulation output too noisy—for a single new input value; namely, $z = 6$. Fit a first-order polynomial to these new data ($z = 6$, $\overline{w}_6$) and the old I/O data that are closest to these new input value; namely, ($z = 5.50$, $\overline{w}_{5.5}$). Do you find $\widehat{\gamma}_1 > 0$?*

Summarize your experimental results in a scatterplot such as Fig. 2.1, which shows results for a specific PRN stream; i.e., the horizontal axis represents z_i ($i = 1, \ldots, n$) and the vertical axis represents $w_{i;r}$ ($r = 1, \ldots, m$); various metamodels are fitted.

2.3 Linear Regression: First-Order Polynomials

To estimate the parameters of a black-box metamodel—e.g., the parameter vector $\boldsymbol{\beta}$ in the linear regression model defined in Eq. (2.10)—we must experiment with the simulation model; i.e., we must first change the inputs of the simulation and run the simulation, and next we must analyze the resulting I/O data. In this section, we examine first-order polynomial metamodels.

Let us start with the simplest metamodel; namely, the first-order polynomial with a single standardized input x:

$$y = \beta_0 + \beta_1 x + e. \tag{2.37}$$

Obviously, this polynomial has $q = 2$ regression parameters; whereas mathematicians use the symbols β_0 and β_1 as we do in Eq. (2.37), statisticians use β_1 and β_2 as we do in Eq. (2.10) for the general linear regression model. Geometrically, Eq. (2.37) implies a straight line in the (x, y) plane. To fit such a straight line, it obviously suffices to have only $n = 2$ observations (x_i, y_i) $(i = 1, 2)$; see again Fig. 2.1, which includes a first-order polynomial (besides a second-order polynomial). A first-order polynomial may provide a valid metamodel for a "small" experimental area; i.e., the first-order polynomial is fitted only *locally* (Taylor series argument). Furthermore, selecting the two input values x_1 and x_2 *as far apart as possible* gives the "best" estimator of the first-order effect (slope) β_1—given the white noise assumption for e. This assumption implies a *constant variance* $\sigma_w^2 = \sigma_y^2 = \sigma_e^2$ and statistical *independence* or $\mathbf{\Sigma_w} = \sigma_w^2 \mathbf{I}$. So $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}} = \sigma_w^2 (\mathbf{X}'\mathbf{X})^{-1}$; see Eq. (2.18).

Exercise 2.6 *Prove that the OLS estimator $\widehat{\beta}_1$ has minimum variance if the lower value of x in Eq. (2.37) denoted by (say) l and the upper value u are as far apart as possible.*

Next we consider the (more general) *first-order polynomial* metamodel with $k \geq 1$ independent variables x_j $(j = 1, \ldots, k)$:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + e. \tag{2.38}$$

This metamodel implies that the general linear regression model defined in Eq. (2.10) now has $q = k + 1$ regression parameters. An example is the first-order polynomial metamodel with the arrival rate λ and the service rate μ for the M/M/1 simulation model, given in Eq. (2.7).

In practice, such a metamodel may be useful when estimating the *optimal* values for the inputs of a simulation model. For example, we may wish to estimate the combination of input values that maximizes the profit of a simulated company. There are many methods for estimating the optimal input combination (see Chap. 6). Some of these methods use the gradient, which quantifies *local* marginal effects; see the next definition.

Definition 2.8 *The gradient $\nabla(y)$ of a function $y(x_1, \ldots, x_k)$ is the vector with the first-order partial derivatives: $\nabla(y) = (\partial y/\partial x_1, \ldots, \partial y/\partial x_k)'$.*

2.3.1 Scaling the Inputs

It is convenient and traditional in DOE to use *scaled*—also called *coded* or *standardized*—inputs. If each input has only two values in the whole experiment involving n input combinations, then these values may be denoted by -1 and $+1$. This implies the following linear transformation where z_j denotes the quantitative input j measured on the original scale, l_j denotes the lower value of z_j in the experiment (so $l = \min_i z_i$ with $i = 1, \ldots, n$), and u_j the upper value (so $u = \max_i z_i$):

$$x_{ij} = a_j + b_j z_{ij}$$
$$\text{with } a_j = \frac{l_j + u_j}{l_j - u_j};\ b_j = \frac{2}{u_j - l_j};\ j = 1, \ldots, k;\ i = 1, \ldots n. \tag{2.39}$$

This transformation implies

$$x_{ij} = \frac{z_{ij} - \overline{z}_j}{(u_j - l_j)/2} \tag{2.40}$$

where $\overline{z}_j$ denotes the average value of input j in a *balanced* experiment, which means that each input is observed at its lower value in $n/2$ combinations (hence this input is observed at its upper value in the other half), as is the case in a 2^k design. The factor $(u_j - l_j)$ in the denominator of Eq. (2.40) is known as the *range* of input j; the range is a well-known quantitative measure for the variation of a variable, besides the variance.

Exercise 2.7 *Simulate an M/M/1 queue with a traffic rate between 0.2 and 0.5, and fit a first-order polynomial metamodel; also see Eq. (2.8). Use standardized inputs for this metamodel applying Eq. (2.39). Use this metamodel to predict the simulation output for a traffic rate of 0.3 and 0.4, respectively. Which standardized x-values correspond with the original traffic rates 0.3 and 0.4?*

The scale of the original input z in Eq. (2.39) may be an interval, a ratio, or an absolute scale; see the discussion of scales at the end of Sect. 1.3. If z has either a nominal scale or an ordinal scale and z has only two levels, then the coding remains simple; i.e., we arbitrarily associate one level with -1 and the other level with $+1$ (on purpose, we now speak of "level" instead of "value"). For example, in a queueing simulation, one level may represent the FIFO priority rule, and the other level may represent LIFO.

The coding does not remain so simple if an input has a *nominal scale with more than two levels*. For example, Kleijnen (1995) discusses a simulation model of a sonar system that searches for mines on the sea bottom; this bottom is a nominal input with the three values clay, sand, or rocks. The type of sea bottom may affect the sonar's output. In this case study, the simulation analysts erroneously coded these three bottom types as -1, 0, and $+1$. The correct coding may be done through *multiple binary* variables—each coded as 0 and 1; mathematical details are discussed in Appendix 1. In the remainder of this book, we do not consider qualitative inputs with more than two levels.

Standardizing in such a way that each input—either quantitative or qualitative—varies between -1 and $+1$ is useful when *comparing* the effects of multiple inputs, as we do in sensitivity analysis. Figure 2.2 gives an

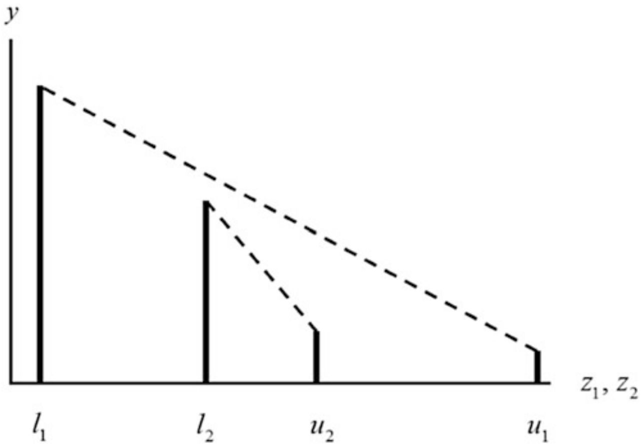


FIGURE 2.2. Scaling effects when comparing the effects of two inputs z_1 and z_2 with different ranges, in a first-order metamodel with output y

example with two quantitative inputs with different ranges, but the same scale (if the two scales were different, then two horizontal axes would be needed). The marginal effect of z_2 is higher than the marginal effect of z_1; see the slopes of the dashed lines (response curves). However, the range of z_1 is much bigger so "the" effect of this input is larger. If the standardization defined in Eq. (2.40) is applied, then the standardized effect of z_1 exceeds that of z_2.

Instead of the standardized inputs x_j in Eq. (2.38), we may use the original inputs z_j with $l_j \leq z_j \leq u_j$:

$$y = \gamma_0 + \gamma_1 z_1 + \ldots + \gamma_k z_k + e. \tag{2.41}$$

The intercept in Eq. (2.38) equals the expected output at the center of the experimental area, because $E(y) = \beta_0$ if $x_j = 0$ for all j. However, the intercept in Eq. (2.41) equals the output when $z_j = 0$ for all j—which may be very far away from the experimental area! Obviously, the marginal effects in Eq. (2.38) are $\partial y/\partial x_j = \beta_j$; the marginal effects in Eq. (2.41) are $\partial y/\partial z_j = \gamma_j$. The total effect in Eq. (2.38) when changing the inputs over their experimental domain is $2\beta_j$, because all standardized inputs x_j have the same range; namely, $1 - (-1) = 2$. So $\boldsymbol{\beta} = (\beta_1, \ldots \beta_k)'$ quantifies the relative importance of the k inputs. The total effect in Eq. (2.41) when changing the inputs over their experimental domain are $\gamma_j(u_j - l_j)$. To *rank* the input effects, the absolute values of the standardized effects β_j should be sorted—if a first-order polynomial is a valid metamodel (else, interactions should also be considered; see Sect. 2.5 below); the absolute values are needed for qualitative inputs, which have levels that are arbitrarily associated with the standardized values -1 and 1. We find the original scales less convenient in sensitivity analysis. The original scales are used in optimization through response surface methodology (RSM), which uses the gradient $\nabla(y) = (\partial y/\partial z_1, \ldots, \partial y/\partial z_k)'$; see Sect. 6.2. The t-statistic defined in Eq. (2.19) has the same value for the original and the standardized effects, as is easy to prove ($\mathrm{Var}[(a_j + b_j\widehat{\gamma}_j)] = b_j^2\mathrm{Var}(\widehat{\gamma}_j)$, etc).

Exercise 2.8 *Bettonvil and Kleijnen (1990) discusses a third type of standardization that centers the original inputs z_j around $\overline{z}_j$, defined below Eq. (2.40):*

$$y = \delta_0 + \delta_1(z_1 - \overline{z}_1) + \ldots + \delta_k(z_k - \overline{z}_k) + e. \tag{2.42}$$

Derive the marginal effects of z_j, and the total effect over the range of z_j, when using this metamodel and a balanced design.

2.3.2 One-Factor-at-a-Time Designs Versus Factorial Designs

To estimate the gradient (defined in Definition 2.8), many mathematicians change one input at a time—using two or three values for each input. However, the statistical theory on DOE proves that it is more efficient to estimate the gradient from a first-order polynomial estimated through a so-called factorial design that is either a full factorial or a fractional factorial design (we shall define these designs below). Not only for optimization but also for other goals of simulation, the LS estimator of the $k+1$ parameters in the vector $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ in Eq. (2.38) often uses one of the following two design types:

- One-factor-at-a-time designs
- Full factorial designs

In practice, the simulationists often change each input one-at-a-time (called the *ceteris paribus* approach in econometrics). DOE, however, may use a 2^k design where k denotes the number of inputs and 2 denotes the number of levels (values) per input; this design is called a two-level full factorial. Obviously, two values suffice to estimate the first-order polynomial in Eq. (2.38).

To compare one-at-a-time designs and full factorial designs, we first discuss the simplest example with multiple inputs—namely, $k = 2$ inputs—in detail.

Example 2.2 *Suppose that the number of inputs is only two, so $k = 2$. To evaluate the different design types, we compare the variances of the estimated regression parameters in a one-at-a-time design and in a full factorial design, respectively—assuming a first-order polynomial metamodel. We also assume that there are no replications, so $m_i = 1$.*

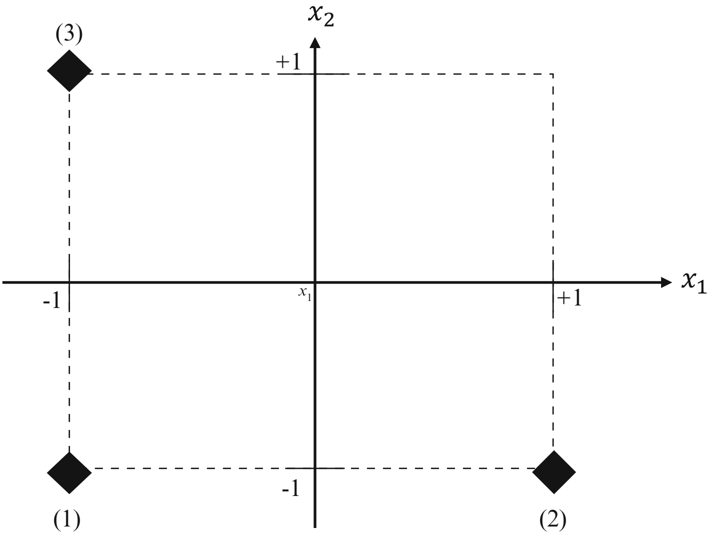


FIGURE 2.3. A one-at-a-time design for two inputs x_1 and x_2

A one-at-a-time design is presented in Fig. 2.3. This design is only one of the possible designs that belong to this popular design class; other designs in this class use three (instead of two) values, but we have already pointed out that two values suffice for a first-order polynomial (see the discussion of Eq. (2.37)). Moreover, we assume that the combination denoted by (1) in this plot, is the so-called base value; e.g., the current input combination in the real system being simulated. The other two combinations in Fig. 2.3 increase input 1 and input 2, respectively. Obviously, the design could also be "mirrored" so the first combination would become $(+1,+1)$ *instead of* $(-1,-1)$*. Figure 2.3 corresponds with the following design matrix:*

$$\mathbf{D} = \begin{bmatrix} -1 & -1 \\ +1 & -1 \\ -1 & +1 \end{bmatrix}.$$

This $\mathbf{D}$ *gives* $\mathbf{X}$ *for the general linear regression model in Eq. (2.10):*

$$\mathbf{X} = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \\ +1 & -1 & +1 \end{bmatrix} = [\mathbf{1}_3\mathbf{D}]$$

where $\mathbf{1}_3 = (1,1,1)'$*; in general, we let* $\mathbf{1}_n$ *denote a column vector with all its* n *elements equal to* 1*. For convenience we assume that* $\sigma_w^2 = 1$*.*

This gives

$$\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix}^{-1} = \begin{bmatrix} 0.5 & 0.25 & 0.25 \\ 0.25 & 0.5 & 0.25 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}$$

where

$$\widehat{\boldsymbol{\beta}}=(\widehat{\beta}_0,\widehat{\beta}_1,\widehat{\beta}_2)'=(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}=\begin{bmatrix} 0 & 0.5 & 0.5 \\ -0.5 & 0.5 & 0 \\ -0.5 & 0 & 0.5 \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}=\begin{bmatrix} 0.5w_2+0.5w_3 \\ 0.5w_2-0.5w_1 \\ 0.5w_3-0.5w_1 \end{bmatrix}.$$

This $\widehat{\boldsymbol{\beta}}$ *agrees with common sense; e.g.,* β_2 *is estimated by the difference between the third observation in Fig. 2.3 and the base observation which is combination 1 in this plot. We point out that each of the three regression parameters is estimated from only two of the three simulation outputs.*

The 2^2 *design adds a fourth combination to Fig. 2.3; namely, the combination* $(+1,+1)$*. Hence,* $\mathbf{X}$ *becomes*

$$\mathbf{X} = \begin{bmatrix} +1 & -1 & -1 \\ +1 & +1 & -1 \\ +1 & -1 & +1 \\ +1 & +1 & +1 \end{bmatrix}$$

This $\mathbf{X}$ *together with* $\sigma_w^2 = 1$ *gives*

$$\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix}^{-1} = \begin{bmatrix} 0.25 & 0 & 0 \\ 0 & 0.25 & 0 \\ 0 & 0 & 0.25 \end{bmatrix}$$

and

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} = \begin{bmatrix} 0.25w_1+0.25w_2+0.25w_3+0.25w_4 \\ 0.25w_2-0.25w_1-0.25w_3+0.25w_4 \\ 0.25w_3-0.25w_2-0.25w_1+0.25w_4 \end{bmatrix}.$$

This $\widehat{\beta}$ *again agrees with common sense; e.g.,* β_2 *is now estimated by subtracting the average of the first and second outputs from the average of the third and fourth outputs—which agrees with Fig. 2.3 augmented with the fourth combination. We emphasize that each of the three regression parameters is now estimated from all four outputs.*

The variances of the estimated parameters are 0.25 *for the factorial design, whereas these variances are* 0.5 *for the one-at-a-time design. These variances, however, should be corrected for the number of combinations; this correction gives* $4 \times 0.25 = 1.0$ *and* $3 \times 0.5 = 1.5$ *. So the factorial design is more "efficient". (We shall also discuss examples with exactly the same number of combinations in both design types, which simplifies* *the comparison of their efficiencies.) Moreover, the estimated parameters are uncorrelated in the factorial design; in the one-at-a-time design, the correlations are* $0.25/0.5 = 0.5$*. Under the normality assumption, zero correlation implies independence; obviously, independent estimators simplify the statistical analysis.*

The 2^2 design, which features in this example, has "orthogonal" columns, defined as follows.

Definition 2.9 *Two* n*-dimensional vectors (say)* $\mathbf{z}_j$ *and* $\mathbf{z}_{j'}$ *(with* $j \neq j'$*) are orthogonal if their inner product* $\sum_{i=1}^{n} z_{i;j} z_{i;j'}$ *is zero.*

Note: A similar critique of one-at-a–time designs can be found in Spall (2010). However, Voelkel (2005) gives a more favorable discussion of one-at-a-time designs; e.g., factorial designs imply input combinations that are more extreme (the distance between these combinations and the center coded as $\mathbf{0}$ is $\sqrt{k}$). Such extreme combinations may lead to nonrealistic simulation outputs; an example is the ecological case study in Chap. 4 on screening. Actually, Voelkel (2005) assumes $2k$ input combinations instead of only $k + 1$ combinations. Frey and Wang (2006) recommends one-at-a-time designs if the goal of the experiment is "to seek improvements in the performance", which is closely related to the goal called "optimization" in the preceding chapter (see Sect. 1.2). Frey and Wang (2006) assumes small experimental error σ_w^2 and large two-factor interactions. One-at-a-time designs are also reviewed in Alaeddini et al. (2013).

Let us examine one more example; namely, a 2^k design with $k = 3$ inputs.

Example 2.3 *Obviously, a* 2^k *design with* $k = 3$ *inputs has an* 8×3 *design matrix* $\mathbf{D}$*. We use the notation that is conventional in DOE; i.e., we display only the signs of the elements of* $\mathbf{D}$ *so* $-$ *means* -1 *and* $+$ *means* $+1$*:*

$$\mathbf{D} = \begin{bmatrix} - & - & - \\ + & - & - \\ - & + & - \\ + & + & - \\ - & - & + \\ + & - & + \\ - & + & + \\ + & + & + \end{bmatrix}.$$

Because of the intercept β_0*, the matrix of explanatory variables* $\mathbf{X}$ *adds the column* $\mathbf{1}_8 = (1, \ldots, 1)'$*with eight elements equal to* 1*, to* $\mathbf{D}$ *so* $\mathbf{X} = [\mathbf{1}_8, \mathbf{D}]$*.*

It is easy to verify that the columns of this $\mathbf{X}$ *are orthogonal. Furthermore, the design* $\mathbf{D}$ *is balanced; i.e., each column has the same number of pluses and minuses; namely,* $2^{k-1} = 4$*. We may use orthogonality and balance to have the computer check for typos in* $\mathbf{D}$ *and* $\mathbf{X}$*.*

In general, a 2^k design results in an *orthogonal* matrix of independent variables for the first-order polynomial in Eq. (2.38):

$$\mathbf{X}'\mathbf{X} = n\mathbf{I} \quad \text{with} \quad n = 2^k \tag{2.43}$$

where $\mathbf{I}$ denotes an $n \times n$ identity matrix. This orthogonality property follows directly from the following general procedure for constructing a 2^k design (also see the preceding example with $k = 3$):

1. Make the first 2 elements of column 1 of $\mathbf{D}$ equal to $(-1, +1)'$; repeat these two elements, until the column is filled with $n = 2^k$ elements.
2. Make the first 2^2 elements of column 2 equal to $(-1, -1, +1, +1)'$; repeat these 2^2 elements, until this column is filled.
3. Make the first 2^3 elements of column 3 equal to $(-1, -1, -1, -1, +1, +1, +1, +1)'$; repeat these 2^3 elements, until this column is filled.
4. ...
5. Make the first 2^{k-1} elements of column k equal to 2^{k-1} consecutive elements -1, followed by 2^{k-1} consecutive elements $+1$.

Note: Orthogonal matrixes are related to so-called *Hadamard* matrixes; see Craigen (1996). The orthogonality property in Eq. (2.43) simplifies the LS estimator; i.e., substituting Eq. (2.43) into Eq. (2.13) gives

$$\hat{\boldsymbol{\beta}} = (n\mathbf{I})^{-1}\mathbf{X}'\mathbf{w} = \mathbf{X}'\mathbf{w}/n = (\mathbf{x}_j\mathbf{w}/n) = \left(\frac{\sum_{i=1}^{n} x_{i;j}w_i}{n}\right) \; (j = 1, \ldots q). \tag{2.44}$$

This equation does not require matrix inversion. Avoiding matrix inversion improves the numerical accuracy of the LS estimation (numerical inaccuracy may be a major problem in Kriging metamodels, as we shall see in Chap. 5). Historically, avoiding matrix inversion was very useful when no computers were available, as was the case when DOE started seriously with Fisher (1935).

Obviously, 2^k designs are balanced; i.e., for each j, half the x_{ij} equals -1 and the other half equals $+1$. Consequently, the estimator $\widehat{\beta}_j$ is simply the difference between the two averages $\overline{w}_{1;j}$ denoting the average simulation output when input j is $+1$, and $\overline{w}_{2;j}$ denoting the average simulation output when factor j is -1:

$$\widehat{\beta}_j = \frac{\sum_{i=1}^{n} x_{ij}w_i/(n/2)}{2} = \frac{\overline{w}_{1;j} - \overline{w}_{2;j}}{2}. \tag{2.45}$$

So the mathematical criterion of LS gives an "intuitive" estimator.

Furthermore, the orthogonality property simplifies the covariance matrix in Eq. (2.18) to

$$\mathbf{\Sigma}_{\hat{\boldsymbol{\beta}}} = (n\mathbf{I})^{-1}\sigma_w^2 = \mathbf{I}\frac{\sigma_w^2}{n}. \tag{2.46}$$

So all the q estimators $\widehat{\beta}_j$ have the same variance σ_w^2/n, and they are statistically independent. Because the $\widehat{\beta}_j$ have the same estimated variances, we can rank $\widehat{\beta}_j$—in order of importance—using either these $\widehat{\beta}_j$ themselves or the t-values defined in Eq. (2.19). Because all q estimated effects are independent, the "full" regression model with q effects and the "reduced" model with nonsignificant effects eliminated have identical values for those estimated effects that occur in both models. If $\mathbf{X}$ is not orthogonal, then this so-called "backwards elimination" of nonsignificant effects changes the remaining estimates. Finally, it can be proven that the variances of $\widehat{\beta}_j$—the elements on the main diagonal in Eq. (2.18)—are minimal if $\mathbf{X}$ is orthogonal; see Box (1952).

Altogether, 2^k designs have many attractive properties. Unfortunately, the number of combinations is $n = 2^k$, so n grows exponentially with the number of inputs k. At the same time, the number of effects is only $q = k+1$ in a first-order polynomial metamodel, so 2^k designs become inefficient for high values of k; e.g., $k = 7$ gives $n = 2^7 = 128$ whereas $q = 8$. Therefore we now present designs that require only a fraction of these 2^k combinations.

Definition 2.10 *An incomplete design has fewer combinations than the corresponding full factorial design.*

This definition deserves the following comments:

- The simplest incomplete designs are 2^{k-p} designs, which are a fraction 2^{-p} of the 2^k design. For example, if $k = 7$, then a 2^{7-4} design with only $n = 8$ combinations suffices to fit a first-order polynomial. Details will follow in Sect. 2.4.

- There are also fractions of *mixed-level* designs such as $2^{k_1}3^{k_2}$ designs. These designs are rather complicated, and are hardly ever applied in simulation. We shall briefly discuss such designs in Sect. 2.10.

2.4 Designs for First-Order Polynomials: Resolution-III

Definition 2.11 *A resolution-III (R-III) design gives unbiased estimators of the parameters of a first-order polynomial, assuming such a polynomial is a valid metamodel.*

We provide the following comments on this definition.

- This definition goes back to the definition in Box and Hunter (1961a).

- These designs are also known as *Plackett-Burman* designs, originally published in Plackett and Burman (1946).

Combination	**1**	**2**	**3** = **1.2**
1	−	−	+
2	+	−	−
3	−	+	−
4	+	+	+

TABLE 2.1. A fractional factorial two-level design for three inputs with generator 3 = 1.2

- A subclass of Plackett-Burman designs are *fractional factorial two-level* or 2^{k-p} designs with positive integer p such that $p < k$ and $2^{k-p} \geq q$ where q denotes the number of parameters in the metamodel—or in a more explicit notation 2_{III}^{k-p} designs. Obviously, 2^{k-p} designs have n (number of combinations) equal to a power of two. More general, Plackett-Burman designs have n equal to a multiple of four and at least equal to $k+1$; e.g., for $8 \leq k \leq 11$ the Plackett-Burman design has $n = 12$. First we discuss 2_{III}^{k-p} designs in Sect. 2.4.1; next we discuss general Plackett-Burman designs in Sect. 2.4.2.

2.4.1 2^{k-p} Designs of Resolution-III

Let us start with the simplest example of a 2_{III}^{k-p} design with $0 < p < k$; namely, a design with $k = 3$ ($k = 2$ and $p = 1$ would imply $n = 2^{2-1} = 2 < q = k + 1 = 3$ so a 2^{2-1} design does not have resolution III). A full-factorial 2^3 design would require $n = 8$ combinations; see again Example 2.3. However, the number of parameters is only $q = k + 1 = 4$. Therefore a 2^{3-1} design suffices to estimate the $q = 4$ parameters; this 2_{III}^{3-1} design requires only $n = 4$ combinations. Table 2.1 gives a 2_{III}^{3-1} design, which we now discuss in detail. Its heading "Combination" stands for "input combination". The symbol **1** stands for the column $(x_{1;1}, \ldots, x_{n;1})'$ for input 1 where in this example $n = 4$. Likewise, **2** stands for the column for input 2. The heading **3** stands for input 3 and **3** = **1.2** for $x_{i;3} = x_{i;1}x_{i;2}$ with $i = 1, \ldots, n$, so the first element ($i = 1$) in the last column is $x_{1;3} = x_{1;1}x_{1;2} = (-1)(-1) = +1$ so the entry is a plus (+). The DOE literature calls "**3** = **1.2**" a design *generator*; we will discuss generators in more detail, when discussing the 2_{III}^{7-4} design in Table 2.3.

It is easy to verify that Table 2.1 gives an orthogonal **X**. The design is also balanced; i.e., each column of **D** in Table 2.1 has two minuses and two pluses.

Figure 2.4 gives a *geometric* presentation of the design in Table 2.1. This plot has the following property: each combination corresponds with a vertex that cannot be reached via traversing only one edge of the cube.

Combination	**1**	**2**	**3** $= -$**1.2**
1	$-$	$-$	$-$
2	$+$	$-$	$+$
3	$-$	$+$	$+$
4	$+$	$+$	$-$

TABLE 2.2. A fractional-factorial two-level design for three factors with generator 3 = -1.2

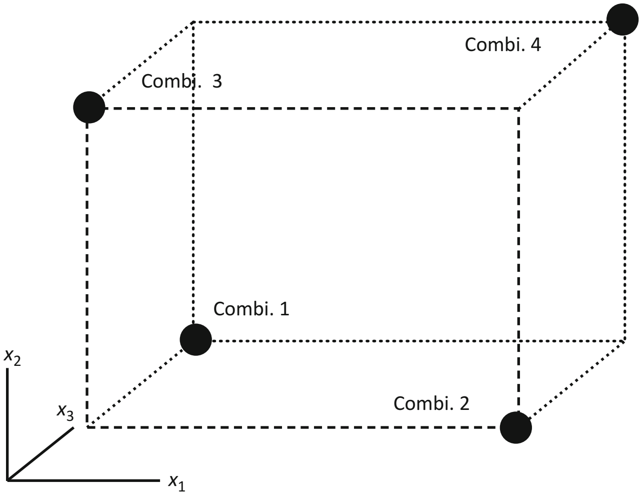


FIGURE 2.4. The fractional factorial two-level design for three inputs with generator **3** = **1.2**

Actually, the design in Table 2.1 is only one of the two possible 2^{3-1}_{III} designs; the other 2^{3-1}_{III} design is displayed in Table 2.2. It is straightforward to verify that this design is also balanced and gives an orthogonal **X**.

The two designs specified in Tables 2.1 and 2.2 belong to the same *family*. In this simple example with $k = 3$, these two designs together form the full factorial design that was listed in Example 2.3; i.e., the "dots" in Fig. 2.4 represent the 2^{3-1}_{III} design with the generator $\mathbf{3} = \mathbf{1.2}$, but the corners without dots represent the 2^{3-1}_{III} design with $\mathbf{3} = -\mathbf{1.2}$. The choice between these two designs is *arbitrary* (random). The association between the three inputs and the three columns in the design is also arbitrary; e.g., input 1 may be associated with (say) column 3. The association between the original levels (l_j and u_j) and the $+$ and $-$ signs is also arbitrary; e.g., the highest value of an input may be associated with the minus sign. If the input is quantitative, then such an association may confuse some users so we do not recommend it.

Combination	$\mathbf{1}$	$\mathbf{2}$	$\mathbf{3}$	$\mathbf{4 = 1.2}$	$\mathbf{5 = 1.3}$	$\mathbf{6 = 2.3}$	$\mathbf{7 = 1.2.3}$
1	$-$	$-$	$-$	$+$	$+$	$+$	$-$
2	$+$	$-$	$-$	$-$	$-$	$+$	$+$
3	$-$	$+$	$-$	$-$	$+$	$-$	$+$
4	$+$	$+$	$-$	$+$	$-$	$-$	$-$
5	$-$	$-$	$+$	$+$	$-$	$-$	$+$
6	$+$	$-$	$+$	$-$	$+$	$-$	$-$
7	$-$	$+$	$+$	$-$	$-$	$+$	$-$
8	$+$	$+$	$+$	$+$	$+$	$+$	$+$

TABLE 2.3. A one-sixteenth fractional factorial design for seven inputs

Now we continue with another example of a 2_{III}^{k-p} design; namely, a design with $n = 8$ combinations (the preceding example had $n = 4$). The number of inputs follows from $2^{k-p} = 8$ so $k - p = 3$ with positive integers k and p such that $0 < p < k$ and $2^{k-p} > k$ because $n \geq q = 1 + k$. A solution is $k = 7$ and $p = 4$. This gives Table 2.3, which is the analogue of Table 2.1. It is again easy to check that this design gives an orthogonal $\mathbf{X}$, and it is balanced (each column has $2^{7-5} = 4$ minuses and 4 pluses).

The design in Table 2.3 belongs to a bigger *family*. This family is formed by substituting a minus sign for the (implicit) plus sign in one or more generators; e.g., substituting $\mathbf{4} = -\mathbf{1.2}$ for $\mathbf{4} = \mathbf{1.2}$ in Table 2.3 gives one other member of the family. All the $2^7/2^{7-4} = 16$ family members together form the 2^7 design, which is the unique full-factorial two-level design.

Table 2.3 gives a so-called *saturated* design for seven inputs; Tables 2.1 and 2.2 gave saturated designs for three inputs.

Definition 2.12 *A saturated design has as many combinations as the number of parameters to be estimated.*

This definition leads to the following comments.

- In symbols, the definition means $n = q$ in Eq. (2.10).
- Hence, no degrees of freedom are left for the MSR in Eq. (2.20), so MSR is not defined and the lack-of-fit F-test in Eq. (2.31) cannot be applied. This problem can be easily solved: randomly select one or more combinations from another member of the family, and simulate this combination; if the inputs are quantitative, then simulate the center point $\mathbf{x} = \mathbf{0}$.

After our discussion of the 2_{III}^{3-1} and the 2_{III}^{7-4} designs, we now consider *intermediate* k values; namely, $4 \leq k \leq 6$. We can still use Table 2.3; i.e., for $k = 4$ we delete three columns (e.g., the last three columns), for $k = 5$ we delete two columns, and for $k = 6$ we delete one column. Obviously, the resulting designs are not saturated. Of course, we may also add one or more extra inputs to our original list with $4 \leq k \leq 6$ inputs; these extra inputs do not require a bigger experiment; i.e., n remains eight.

Our next example (after Table 2.1 with $n = 4$ and Table 2.3 with $n = 8$) has $n = 2^{k-p} = 16$ input combinations. So a saturated design implies $k = 15$ inputs. Hence $k - p = 4$ implies $p = 15 - 4 = 11$. We may construct this 2^{15-11} design through the following simple algorithm.

Algorithm 2.1

1. *Construct the* 2^4 *design for the first four inputs, and obtain a* 16×4 *design matrix.*
   Comment: The 2^4 *design is a full-factorial two-level design with* $k = 4$.

2. *Add all* $4 \times (4 - 1)/2 = 6$ *pairwise generators* $\mathbf{5} = \mathbf{1.2}$, $\mathbf{6} = \mathbf{1.3}$, $\mathbf{7} = \mathbf{1.4}$, …, $\mathbf{10} = \mathbf{3.4}$.

3. *Add all four triplet generators* $\mathbf{11} = \mathbf{1.2.3}$, $\mathbf{12} = \mathbf{1.2.4}$, $\mathbf{13} = \mathbf{1.3.4}$, $\mathbf{14} = \mathbf{2.3.4}$.

4. *Add the single quadruple generator,* $\mathbf{15} = \mathbf{1.2.3.4}$.

Exercise 2.9 *Specify the design that follows from this algorithm.*

Obviously, the design that follows from this algorithm is only one of the members of the *family* with $2^{15} = 32{,}768$ members that can be generated through the addition of one or more minus signs to one or more generators in Algorithm 2.1.

Our final example of a 2^{k-p}_{III} design has $n = 32$ combinations (the preceding examples had $n = 4, 8, 16$). Obviously, a saturated design with $n = 32$ implies $k = 31$. Hence $k - p = 5$ so $2^5 = 32$. This implies $p = 31 - 5 = 26$. The construction of this 2^{31-26} design remains quite simple, but tedious. A computerized algorithm is then helpful. To check the computed results, we recommend to verify the orthogonality and balance of the resulting design. It is simple to write such an algorithm.

Note: A different algorithm with so-called *Walsh functions* is applied in Sanchez and Sanchez (2005, p. 366).

We do not discuss 2^{k-p}_{III} designs with higher k values, because in practice such high k values are rare—except for some military simulations discussed in Oh et al. (2009). One explanation is the psychological argument originally formulated in Miller (1956); namely, a human's capacity for processing information is limited to seven plus or minus two inputs. In simulation experiments, this argument implies that 2^{k-p}_{III} designs with $k > 9$ enable

Combination	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
1	+	−	+	−	−	−	+	+	+	−	+
2	+	+	−	+	−	−	−	+	+	+	−
3	−	+	+	−	+	−	−	−	+	+	+
4	+	−	+	+	−	+	−	−	−	+	+
5	+	+	−	+	+	−	+	−	−	−	+
6	+	+	+	−	+	+	−	+	−	−	−
7	−	+	+	+	+	+	+	−	+	−	−
8	−	−	+	+	−	−	+	+	−	+	−
9	−	−	−	+	+	+	−	+	+	−	+
10	+	−	−	−	+	+	+	−	+	+	−
11	−	+	−	−	−	+	+	+	−	+	+
12	−	−	−	−	−	−	−	−	−	−	−

TABLE 2.4. The Plackett-Burman design for eleven inputs

the estimation of all k first-order effects β_j ($j = 1, \ldots, k$), but the estimates $\widehat{\beta}_j$ are used only to estimate the 7 ± 2 most important inputs. In Chap. 4 on screening we shall discuss designs that are more efficient than 2_{III}^{k-p} designs, provided we know the signs of the first-order effects. Different computer procedures for the construction of 2_{III}^{k-p} designs with high k values are presented in Ryan and Bulutoglu (2010) and Shrivastava and Ding (2010).

2.4.2 Plackett-Burman Designs of Resolution-III

As we have already mentioned in the beginning of Sect. 2.4, Plackett-Burman designs include 2_{III}^{k-p} designs. So we may define Plackett-Burman designs *in the narrow sense* as R-III designs that have a number of combinations that is a multiple of four but not a power of two. Actually, Plackett and Burman (1946) lists such designs for $12 \leq n \leq 96$; for $12 \leq n \leq 36$ these designs are reproduced in Montgomery (2009, p. 326) and Myers et al. (2009, pp. 165). For simulation practice, we display a Plackett-Burman design in the narrow sense in Table 2.4, which has $n = 12$ combinations of $k = 11$ inputs. Plackett-Burman designs are again balanced and orthogonal.

Exercise 2.10 *Use a R-III design to experiment with a simulation model of your own choice, provided this model enables you to experiment with (say) between five and twenty inputs. Select the ranges of these inputs so "small" (e.g., 1 % changes from the base values) that you may assume a first-order polynomial is a valid metamodel. If the simulation model is random, then simulate (say) five replications. Estimate the first-order ef-*

fects of these inputs, using the standardized and the original input values, respectively. Test whether these effects are significantly different from zero. Give a list that sorts the inputs in their order of importance.

2.5 Linear Regression: Interactions

Definition 2.13 *Interaction means that the effect of one input depends on the levels of one or more other inputs.*

Let us consider the simplest example; namely, only two inputs, a first-order polynomial augmented with the interaction between these two inputs, and white noise so $e \sim \mathrm{NIID}(0, \sigma_e^2)$:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{1;2} x_1 x_2 + e. \tag{2.47}$$

This equation implies $\partial y / \partial x_1 = \beta_1 + \beta_{1;2} x_2$, so the effect of x_1 indeed depends on x_2. In geometric terms, interaction means that the response curves for $y(x_1|x_2)$ are not parallel for different values of x_2; see Fig. 2.5, which uses the standardized values -1 and $+1$ for the two inputs. Obviously, interaction is also defined for deterministic simulation models, which imply that e vanishes in Eq. (2.47) so the E operator in Fig. 2.5 becomes redundant. If interactions are important, then the relative importance of an input is not measured by (the absolute value of) its first-order effect only.

Note: In metamodels that are more general than Eq. (2.47), we have $\partial y / \partial x_j = f(x_{j'})$ with $j \neq j'$. If the residual e is not white noise but has a variance that depends on the input combination $\mathbf{x} = (x_1, \ldots, x_k)'$, then interaction between x_j and $x_{j'}$ may imply the effect of the other input $x_{j'}$. In Sect. 5.8 we shall discuss a generalized definition of interactions in nonlinear metamodels such as Kriging models, and their analysis through *functional analysis of variance* (FANOVA) using so-called *Sobol indexes*.

If the interaction between two inputs is positive, then the inputs are called *complementary*. A classic example is a pair of shoes; i.e., obtaining more shoes for the left foot gives higher utility only if more shoes for the right foot are also obtained. If the interaction is negative, then the inputs are *substitutes* for each other. A classic example is provided by butter and margarine.

In the general case of k inputs, we may augment the first-order polynomial in Eq. (2.38) with the interactions between all pairs of inputs k and k' with $k \neq k'$:

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k-1} \sum_{j'=j+1}^{k} \beta_{j;j'} x_j x_{j'} + e \tag{2.48}$$

where $\beta_{j;j'}$ is called the two-factor interaction between the inputs j and j'; $\beta_{j;j'}$ is also called the two-way or pairwise interaction, or the cross-product. It is easy to prove that the total number of two-factor interactions

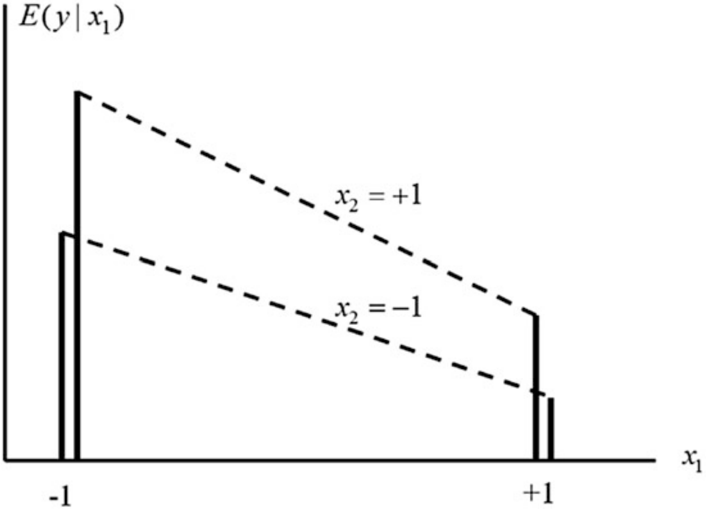


FIGURE 2.5. Interaction between two inputs x_1 and x_2, in a first-order metamodel with output y

in this equation is $k(k-1)/2$, so the total number of parameters is $q = 1 + k + k(k-1)2 = 1 + k(k+1)/2$. It is easy to see that the $N \times q$ matrix of independent variables $\mathbf{X}$ follows from the $n \times k$ design matrix $\mathbf{D}$ and the n-dimensional vector $\mathbf{m}$ with the number of replications for combination i $(i = 1, \ldots, n)$ so $\mathbf{m} = (m_1, \ldots, m_n)'$:

$$\mathbf{X} = (\mathbf{x}_i) = (1, d_{i;1}, \ldots, d_{i;k}, d_{i;1}d_{i;2}, \ldots, d_{i;k-1}d_{i;k}) \; (i = 1, \ldots, N) \quad (2.49)$$

with $N = \sum_{i=1}^{n} m_i$ where m_i is a positive integer, possibly 1; see again Eq. (2.24). If the simulation output is an average, then a single replication gives an unbiased estimator of the expected simulation output; however, an estimated quantile requires $m \gg 1$ replications. Obviously, in deterministic simulation we have $m_i = 1$ so $N = n$. We shall further discuss the selection of the number of replications, later in this chapter and in the following chapters.

In the following example a first-order polynomial does not give a valid metamodel, but augmenting this polynomial with two-factor interactions does give an adequate approximation.

Example 2.4 *Kleijnen and Standridge (1988) studies a deterministic simulation model of a flexible manufacturing system (FMS). The machine mix for this FMS is determined by the input combination* $\mathbf{d}$ *of the simulation experiment. This* $\mathbf{d}$ *determines the original inputs* z_1*,* z_2*, and* z_3 *that denote the number of machines performing operation #1, #2, and #3, and* z_4 *that denotes the number of "flexible" machines or robots capable of performing any of these three operations. The experimental area is defined by the following constraints:* $5 \leq z_1 \leq 6$*,* $1 \leq z_2 \leq 2$*,* $2 \leq z_3 \leq 3$*, and* $0 \leq z_4 \leq 2$*. This domain is quite small, so a first-order polynomial may result in a valid metamodel. Originally, an incomplete design with* $n = 8$ *combinations is intuitively specified. Next a* 2^{4-1} *design is specified; this design has the same number of combinations* $n = 8$*; see Table 2.3 with the last three columns deleted so the generator is* $\mathbf{4} = \mathbf{1}.\mathbf{2}$*. Both designs give I/O data that allow the fitting of first-order polynomials using LS; see Eq. (2.38) with* $k = 4$*. Kleijnen and Standridge (1988) ignores the fact that the fitting error* e *is not white noise, and applies classic regression analysis. Because the original scales are used instead of the standardized scales, the* 2^{4-1} *design does not give constant estimated variances for the estimated regression parameters. The intuitive design gives bigger estimated variances for the estimated regression parameters; e.g., the estimated variance for the estimated effect of* z_4 *is nearly four times higher. Further analysis of the fitted metamodel—based on the data from the* 2^{4-1} *design—suggests that the first-order polynomial is not adequate, and that the effects of* z_1 *and* z_3 *are negligible (this analysis uses cross-validation, which we shall discuss in Sect. 3.6.2). So next, a first-order polynomial is fitted for the remaining two inputs* z_2 *and* z_4 *and their interaction; see Eq. (2.47). This metamodel is fitted to the "old" I/O data resulting from the* 2^{4-1} *design. Further analysis suggests that the resulting metamodel is valid. This metamodel implies that the machines in groups #2 and #4 are the bottlenecks of the FMS, and—because the estimated interaction turns out to be negative—that machine group #4 (the robots) can serve as a substitute for machine group #2.*

This example demonstrates the usefulness of first-order polynomials augmented with two-factor interactions. The DOE literature also uses higher-order interactions, e.g., three-factor interactions:

$$\begin{aligned} y = \beta_0 + \sum_{j=1}^{k}\beta_j x_j + \sum_{j=1}^{k-1}\sum_{j'=j+1}^{k} \beta_{j;j'} x_j x_{j'} \\ + \sum_{j=1}^{k-2}\sum_{j'=j+1}^{k-1}\sum_{j''=j'+1}^{k} \beta_{j;j';j''} x_j x_{j'} x_{j''} + e. \end{aligned} \tag{2.50}$$

We do not give the definition of these high-order interactions, for two reasons:

1. High-order interactions are hard to interpret, so these interactions are difficult to explain to the simulation users.

2. High-order interactions are often unimportant in practice.

Throughout this book, we assume that interactions among three or more inputs are unimportant. Of course, this assumption should be checked; see the "lack-of-fit" and "validation" of metamodels discussed throughout this book. A counterexample is Ekren and Ornek (2008), discussing a simulation model of a manufacturing system that gives a metamodel with significant three-factor interactions among a few factors.

2.6 Designs Allowing Two-Factor Interactions: Resolution-IV

Definition 2.14 *A resolution-IV (R-IV) design gives unbiased estimators of the parameters of a first-order polynomial, even if two-factor interactions are nonzero; all other effects are assumed to be zero.*

Box and Wilson (1951) includes a proof of the so-called *foldover theorem*, which we briefly formulate as follows:

Theorem 2.1 *If a R-III design* $\mathbf{D}$ *is augmented with its so-called mirror design* $-\mathbf{D}$*, then the resulting design is a R-IV design.*

So the price for augmenting a R-III design to a R-IV design is the doubling of the number of combinations. We give some examples.

Example 2.5 *Table 2.1 gave the* 2_{III}^{3-1} *design with the generator* $\mathbf{3} = \mathbf{1}.\mathbf{2}$*. The mirrored design was shown in Table 2.2, which is the* 2_{III}^{3-1} *design with the generator* $\mathbf{3} = -\mathbf{1}.\mathbf{2}$*. Combining these two designs into a single design gives a* 2^3 *design. This design results in* $\mathbf{X}$*, which has* $n = 8$ *rows and* $q = 1 + 3(3 + 1)/2 = 7$ *columns that correspond with the intercept, the three first-order effects, and the three two-factor interactions. Because all these columns are orthogonal,* $\mathbf{X}$ *is certainly not collinear so LS estimation is possible. The* $q = 7$ *estimators leave* $n - q = 8 - 7 = 1$ *degree of freedom, which could be used to estimate the three-factor interaction; see Eq. (2.50) with* $k = 3$*. However, if we assume that this high-order interaction is zero, then we can use this degree of freedom to estimate the common variance* $\sigma_w^2 = \sigma_y^2 = \sigma_e^2$ *through MSR defined in Eq. (2.20).*

The following example demonstrates that adding the mirror design gives unbiased estimators of the first-order effects, but does not always enable unbiased estimators of the individual two-factor interactions.

Example 2.6 *Table 2.3 gave a 2_{III}^{7-4} design. Combining this design with its mirrored design gives a design with $n = 16$ combinations; namely, a 2_{IV}^{7-3} design, as we shall see below.* **X** *follows from Eq. (2.48) with $k = 3$; i.e.,* **X** *has $n = 16$ rows and $q = 1 + 7(7 + 1)/2 = 29$ columns so $n < q$, which implies that* **X** *is collinear. Hence, LS estimation of the 29 individual regression parameters is impossible. However, it is possible to compute the LS estimator of the intercept and the seven first-order effects; see the next exercise.*

Exercise 2.11 *Derive* **X** *for the intercept and the seven first-order effects, using the combined design in Example 2.6. Check that—for example—the column for the interaction between the inputs 6 and 7 is balanced and orthogonal to the columns for the first-order effects of the inputs 6 and 7.*

The construction of R-IV designs is easy, once a R-III design is available; i.e., we simply augment a $\mathbf{D}_{III}$ (Plackett-Burman) design with its mirror design, denoted by $-\mathbf{D}_{III}$. For the Plackett-Burman subclass of $2_{III}^{(k-1)-p}$ designs, we may construct the 2_{IV}^{k-p} designs by first defining the full-factorial design in $k - p$ inputs, and then *aliasing* or *confounding* the remaining p inputs with high-order interactions among these first $k - p$ inputs; i.e., we use these interactions as generators (we shall return to aliasing at the end of this section; see Eq. (2.52)). For example, $k = 8$ and $n = 16 = 2^4$ leads to a 2^{8-4} design. So first we construct a 2^4 design in four inputs. Suppose we label these four inputs 1, 2, 3, and 4. Next, we may use the following generators: $\mathbf{5} = \mathbf{1.3.4}$, $\mathbf{6} = \mathbf{2.3.4}$, $\mathbf{7} = \mathbf{1.2.3}$, and $\mathbf{8} = \mathbf{1.2.4}$. It can be derived that the 28 two-factor interactions are confounded in seven groups of size four; see Kleijnen (1975, pp. 336–344) or Kleijnen (19857, pp. 303–305). In Appendix 2 we present some useful manipulations with generators, following the DOE literature.

Now we consider Plackett-Burman designs in the narrow sense, which do not have the simple confounding patterns of 2^{k-p} designs. The latter designs use design generators, which imply that a given column is identical to some other column of **X** when that **X** includes columns for all the interactions among these k inputs. Plackett-Burman designs in the narrow sense lead to an **X** that also has $q = 1 + k + k(k-1)/2$ columns. Applying linear algebra, we can prove that if $n < q$ then **X** is collinear. A R-IV design implies that the columns for the first-order effects and the intercept are orthogonal to the two-factor interaction columns, but the latter $k(k-1)/2$ columns are not necessarily mutually orthogonal or identical.

The R-IV designs discussed so far imply that the number of combinations increases with jumps of eight ($n_{IV} = 8, 16, 24, 32, 40, \ldots$), because the underlying R-III designs have a number of combinations that jump with four ($n_{III} = 4, 8, 12, 16, 20, \ldots$). However, Webb (1968) derives R-IV designs with n_{IV} increasing in smaller jumps; i.e., $n_{IV} = 2k$ where k does not need to be a multiple of four. Webb's designs also use the foldover theorem. Because we are not aware of any simulation applications, we refer to Kleijnen (1975, pp.344–348) for details of these designs and their analysis.

In practice, a single simulation run may require so much computer time that a R-IV design is hardly possible. In case of such expensive simulation, the following algorithm may help—but we have no experience with the application of this algorithm in practice.

Algorithm 2.2

1. Simulate all combinations of the R-III design.
2. Use the I/O data resulting from step 1, to estimate the first-order polynomial metamodel.
3. Use the metamodel resulting from step 2, to predict the simulation outputs of the mirror design of the R-III design.
   Comment: The original R-III design plus its mirror design form the R-IV design.
4. Initialize a counter (say) i: $i = 1$.
5. Simulate combination i of the mirror design.
6. Compare the metamodel prediction from step 3 and the simulation output from step 5; if the prediction error is not acceptable, then increase the counter to $i+1$ and return to step 5; else stop simulating.

We conclude this section on R-IV designs with a general discussion of *aliasing* or *confounding*. Assume that a valid linear regression metamodel is

$$y = \mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2 + e \tag{2.51}$$

where e denotes white noise. An example of this equation is an $\mathbf{X}_1$ corresponding with the intercept and the first-order effects collected in $\boldsymbol{\beta}_1$, and an $\mathbf{X}_2$ corresponding with the two-factor interactions $\boldsymbol{\beta}_2$. Suppose that we start with a tentative simple metamodel without these interactions. Then we estimate the first-order polynomial parameters through

$$\widehat{\boldsymbol{\beta}}_1 = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{w}. \tag{2.52}$$

So combining Eqs. (2.52) and (2.51) and assuming a valid metamodel is Eq. (2.51) so $E(w) = E(y)$ gives

$$\begin{aligned} E(\widehat{\boldsymbol{\beta}}_1) &= (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'E(\mathbf{w}) = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'(\mathbf{X}_1\boldsymbol{\beta}_1 + \mathbf{X}_2\boldsymbol{\beta}_2) \\ &= \boldsymbol{\beta}_1 + (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2\boldsymbol{\beta}_2. \end{aligned} \tag{2.53}$$

This equation includes the matrix (say) $\mathbf{A} = (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2$, which Box and Draper (1959) calls the *alias matrix*. Equation (2.53) implies an unbiased estimator of $\boldsymbol{\beta}_1$ if either $\boldsymbol{\beta}_2 = \mathbf{0}$ or $\mathbf{X}_1'\mathbf{X}_2 = \mathbf{0}$. Indeed, R-III designs assume that $\boldsymbol{\beta}_2 = \mathbf{0}$ where $\boldsymbol{\beta}_2$ consists of the two-factor interactions; R-IV designs ensure that $\mathbf{X}_1'\mathbf{X}_2 = \mathbf{0}$ (the two-factor interaction columns are orthogonal to the columns for the first-order effects and the intercept).

Note: In this section we discussed the construction of R-IV designs from R-III designs, using the foldover principle. However, instead of reversing the signs of *all* columns of the R-III design, we may reverse the signs of only one column or a few columns. Of course, the latter construction gives a different alias pattern compared with the former construction, and does not give a R-IV design. Detailed discussions of various foldover constructions for two-level fractional factorials are Elsawah and Qin (2015) and Li and Lin (2015).

2.7 Designs for Two-Factor Interactions: Resolution-V

Definition 2.15 *A resolution-V (R-V) design enables LS estimation of the first-order effects, the two-factor interactions, and the intercept; all other effects are assumed to be zero.*

Estimation of the individual two-factor interactions may be desirable, as Example 2.4 involving a FMS has already illustrated. In that example, the number of inputs was originally $k = 4$, but analysis of the I/O data of the original 2_{III}^{4-1} design resulted in elimination of two nonsignificant inputs; consequently, $k = 2$ and the original 2_{III}^{4-1} design gave a 2^2 design for these $k = 2$ significant inputs.

Let us consider a 2_{IV}^{8-4} design; such a design is derived in Appendix 2 and can also be found in the DOE literature. Obviously, it is impossible to compute the LS estimators of the $q = 1+8(8+1)/2 = 37$ regression parameters from only $n = 16$ combinations; LS estimation of these 37 parameters is possible from $n = 2^{8-2} = 64$ combinations—provided these combinations are selected correctly; again see Appendix 2. In general, the first-order polynomial augmented with all the two-factor interactions implies that q (number of parameters) becomes $1+k+k(k-1)/2 = (k^2+k)/2+1$, so the number of parameters is of order k^2; i.e., many more combinations need to be simulated compared with a first-order polynomial. Box and Hunter (1961b) includes a table—reproduced in Table 2.5—with generators for 2^{k-p} designs of resolution V and higher; the definition of a resolution higher than V is unimportant for DASE.

Note: Sanchez and Sanchez (2005) includes an algorithm for constructing R-V designs in case the number of inputs is very large; e.g., $k = 120$ leads to a $2_V^{120-105}$ design. Unfortunately, 2_V^{k-p} designs—except for the 2_V^{5-1} design (see Table 2.5)—require relatively many combinations to estimate the regression parameters; i.e., these designs are certainly not saturated. For example, the 2_{VI}^{9-2} design in Table 2.5 requires 128 combinations to estimate

k	n	Generators
5	$2_V^{5-1} = 16$	$\mathbf{5} = \mathbf{1.2.3.4}$
6	$2_{VI}^{6-1} = 32$	$\mathbf{6} = \mathbf{1.2.3.4.5}$
7	$2_{VII}^{7-1} = 64$	$\mathbf{7} = \mathbf{1.2.3.4.5.6}$
8	$2_V^{8-2} = 64$	$\mathbf{7} = \mathbf{1.2.3.4}$; $\mathbf{8} = \mathbf{1.2.5.6}$
9	$2_{VI}^{9-2} = 128$	$\mathbf{9} = \mathbf{1.4.5.7.8}$; $\mathbf{10} = \mathbf{2.4.6.7.8}$
10	$2_V^{10-3} = 128$	$\mathbf{8} = \mathbf{1.2.3.7}$; $\mathbf{9} = \mathbf{2.3.4.5}$; $\mathbf{10} = \mathbf{1.3.4.6}$
11	$2_V^{11-4} = 128$	See $k = 10$; $\mathbf{11} = \mathbf{1.2.3.4.5.6.7}$

TABLE 2.5. Generators for fractional-factorial two-level designs of resolution V, VI, and VII

Effect type	Generator
Intercept	$(-1, \ldots, -1)$ for all k inputs
First-order effect	$(-1, +1, \ldots, +1)$ for all k inputs
Two-factor interaction	$(1, 1, -1, \ldots, -1)$ for $k > 3$ inputs

TABLE 2.6. Generators for Rechtschaffner's resolution-V designs

$q = 1 + 9(9+1)/2 = 46$ parameters so its *efficiency* is only $46/128 = 0.36$; the $2_V^{120-105}$ design requires $n = 32{,}768$ whereas $q = 7{,}261$ so its efficiency is only $7{,}261/32{,}768 = 0.22$. There are R-V designs that require fewer runs. For example, Mee (2004) gives a design for 47 factors that requires 2,048 combinations so its efficiency is $1{,}129/2{,}048 = 0.55$, whereas Sanchez and Sanchez (2005) requires 4,096 combinations so its efficiency is 0.28. For further comparisons among these types of R-V designs, we refer to Sanchez and Sanchez (2005, pp. 372–373).

Actually, if a simulation run takes much computer time, then *saturated* designs are attractive (whereas the designs in Table 2.5 are not saturated). Rechtschaffner (1967) includes saturated fractions of two-level (and three-level) designs; see Table 2.6. Their construction is simple: the *generators* are permuted in the different input combinations; see the design for $k = 4$ inputs in Table 2.7. These designs are not orthogonal. Qu (2007) further investigates the statistical properties of Rechtschaffner's designs.

Exercise 2.12 *Compute the variances of the estimated regression parameters that result from the design in Table 2.7, assuming $\sigma_w^2 = 1$. What would these variances have been, had there been an orthogonal saturated R-V design for $k = 4$?*

Rechtschaffner's type of design is applied in the following example.

Example 2.7 *The Dutch OR Society organized a competition, challenging the participants to find the combination of $k = 6$ inputs that maximizes the output of a simulated system. This challenge was accepted by twelve teams*

Combination	Generator	**1**	**2**	**3**	**4**
1	$(-1, \ldots, -1)$	-1	-1	-1	-1
2	$(-1, +1, \ldots, +1)$	-1	$+1$	$+1$	$+1$
3		$+1$	-1	$+1$	$+1$
4		$+1$	$+1$	-1	$+1$
5		$+1$	$+1$	$+1$	-1
6	$(+1, +1, -1, \ldots, -1)$	$+1$	$+1$	-1	-1
7		$+1$	-1	$+1$	-1
8		$+1$	-1	-1	$+1$
9		-1	$+1$	$+1$	-1
10		-1	$+1$	-1	$+1$
11		-1	-1	$+1$	$+1$

TABLE 2.7. Rechtschaffner's design for four inputs

from academia and industry. Because each team was allowed to run only 32 combinations, Kleijnen and Pala (1999, Table 1) uses Rechtschaffner's saturated R-V design; so the number of combinations is $n = 1 + 6 + 6\,(6-1)/2 = 22$.

2.8 Linear Regression: Second-Order Polynomials

A second-order polynomial may be a better metamodel as the experimental area of the simulation experiment gets bigger or the I/O function of the underlying simulation model gets more complicated; see the Taylor series expansion of a function about a point given by a specific input combination. An example is the M/M/1 simulation, in which—for higher traffic rates x—a better metamodel than the first-order polynomial defined in Eq. (2.8) seems

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + e. \tag{2.54}$$

Obviously, estimation of the three parameters in Eq. (2.54) requires the simulation of at least three input values. Indeed, practitioners often use a one-at-a-time design with three values per input (they even do so, when fitting a first-order polynomial; Example 2.2 showed that such a design is inferior compared with a factorial design). DOE also provides designs with three values per input; e.g., 3^k designs. However, more popular in simulation are *central composite designs* (CCDs), which usually have five values per input; see Sect. 2.9 below.

We emphasize that second-order polynomials such as Eq. (2.54) are nonlinear in $\mathbf{x}$ (independent regression variables) but linear in β (regression parameters). Consequently, second-order polynomial metamodels remain linear regression models, which were specified in Eq. (2.10).

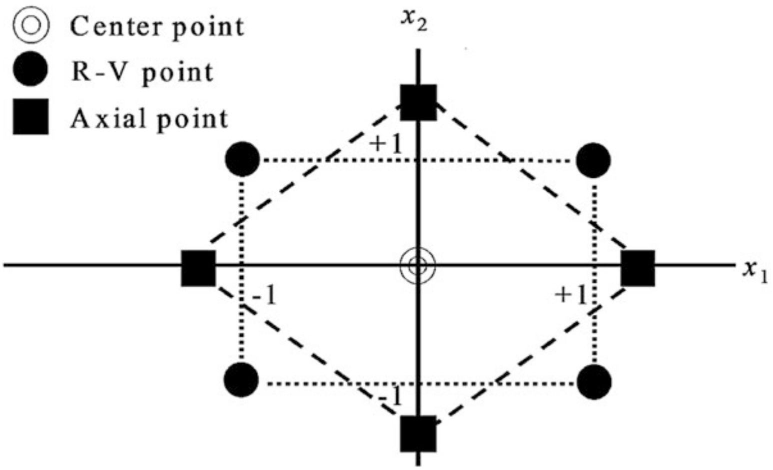


FIGURE 2.6. A CCD for two inputs

The general second-order polynomial metamodel in k factors is

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k} \sum_{j' \geq j}^{k} \beta_{j;j'} x_j x_{j'} + e. \tag{2.55}$$

Obviously, this metamodel adds k *purely quadratic* effects $\beta_{j;j}$ to Eq. (2.48); consequently, q (number of effects) becomes $(k + 1)(k + 2)/2$. Substitution of the linear transformation defined in Eq. (2.39) into Eq. (2.55) gives the metamodel in the original input values. The purely quadratic effects $\beta_{j;j}$ quantify diminishing or increasing rates of return. In practice, second-order polynomials are applied either locally or globally. *Local* fitting may be used when searching for the optimum input combination; an example is Example 2.7. We shall return to simulation optimization in Chap. 6. *Global* fitting (e.g., an M/M/1 queueing model with a traffic rate x such that $0 < x < 1$) using second-order polynomials has indeed been applied, but in general Kriging provides better metamodels; see Chap. 5.

2.9 Designs for Second-Degree Polynomials: Central Composite Designs

A CCD enables LS estimation of all the effects in a second-order polynomial, assuming all effects of higher order are zero. More precisely, a CCD augments a R-V design such that the purely quadratic effects can also be estimated. Figure 2.6 gives a possible CCD for $k = 2$ standardized inputs denoted by x_1 and x_2. In general, a CCD consists of the following combinations of the standardized inputs:

- a R-V design or a design of higher resolution (see Sect. 2.7);
- the *central* point $(0, \ldots 0)'$;
- the $2k$ *axial* points, which form a so-called *star design*, where the "positive" axial point for input j ($j = 1, \ldots, k$) is $x_j = c$ while all other $(k-1)$ inputs are fixed at the center so $x_{j'} = 0$ ($j' = 1, \ldots, k$ and $j' \neq j$), and the "negative" axial point for input j is $x_j = -c$ and $x_{j'} = 0$.

If we select the appropriate value c for the axial points, then we obtain a so-called *rotatable* design; i.e., if e is white noise, then the CCD gives a constant variance for the predicted output at a fixed distance from the origin (so the contour functions for these variances are circles). Such a rotatable design requires $c = n_V^{1/4}$ where n_V denotes the number of combinations in the R-V fractional factorial design that is part of the CCD; see Myers et al. (2009, pp. 307). Obviously, if $c \neq 1$, then a CCD has five values per input; if $c = 1$, then a CCD has only three values per input.

A CCD does not give an orthogonal $\mathbf{X}$, so the estimated parameters of the second-degree polynomial are correlated. Letting n_{CCD} denote the total number of combinations in a CCD, we obtain $n_{\text{CCD}} = n_V + 1 + 2k$; e.g., Fig. 2.6 with $k = 2$ implies $n_{\text{CCD}} = 2^2 + 1 + 2 \times 2 = 9$. (For $k = 120$, the design in Sanchez and Sanchez (2005) implies $n_{\text{CCD}} = 32{,}768 + 1 + 2 \times 120 = 33{,}009$.) Most experiments with real systems or random simulation models replicate only the central point, to estimate the common variance σ_e^2 and to compute the lack-of-fit F-statistic defined in Eq. (2.31). For further discussion of CCDs, we refer to Myers et al. (2009, pp. 296–317) and to NIST/SEMATECH's e-handbook of statistical methods on the website

`http://www.itl.nist.gov/div898/handbook/`

Exercise 2.13 *By definition, a rotatable CCD gives a constant variance for the predicted output at a given distance from the origin. Will this constant variance increase or decrease as the output is predicted at a distance farther away from the origin?*

CCDs are rather inefficient because they use inefficient R-V designs and add $2k$ axial points so—together with the center point—CCDs use five (or three if $c = 1$) values per input. Therefore, Example 2.7 simulates only half of the star design; e.g., if the better outputs seem to lie in the southwestern corner of Fig. 2.6, then it is efficient to simulate only the two points $(-c, 0)'$ and $(0, -c)'$. We have already emphasized that classic R-V designs are very inefficient, so we prefer Rechtschaffner's saturated designs. Kleijnen (19857, pp. 314–316) presents three other types of saturated designs for second-order polynomials; namely, Koshall, Scheffé, and Notz designs. Furthermore, Draper and Lin (1990) also presents small designs for such polynomials. More designs for second-order polynomials are surveyed in Barton and Meckesheimer (2006) and Khuri and Mukhopadhyay (2010). However, we are not aware of any simulation applications of these designs.

Exercise 2.14 *Select a model with a known unconstrained optimum in your favorite literature (e.g., the Operations Research/Management Science literature on inventory management). Fit a second-order polynomial in the neighborhood of the true optimum, using the standardized and the original input values, respectively. To fit this polynomial, use a design that enables unbiased estimation of all the coefficients of this polynomial; e.g., a CCD with axial points with a standardized value equal to $c = n_V^{1/4}$. Replicate only the center point of this design $m > 1$ times. Next estimate the optimal input and output of this simulation model, using the fitted polynomial with standardized and original values, respectively. Furthermore, you should estimate the optimal input and output using the full and the reduced metamodel, respectively, where the reduced model eliminates all nonsignificant effects in the full model—except for those nonsignificant effects that involve inputs that have significant higher-order effects; e.g., if the estimated main effect $\widehat{\beta}_1$ is not significant, but $\widehat{\beta}_{1;2}$ is, then $\widehat{\beta}_1$ is not set to zero (see the heredity assumption in Wu and Hamada (2009)). Check whether the estimated optimal input combination lies inside the experimental area.*

2.10 Optimal Designs and Other Designs

In this section we shall discuss various optimality criteria for selecting a design, and we shall mention some more design types besides the designs we discussed in the preceding sections; namely, two-level designs of resolution III, IV, and V and the CCDs.

2.10.1 Optimal Designs

Below Eq. (2.46) we mentioned that Box (1952) proves that the variances of $\widehat{\beta}_j$ with $j = 1, \ldots, q$ are minimal if $\mathbf{X}$ is orthogonal. Now we might wonder whether orthogonal designs are "optimal"; consequently, we might wonder whether nonorthogonal CCDs are not optimal. However, this raises the question: what is an *optimal* design? The DOE literature discusses the following optimality criteria, which include the so-called *alphabetic optimality* criteria (A, D, and G).

- *A-optimality*: minimize the *trace* of $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}}$. Obviously, this criterion is related to minimizing the individual variances of the estimated regression parameters, $\mathrm{Var}(\widehat{\beta}_j)$. The A-optimality criterion neglects the off-diagonal elements of $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}}$; these elements are incorporated in the following criterion.

- *D-optimality*: minimize the determinant of $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}}$.
- *G-optimality:* minimize the maximum variance of the regression predictor, $\mathrm{Var}(\widehat{y})$ with $\widehat{y}$ defined in Eq. (2.12).
- *IMSE-optimality:* integrated mean squared error means minimization of the MSE integrated over the experimental area, with MSE defined in Eq. (2.20); related to the MSE criterion is the *root MSE*, $\mathrm{RMSE} = \sqrt{\mathrm{MSE}}$.

This literature shows that optimal designs do not need to be orthogonal; i.e., these designs may give correlated $\widehat{\beta}_j$.

Note: Actually, there is quite some literature on optimal designs. A classic article is Kiefer and Wolfowitz (1959), and a classic textbook is Fedorov (1972); recent updates are Fedorov and Leonov (2013) and Pronzato and Zhigljavsky (2009). An article on optimal designs specifically for simulation is Bursztyn and Steinberg (2006). Algorithms for the construction of "optimal" designs can be found on the Internet; see

http://optimal-design.biostat.ucla.edu/optimal/home.aspx

and

http://www.itl.nist.gov/div898/handbook/pri/section5/pri521.htm.

Algorithms for the construction of optimal designs assume a given n (total number of combinations) and a specified metamodel; e.g., a first-order polynomial. Other approaches allow for sequential designs (so n is not fixed) and competing metamodels (e.g., first-order and second-order polynomials). Selecting a metamodel among competing models is called *model discrimination.* Tommasi (2009) discusses various optimal-design criteria for model discrimination and parameter estimation.

We shall return to algorithms for the construction of optimal designs in the chapters on Kriging (Chap. 5) and optimization (Chap. 6). We may also use such algorithms to find design types that have the characteristics discussed next.

2.10.2 More Design Types

The DOE literature gives many more design types. For example, R-V designs enable the estimation of *all* $k(k-1)/2$ two-factor interactions, but some designs enable the estimation of *specific* two-factor interactions only—besides the k first-order effects and the intercept.

Note: Ghosh and Tian (2006) assumes that not all two-factor interactions are important; this reference investigates how to discriminate among regression models with different subsets of two-factor interactions. A recent article including references to more publications and software is Grömping (2013).

We may be interested in the identification and estimation of higher-order effects (e.g., third-order effects and thresholds)—in addition to the second-order effects. MacCalman et al. (2013) focuses on sensitivity analysis through metamodels for random simulation models, using a nearly-orthogonal type of Latin hypercube sampling (LHS); we shall further discuss LHS in Sect. 5.5. The resulting LHS designs for $3 \leq k \leq 12$ inputs are catalogued at

`http://harvest.nps.edu`.

This website gives many more design types for simulation studies at the Naval Postgraduate School (NPS) in Monterey, California.

In *mixed-level* designs, some inputs have two levels, some inputs have three levels, etc.; e.g., some inputs are qualitative with more than two levels and some inputs are quantitative with two levels. A textbook that includes mixed-level designs is Wu and Hamada (2009); a recent article is Vieira et al. (2013).

The DOE literature gives many details on *blocked* designs. Such blocking is important in real-life experiments, but not in simulation experiments. Indeed, in real life the environment cannot be controlled, which may lead to effects such as learning effects during experimentation involving humans, and extra wear during experiments with car tires (the right-front tire may wear more than any of the other three tires). In simulation experiments, however, such undesired effects do not occur because everything is completely controlled—except for the PRNs. Antithetic random numbers (ARN) and CRN can be used as a block factor, as originally proposed by Schruben and Margolin (1978) and later on extended in Chih (2013) and Song and Chiu (2007).

In *weighing* designs—also called *mixture* designs—the input values sum-up to 100 %; e.g., chemical experiments may involve inputs that denote the proportion of chemicals used to produce a specific product; see the textbooks Cornell (2011) and Sinha et al. (2014), and the recent article Ceranka and Graczyk (2013).

Usually the experimental area is a k-dimensional *rectangle* or—if the inputs are standardized—a square. Some applications, however, have experimental areas that do not have simple "box" constraints, but more general constraints such that the experimental areas have different shapes. For example, Kleijnen et al. (1979) includes a specific polygon experimental area because the harbor simulation has inputs with values such that the traffic rate remains smaller than 100 %.

2.11 Conclusions

In this chapter we explained linear regression metamodels—especially first-order and second-order polynomials augmented with white noise—and the corresponding statistical designs—namely, designs of resolution III, IV, and

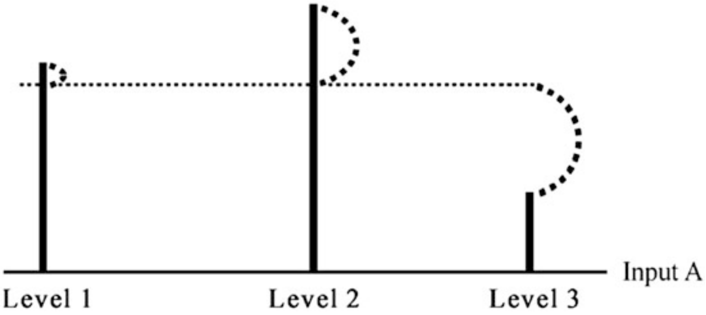


FIGURE 2.7. Input A with three levels

V, and designs called CCDs. We also discussed the lack-of-fit F-test for the validation of the estimated metamodel. In the next chapter, we shall drop the white-noise assumption and discuss the consequences.

Appendix 1: Coding of Nominal Inputs

To illustrate how to represent nominal inputs with two or more levels, Kleijnen (1975, p. 299) discusses an example with two inputs, called A and B; input A has three levels, B has two levels, and there are no replications (so $m_i = 1$). So the matrix of independent variables in the general linear regression model defined in Eq. (2.10) is

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{2.56}$$

where column 1 corresponds with the dummy input, columns 2 through 4 correspond with input A, and columns 5 and 6 with input B. Row 1 means that in input combination 1, A is at its first level and B is also at its first level. Row 2 means that in combination 2, A is still at its first level, but B is at its second level. Row 3 means that in combination 3, A is at its second level, and B is at its first level. And so on, until the last combination (row 6) where A is at its third level, and B is at its second level.

This example implies that the column of regression parameters in Eq. (2.10) becomes $\boldsymbol{\beta} = (\beta_0, \beta_1^A, \beta_2^A, \beta_3^A, \beta_1^B, \beta_2^B)'$. If w denotes the simulation output, then β_0 is the overall or *grand mean*:

$$\beta_0 = \frac{\sum_{i=1}^{3} \sum_{j=1}^{2} E(w_{i;j})}{6}. \tag{2.57}$$

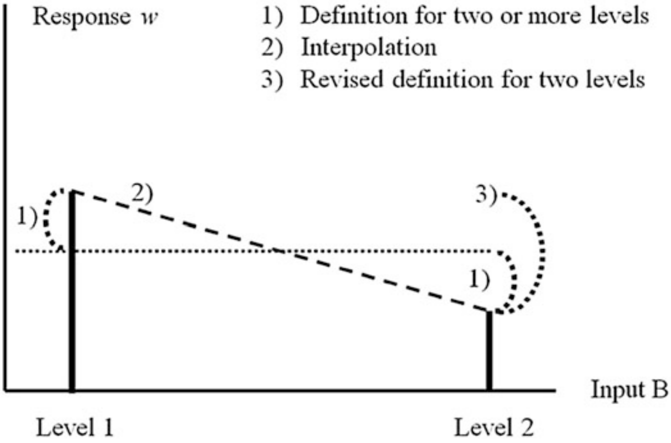

Response w
1) Definition for two or more levels
2) Interpolation
3) Revised definition for two levels
1)
2)
3)
1)
Input B
Level 1
Level 2

FIGURE 2.8. Input B with two levels only

The *main* effect of A at level i is

$$\beta_i^A = \frac{\sum_{j=1}^{2} E(w_{i;j})}{2} - \beta_0 \quad (i = 1, 2, 3) \tag{2.58}$$

—also see Fig. 2.7—and the main effect of B at level j is

$$\beta_j^B = \frac{\sum_{i=1}^{3} E(w_{i;j})}{3} - \beta_0 \quad (j = 1, 2); \tag{2.59}$$

see Fig. 2.8, especially the Legend sub 1. Equations (2.57)–(2.59) give the following two constraints:

$$\beta_1^A + \beta_2^A + \beta_3^A = 0 \tag{2.60}$$

and

$$\beta_1^B + \beta_2^B = 0, \tag{2.61}$$

because the three main effects of A are defined as the deviations from the average response, as is illustrated in Fig. 2.7 where this average is the dotted horizontal line; for B a similar argument applies.

If an input is quantitative, then *interpolation* makes sense; see the dashed line that connects the two responses in Fig. 2.8, especially the legend sub 2. (Input A seems to require a second-order polynomial.) Now we may use the coding that gives -1 and $+1$ discussed in Sect. 2.3.1 (instead of 0 and $+1$, used so far in this appendix). Then β_0 becomes the intercept of the polynomial, β^B becomes the marginal effect $\partial E(w)/\partial B$ (which is an element of the gradient) or the slope of the first-order polynomial, etc. If the inputs have two levels only, then an alternative definition also makes sense; see the legend sub 3 in the plot. This alternative defines "the" effect of an input not as the deviation from the average, but as the difference between the two mean outputs averaged over all levels of the other inputs:

$$\beta^{B} = \frac{\sum_{i=1}^{3} E(w_{i1})}{3} - \frac{\sum_{i=1}^{3} E(w_{i2})}{3}. \tag{2.62}$$

This definition gives values twice as big as the original definition.

The 6×6 matrix $\mathbf{X}$ in Eq. (2.56) does not have full rank; e.g., summing the columns 2 through 4 or the columns 5 and 6 gives column 1. It can be proven that the rank of $\mathbf{X}$ is only four. The normal equations defined in Eq. (2.14) together with the two constraints in Eq. (2.60) and Eq. (2.61) give the unique LS estimate $\widehat{\beta}$; see Bayhan (2004) and its references to classic textbooks. Analysis-of-variance (ANOVA) software uses these computations.

Appendix 2: Manipulating the Generators

Following the DOE literature, we demonstrate some manipulations with design generators. Table 2.1 specified the 2_{III}^{3-1} design with the generator $\mathbf{3} = \mathbf{1.2}$. Remember that $\mathbf{3} = \mathbf{1.2}$ stands for $x_{i3} = x_{i1}x_{i2}$ with $i = 1, \ldots, n$. So postmultiplying both sides of $x_{i3} = x_{i1}x_{i2}$ by x_{i3} gives $(x_{i3})^2 = x_{i1}x_{i2}x_{i3}$. Because x_{i3} is either -1 or $+1$ in a 2^{k-p} design, we may write $(x_{i3})^2 = +1$. Hence, $x_{i1}x_{i2}x_{i3} = +1$. Moreover, the dummy input corresponding with the intercept β_0 implies $x_{i0} = +1$. So, $x_{i1}x_{i2}x_{i3} = x_{i0}$; i.e., the estimates $\widehat{\beta}_0$ and $\widehat{\beta}_{1;2;3}$ are identical. The DOE literature calls $\widehat{\beta}_0$ and $\widehat{\beta}_{1;2;3}$ *confounded* or *aliased*. It is quite easy to prove that $E(\widehat{\beta}_0) = \beta_0 + \beta_{1;2;3}$. So, if $\beta_{1;2;3} = 0$, then $\widehat{\beta}_0$ is unbiased. Actually, in this book we always start our experiments with the (tentative) assumption that high-order interactions are zero; see Algorithm 1.1.

The DOE literature also writes these manipulations in short-hand notation, using the mathematical function mod(2). Let us start again with the generator $\mathbf{3} = \mathbf{1.2}$. Postmultiplying both sides with $\mathbf{3}$ gives $\mathbf{3.3} = \mathbf{1.2.3}$ or $\mathbf{3}^2 = \mathbf{1.2.3}$. Applying mod(2) to the exponent gives $\mathbf{3}^0 = \mathbf{1.2.3}$ where $\mathbf{3}^0 = \mathbf{I}$ with $\mathbf{I}$ denoting a column with n ones; in this appendix, we follow the DOE literature and use the symbol $\mathbf{I}$ instead of $\mathbf{1}_n$ because $\mathbf{1}_n$ or briefly $\mathbf{1}$ may be confused with $\mathbf{1}$, the column for input 1. So $\mathbf{1.2.3} = \mathbf{I}$, which means that $\widehat{\beta}_{1;2;3}$ and $\widehat{\beta}_0$ are confounded. The DOE literature calls $\mathbf{I} = \mathbf{1.2.3}$ the *defining relation*. It can be proven that in a 2^{k-p} design this relation has 2^p members—called *words*.

Similar manipulations can be used to derive that more effects are confounded in this example. Let us start again with the generator $\mathbf{3} = \mathbf{1.2}$ or $\mathbf{I} = \mathbf{1.2.3}$. So $(\mathbf{2.3})\mathbf{I} = (\mathbf{2.3})(\mathbf{1.2.3}) = \mathbf{1.2}^2.\mathbf{3}^2 = \mathbf{1.2}^0.\mathbf{3}^0 = \mathbf{1.I.I} = \mathbf{1}$. So $\mathbf{2.3} = \mathbf{1}$, which implies $E(\widehat{\beta}_1) = \beta_1 + \beta_{2;3}$. However, Table 2.1 is a 2^{3-1}

design with $\mathbf{3} = \mathbf{1.2}$, so we assume that a first-order polynomial (no interactions) is valid so this 2^{3-1} design is a R-III design. Likewise, it is easy to derive that $\mathbf{1.3} = \mathbf{2}$. Summarizing these equations—in the order of the main effects—gives $\mathbf{1} = \mathbf{2.3}$, $\mathbf{2} = \mathbf{1.3}$, and $\mathbf{3} = \mathbf{1.2}$.

Table 2.2 gave the 2^{3-1} design with the generator $\mathbf{3} = -\mathbf{1.2}$. It is easy to derive that this generator implies $\mathbf{1} = -\mathbf{2.3}$, $\mathbf{2} = -\mathbf{1.3}$, and $\mathbf{3} = -\mathbf{1.2}$, so $E(\widehat{\beta}_1) = \beta_1 - \beta_{2;3}$, $E(\widehat{\beta}_2) = \beta_2 - \beta_{1;3}$, and $E(\widehat{\beta}_3) = \beta_3 - \beta_{1;2}$.

Another example is the 2^{7-4} design in Table 2.3. This design has $p = 4$ generators; namely, $\mathbf{4} = \mathbf{1.2}$, $\mathbf{5} = \mathbf{1.3}$, $\mathbf{6} = \mathbf{2.3}$, and $\mathbf{7} = \mathbf{1.2.3}$. Hence $\mathbf{I} = \mathbf{1.2.4} = \mathbf{1.3.5} = \mathbf{2.3.6} = \mathbf{1.2.3.7}$. So $\mathbf{1} = \mathbf{2.4} = \mathbf{3.5} = \mathbf{1.2.3.6} = \mathbf{2.3.7}$. If we assume that high-order interactions are zero, then the latter equations reduce to $\mathbf{1} = \mathbf{2.4} = \mathbf{3.5}$. Analogously, we derive that the other first-order effect estimators are not confounded with any other first-order effect estimators; the first-order effect estimators are confounded with two-factor interaction estimators. So this 2^{7-4} design is a R-III design.

Exercise 2.15 *Derive the expected value of the first-order effect estimator for input 2 in a 2^{7-4} design with the generators $\mathbf{4} = \mathbf{1.2}$, $\mathbf{5} = \mathbf{1.3}$, $\mathbf{6} = \mathbf{2.3}$, and $\mathbf{7} = \mathbf{1.2.3}$, assuming that all high-order interactions are zero.*

A R-IV design for $k = 7$ inputs may be constructed by adding the mirror design of the preceding 2^{7-4}_{III} design. This gives a design with $n = 16$ combinations. Kleijnen (1975, pp. 336–344) shows how to derive the generators of a 2^{k-p}_{IV} design. Furthermore, $n = 16$ combinations give a R-IV design for eight inputs, denoted as a 2^{8-4}_{IV} design; i.e., we may study one extra input when we augment the 2^{7-4}_{III} with its mirror design.

In general, adding the mirror design to a R-III design for k inputs gives a R-IV design for $k+1$ inputs with $n_{IV} = 2n_{III}$ and n_{III} a multiple of four, possibly a power of two. For example, $k = 11$ requires a Plackett-Burman R-III design with $n_{III} = 12$ combinations; see Eq. (2.4). So a R-IV design with $n_{IV} = 24$ combinations enables the estimation of $k = 12$ first-order effects unbiased by two-factor interactions.

A final example is a 2^{8-2} design. Obviously, this design has two generators. A possible generator is $\mathbf{7} = \mathbf{1.2}$, but this generator gives $\mathbf{I} = \mathbf{1.2.7}$ so $\mathbf{1} = \mathbf{2.7}$, $\mathbf{2} = \mathbf{1.7}$, and of course $\mathbf{7} = \mathbf{1.2}$. Another bad generator is $\mathbf{7} = \mathbf{1.2.3}$, because this generator implies $\mathbf{I} = \mathbf{1.2.3.7}$ so $\mathbf{1.2} = \mathbf{3.7},$ etc. In general, a better selection avoids aliasing two-factors interactions, first-order effects, and the intercept. Therefore the generators should multiply more than two inputs; e.g., $\mathbf{7} = \mathbf{1.2.3.4}$ and $\mathbf{8} = \mathbf{1.2.5.6}$, which imply $\mathbf{I} = \mathbf{1.2.3.4.7} = \mathbf{1.2.5.6.8} = \mathbf{3.4.5.6.7.8}$ where the last equality follows from multiplying the first two members of the identity relation. Hence, these two generators confound two-factor interactions with interactions among three or more inputs—the latter (high-order) interactions are assumed to be zero, in this book.

Exercise 2.16 *Prove that if there are $k = 7$ inputs, then* $\mathbf{6} = \mathbf{1.2.3.4.5}$ *and* $\mathbf{7} = \mathbf{1.2.3.4}$ *imply confounding of first-order effects and two-factor interactions; e.g.,* $\mathbf{5} = \mathbf{6.7}$*.*

Solutions of Exercises

Solution 2.1 $\log(y) = \beta_0 + \beta_1 \log \lambda + \ldots$ *so* $y = e^{\beta_0 + \beta_1 \log \lambda + \cdots}$*. Hence*

$$\frac{d}{d\lambda}(e^{\beta_0 + \beta_1 \log \lambda}) = \beta_1 e^{\beta_0} \lambda^{\beta_1 - 1},$$

which upon substitution into the expression for the elasticity coefficient $(dy/d\lambda)(\lambda/y)$ *gives*

$$(\beta_1 e^{\beta_0} \lambda^{\beta_1 - 1})(\lambda / e^{\beta_0 + \beta_1 \log \lambda}) = \lambda \beta_1 \frac{e^{\beta_0}}{e^{\beta_0 + (\ln \lambda)\beta_1}} \lambda^{\beta_1 - 1},$$

which after some manipulation reduces to β_1*.*

Solution 2.2 $E(\hat{\boldsymbol{\beta}}) = \mathbf{L}[E(\mathbf{w})] = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'[\mathbf{X}\boldsymbol{\beta}] = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})\boldsymbol{\beta} = \boldsymbol{\beta}$.

Solution 2.3 $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}} = \mathbf{L}\boldsymbol{\Sigma}_{\mathbf{w}}\mathbf{L}' = [(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'][\sigma_w^2 \mathbf{I}][(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}']'$*. Because* $(\mathbf{X}'\mathbf{X})^{-1}$ *is symmetric, this expression becomes* $[(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'][\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}]\sigma_w^2 = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1})\sigma_w^2 = (\mathbf{X}'\mathbf{X})^{-1})\sigma_w^2$*.*

Solution 2.4 *Equation (2.1) can be written in matrix notation as* $\overline{w} = \mathbf{L}\mathbf{w}$ *with* $\mathbf{L} = (1, \ldots, 1)/c$ *and* $\mathbf{w} = (w_1, \ldots, w_c)'$*. The assumption of waiting times being independent with constant variance* σ^2 *gives* $\boldsymbol{\Sigma}_{\mathbf{w}} = \sigma^2 \mathbf{I}$*. Combining this result with Eq. (2.17) gives* $Var(\overline{w}) = [(1, \ldots, 1)/c][\sigma^2 \mathbf{I}][(1, \ldots, 1)'/c] = \sigma^2[(1, \ldots, 1)/c][(1, \ldots, 1)'/c] = \sigma^2[c/(c^2)] = \sigma^2/c$*.*

Solution 2.5 *Program this Monte Carlo model, and experiment with this model and variations on this model. These are the results that we found:*

(a) $\widehat{\gamma}_0 = 80.84$ *and* $\widehat{\gamma}_1 = 16.00$

(b) $\widehat{\beta}_0 = 168.84$ *and* $\widehat{\beta}_1 = 56.92$

(c) $F_{5-2;5\times(4-1)} = 9.68 > F_{3;15;0.90} = 2.48$ *so lack-of-fit is significant*

(d) $\widehat{\gamma}_0 = 100.97$, $\widehat{\gamma}_1 = 5.00$ *and* $\widehat{\gamma}_2 = 1.00$ *(extremely close to the true values* $\gamma_0 = 100.00$, $\gamma_1 = 5.00$ *and* $\gamma_2 = 1.00$ *);* $F_{5-3;5\times(4-1)} = 2.0455E{-}28 < F_{2;15;0.90} = 2.69$ *(Note:* $F_{2;15;0.90} = 2.69 > F_{3;15;0.90} = 2.48$*; see (c))*

(e) $\widehat{\gamma}_0 = 70.89$ *and* $\widehat{\gamma}_1 = 16.00$*;* $F_{3-2;3\times(4-1)} = 0.0034 < F_{1;9;0.90} = 3.36$

(f) $\widehat{\gamma}_0 = 74.09$ *and* $\widehat{\gamma}_1 = 16.00$*;* $F_{3-2;3\times(4-1)} = 1.38 < F_{1;9;0.90} = 3.36$

(g) $\widehat{\gamma}_0 = 67.97$ *and* $\widehat{\gamma}_1 = 16.50$*; the numerator of* $F_{2-2;3\times(4-1)}$ *has zero degrees of freedom so no lack-of-fit test is possible.*

Solution 2.6 *Equation (2.18) implies* $\mathbf{\Sigma}_{\hat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\sigma_w^2$*. Suppose* $\sigma_w^2 = 1$ *and*

$$\mathbf{X} = \begin{bmatrix} 1 & l \\ 1 & u \end{bmatrix}.$$

Then

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 1 & 1 \\ l & u \end{bmatrix} \begin{bmatrix} 1 & l \\ 1 & u \end{bmatrix} = \begin{bmatrix} 2 & l+u \\ l+u & l^2+u^2 \end{bmatrix}$$

so

$$(\mathbf{X}'\mathbf{X})^{-\mathbf{1}} = \begin{bmatrix} \frac{l^2+u^2}{-2lu+l^2+u^2} & \frac{-l-u}{-2lu+l^2+u^2} \\ \frac{-l-u}{-2lu+l^2+u^2} & \frac{2}{-2lu+l^2+u^2} \end{bmatrix}$$

so

$$Var(\hat{\beta}_1) = \frac{2}{-2lu+l^2+u^2} = \frac{2}{(u-l)^2}.$$

This variance is minimal if the denominator $(u-l)^2$ *is maximal, which occurs if* l *and* u *are as far apart as possible.*

Solution 2.7 *The experimental area* $0.2 \leq z \leq 0.5$ *implies* $a = (0.2+0.5)/(0.2-0.5) = -2.\,333$ *and* $b = 2/(0.5-0.2) = 6.667$*. Hence* $x = -2.333 + 6.667z$ *so* $x_{min} = -2.333 + (6.667)(0.2) = -1$ *and* $x_{max} = -2.333 + (6.667)(0.5) = 1$*. Further,* $z = 0.3$ *implies* $x = -2.333 + (6.667)(0.3) = -0.333$*. Likewise* $z = 0.4$ *implies* $x = -2.333 + (6.667)(0.4) = 0.333$*.*

Solution 2.8 *The average* $\overline{z}_j$ *is a constant determined before the experiment is carried out; if the design is balanced, then* $\overline{z}_j = (l_j + u_j)/2$*. Hence, the marginal effect of* z_j *is* δ_j*. The total effect over the range of* z_j *is* $\delta_j(u_j - l_j) = 2\beta_j$*.*

Solution 2.9 *The design matrix is a* 16×15 *matrix with all elements either* $-1 or$ $+1$*; to verify that you correctly applied the algorithm, you can use a computer to check that each column has exactly 8 pluses (balanced design), and that all* $15 \times 14/2 = 105$ *columns are orthogonal.*

Solution 2.10 *The solution depends on the simulation model that you selected.*

Solution 2.11 *Table 2.3 gives the following table:*

Combination	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	−	−	−	+	+	+	−
2	+	−	−	−	−	+	+
3	−	+	−	−	+	−	+
4	+	+	−	+	−	−	−
5	−	−	+	+	−	−	+
6	+	−	+	−	+	−	−
7	−	+	+	−	−	+	−
8	+	+	+	+	+	+	+

so adding its mirror design and adding the column **6.7** *for the interaction* $\beta_{6;7}$ *gives the augmented design*

Combination	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**6.7**
1	−	−	−	+	+	+	−	−
2	+	−	−	−	−	+	+	+
3	−	+	−	−	+	−	+	−
4	+	+	−	+	−	−	−	+
5	−	−	+	+	−	−	+	−
6	+	−	+	−	+	−	−	+
7	−	+	+	−	−	+	−	−
8	+	−	−	+	+	+	+	+
9	+	+	+	−	−	−	+	−
10	−	+	+	+	+	−	−	+
11	+	−	+	+	−	+	−	−
12	−	−	+	−	+	+	+	+
13	+	+	−	−	+	+	−	−
14	−	+	−	+	−	+	+	+
15	+	−	−	+	+	−	+	−
16	−	−	−	−	−	−	−	+

so it is easy to check that the column **6.7** *is balanced and orthogonal to the columns* **6** *and* **7**

Solution 2.12 *Adding the dummy column for the intercept to Rechtschaffner's design gives*

$$\mathbf{X} = \begin{bmatrix} 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 \end{bmatrix}$$

so

$$\mathbf{X'X} = \begin{bmatrix} 11 & 1 & 1 & 1 & 1 \\ 1 & 11 & -1 & -1 & -1 \\ 1 & -1 & 11 & -1 & -1 \\ 1 & -1 & -1 & 11 & -1 \\ 1 & -1 & -1 & -1 & 11 \end{bmatrix}$$

so

$$(\mathbf{X'X})^{-1} = \begin{bmatrix} \frac{2}{21} & -\frac{1}{84} & -\frac{1}{84} & -\frac{1}{84} & -\frac{1}{84} \\ -\frac{1}{84} & \frac{2}{21} & \frac{1}{84} & \frac{1}{84} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{2}{21} & \frac{1}{84} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{2}{21} & \frac{1}{84} \\ -\frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{1}{84} & \frac{2}{21} \end{bmatrix},$$

whereas an orthogonal design matrix and $\sigma_w^2 = 1$ *would imply* $Var(\widehat{\beta}_j) = 1/n = 1/11 = 0.09 < 2/21 = 0.95$.

Solution 2.13 *The variance of the predicted output increases as the input combination moves away from the center of the experimental area. (Also see the discussion on steepest ascent in Sect. 6.2.3.)*

Solution 2.14 *The answer depends on the simulation model that you select. Because you select a model with a known optimal solution, you can easily examine the performance of the CCD.*

Solution 2.15 $\mathbf{I} = \mathbf{1.2.4} = \mathbf{1.3.5} = \mathbf{2.3.6} = \mathbf{1.2.3.7}$ *implies* $\mathbf{2} = \mathbf{1.4} = \mathbf{1.2.3.5} = \mathbf{3.6} = \mathbf{1.3.7}$. *Assuming zero high-order effects, we obtain* $\mathbf{2} = \mathbf{1.4} = \mathbf{3.6}$ *so* $E(\widehat{\beta}_2) = \beta_2 + \beta_{1;4} + \beta_{3;6}$.

Solution 2.16 $\mathbf{6} = \mathbf{1.2.3.4.5}$ *implies* $\mathbf{I} = \mathbf{1.2.3.4.5.6}$, *and* $\mathbf{7} = \mathbf{1.2.3.4}$ *implies* $\mathbf{I} = \mathbf{1.2.3.4.7}$. *So* $\mathbf{6.7} = (\mathbf{1.2.3.4.5})(\mathbf{1.2.3.4}) = \mathbf{5}$.

References

Alaeddini A, Yang K, Mao H, Murat A, Ankenman B (2013) An adaptive sequential experimentation methodology for expensive response surface optimization—case study in traumatic brain injury modeling. Qual Reliab Eng Int. doi:10.1002/qre.1523

Barton RR, Meckesheimer M (2006) Chapter 18: Metamodel-based simulation optimization. In: Henderson SG, Nelson BL (eds) Handbooks in Operations Research and Management Science, vol 13. North-Holland, Amsterdam, pp 535–574

Bayhan GM (2004) An alternative procedure for the estimation problem in 2^n factorial experimental models. Comput Indust Eng 47:1–15

Bettonvil B, Kleijnen JPC (1990) Measurement scales and resolution IV designs. Am J Math Manag Sci 10(3 & 4):309–322

Box GEP (1952) Multi-factor designs of first order. Biometrika 39(1):49–57

Box GEP, Draper NR (1959) A basis for the selection of a response surface design. J Am Stat Assoc 54:622–654

Box GEP, Hunter JS (1961a) The 2^{k-p} fractional factorial designs, part I. Technometrics 3:311–351

Box GEP, Hunter JS (1961b) The 2^{k-p} fractional factorial designs, Part II. Technometrics 3:449–458

Box GEP, Wilson KB (1951) On the experimental attainment of optimum conditions. J R Stat Soc Ser B 13(1):1–38

Breukers A (2006) Bio-economic modelling of brown rot in the Dutch potato production chain. Doctoral dissertation, Wageningen University, Wageningen

Bursztyn D, Steinberg DM (2006) Comparison of designs for computer experiments. J Stat Plan Inference 136:1103–1119

Ceranka B, Graczyk M (2013) On the construction of regular A-optimal spring balance weighing designs. Colloq Biom 43:13–19

Chih M (2013) A more accurate second-order polynomial metamodel using a pseudo-random number assignment strategy. J Oper Res Soc 64:198–207

Cooper WW (2009) Origins and uses of linear programming methods for treating and regressions: corrections and comments on Castillo et al. (2008). Eur J Oper Res 198(1):361–362

Cornell JA (2011) A primer on experiments with mixtures. Wiley, Hoboken

Craigen R (1996) Hadamard matrices and designs. Colbourn CJ, Dinitz JH (eds) Handbook of Combinatorial Designs. CRC, Boca Raton, pp 370–377

Dengiz B, Bektas T, Ultanir AE (2006) Simulation optimization based DSS application: a diamond tool production line in industry. Simul Model Pract Theory 14(3):296–312

Draper NR, Lin DKJ (1990) Small response-surface designs. Technometrics 32(2):187–194

Ekren BY, Ornek AM (2008) A simulation based experimental design to analyze factors affecting production flow time. Simul Model Pract Theory 16:278–29

Elsawah AM, Qin H (2015) A new strategy for optimal foldover two-level designs. Statistics & Probability Letters, (in press)

Fang K-T, Li R, Sudjianto A (2006) Design and modeling for computer experiments. Chapman & Hall/CRC, London

Fedorov VV (1972) Theory of optimal experiments. Academic, New York

Fedorov VV, Leonov SL (2013) Optimal design for nonlinear response models. Chapman & Hall/CRC Biostatistics Series. Taylor & Francis, Boca Raton

Fisher RA (1935) The design of experiments. Oliver & Boyd, Oxford

Frey DD, Wang H (2006) Adaptive one-factor-at-a-time experimentation and expected value of improvement. Technometrics 48(3):418–431

Ghosh S, Tian Y (2006) Optimum two level fractional factorial plans for model identification and discrimination. J Multivar Anal 97(6):1437–1450

Grömping U (2013) A note on dominating fractional factorial two-level designs with clear two-factor interactions. Technometrics. doi:10.1080/00401706.2013.822425

Ho Y, Cao X (1991) Perturbation analysis of discrete event dynamic systems. Kluwer, Dordrecht

Kelton WD, Sadowski RP, Sturrock DT (2007) Simulation with Arena; fourth edition. McGraw-Hill, Boston

Khuri AI, Mukhopadhyay S (2010) Response surface methodology. Wiley 1617 Interdiscip Rev Comput Stat 2:128–49

Kiefer J, Wolfowitz J (1959) Optimum designs in regression problems. Ann Math Stat 30:271–294

Kleijnen JPC (1975) Statistical techniques in simulation; part II. Marcel Dekker, New York (Russian translation, Publishing House 'Statistika', Moscow, 1978)

Kleijnen JPC (1987) Statistical tools for simulation practitioners. Marcel Dekker, New York

Kleijnen JPC (1995) Case study: statistical validation of simulation models. Eur J Oper Res 87(1):21–34

Kleijnen J, Mehdad E (2014) Multivariate versus univariate Kriging metamodels for multi-response simulation models. Eur J Oper Res 236(2):573–582

Kleijnen JPC, Pala O (1999) Maximizing the simulation output: a competition. Simulation 73(3):168–173

Kleijnen JPC, Sargent RG (2000) A methodology for the fitting and validation of metamodels in simulation. Eur J Oper Res 120(1):14–29

Kleijnen JPC, Standridge C (1988) Experimental design and regression analysis: an FMS case study. Eur J Oper Res 33(3):257–261

Kleijnen JPC, van den Burg AJ, van der Ham R Th (1979) Generalization of simulation results: practicality of statistical methods. Eur J Oper Res 3:50–64

Kleijnen JPC, van Schaik FDJ (2011) Sealed-bid auction of Netherlands mussels: statistical analysis. Int J Prod Econ 132(1):154–161

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Li W, Lin DKJ (2015) A note on foldover of 2^{k-p} designs with column permutations. Technometrics (in press)

Lin M, Lucas HC, Shmueli G (2013) Research commentary—too big to fail: large samples and the p-value problem. Inf Syst Res 24(4):906–917

MacCalman AD, Vieira H, Lucas T (2013) Nearly orthogonal Latin hypercubes for exploring stochastic simulations. Naval Postgraduate School, Monterey

Mee RW (2004) Efficient two-level designs for estimating all main effects and two-factor interactions. J Qual Technol 36:400–412

Mee R (2009) A comprehensive guide to factorial two-level experimentation. Springer, New York

Miller GA (1956) The magical number seven plus or minus two: some limits on our capacity for processing information. Psychol Rev 63:81–97

Montgomery DC (2009) Design and analysis of experiments, 7th edn. Wiley, Hoboken

Myers RH, Montgomery DC, Anderson-Cook CM (2009) Response surface methodology: process and product optimization using designed experiments, 3rd edn. Wiley, New York

Narula SC, Wellington JF (2007) Multiple criteria linear regression. Eur J Oper Res 181(2):767–772

Oh RPT, Sanchez SM, Lucas TW, Wan H, Nissen ME (2009) Efficient experimental design tools for exploring large simulation models. Comput Math Organ Theory 15(3):237–257

Plackett RL, Burman JP (1946) The design of optimum multifactorial experiments. Biometrika 33:305–325

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007) Numerical recipes: the art of scientific computing, third edition. Cambridge University Press

Pronzato L, Zhigljavsky A (eds) (2009) Optimal design and related areas in optimization and statistics. Springer, New York

Qu X (2007) Statistical properties of Rechtschaffner designs. J Stat Plan Inference 137:2156–2164

Rechtschaffner RL (1967) Saturated fractions of 2^n and 3^n factorial designs. Technometrics 9:569–575

Rubinstein RY, Shapiro A (1993) Discrete-event systems: sensitivity analysis and stochastic optimization via the score function method. Wiley, New York

Ryan KJ, Bulutoglu DA (2010) Minimum aberration fractional factorial designs with large N. Technometrics 52(2):250–255

Sanchez SM, Sanchez PJ (2005) Very large fractional factorial and central composite designs. ACM Trans Model Comput Simul 15(4):362–377

Santner TJ, Williams BJ, Notz WI (2003) The design and analysis of computer experiments. Springer, New York

Schruben LW, Margolin BH (1978) Pseudorandom number assignment in statistically designed simulation and distribution sampling experiments. J Am Stat Assoc 73(363):504–525

Searle SR (1971) Linear models. Wiley, New York

Shi W, Kleijnen JPC, Liu Z (2014) Factor screening for simulation with multiple responses: sequential bifurcation. Eur J Oper Res 237(1):136–147

Shrivastava AK, Ding Y (2010) Graph based isomorph-free generation of two-level regular fractional factorial designs. J Stat Plan Inference 140:169–179

Sinha BK, Mandal NK, Pal M, Das P (2014) Optimal mixture experiments. Springer, New Delhi

Song WT, Chiu W (2007) A five-class variance swapping rule for simulation experiments: a correlated-blocks design. IIE Trans 39:713–722

Spall JC (2003) Introduction to stochastic search and optimization; estimation, simulation, and control. Wiley, New York

Spall JC (2010) Factorial design for efficient experimentation: generating informative data for system identification. IEEE Control Syst Mag 30(5):38–53

Tian Y, Wiens DP (2006) On equality and proportionality of ordinary least squares, weighted least squares and best linear unbiased estimators in the general linear model. Stat Probab Lett 76(12):1265–1272

Tommasi C (2009) Optimal designs for both model discrimination and parameter estimation. J Stat Plan Inference 139(12):4123–4132

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling 719 in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690

Vieira H, Sanchez SM, Kienitz KH, Belderrain MCN (2013) Efficient, nearly orthogonal-and-balanced, mixed designs: an effective way to conduct trade-off analyses via simulation. J Simul 7:264–275

Voelkel JG (2005) The efficiencies of fractional factorial designs. Technometrics 47(4):488–494

Webb S (1968) Non-orthogonal designs of even resolution. Technometrics 10:291–299

Wu CFJ, Hamada M (2009) Experiments; planning, analysis, and parameter design optimization, 2nd edn. Wiley, New York

3
Classic Assumptions Versus Simulation Practice

This chapter is organized as follows. Section 3.1 summarizes the classic assumptions of regression analysis, which were given in the preceding chapter. Section 3.2 discusses multiple simulation outputs (responses, performance measures), which are usual in simulation practice. Section 3.3 addresses possible nonnormality of either the simulation output itself or the regression residuals (fitting errors), including tests of normality, normalizing transformations of the simulation output, and jackknifing and bootstrapping of nonnormal output. Section 3.4 covers variance heterogeneity of the simulation output, which is usual in random simulation. Section 3.5 discusses cross-correlated simulation outputs created through common random numbers (CRN); the use of CRN is popular in random simulation. Section 3.6 discusses the validation of estimated regression models, including the coefficient of determination R^2 and the adjusted coefficient R^2_{adj}, and cross-validation; this section also discusses how classic low-order polynomial metamodels (detailed in the preceding chapter) may be improved in practice. Section 3.7 summarizes the major conclusions of this chapter. The chapter ends with solutions for the exercises, and a long list with references for further study.

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_3

3.1 Introduction

In this chapter, we examine the assumptions of classic linear regression analysis and its concomitant designs that we discussed in the preceding chapter. These assumptions stipulate a single type of output (univariate output) and white noise. In practice, however, these assumptions usually do not hold. Indeed, many simulation models give multiple responses or—in statistical jargon—a *multivariate* random variable. One example is the simple M/M/1 simulation model (see Definition 1.4), which may have as outputs both the mean waiting time and the (say) 90 % quantile of the waiting time distribution (in practice, such a quantile may be more important than the mean). A second related example is the M/M/1 simulation model that has as outputs the mean waiting time and the mean queue length. More examples will follow in Sect. 3.2. *White noise* was defined in Definition 2.3 and was used many times in the preceding chapter; for the reader's convenience we repeat this definition.

Definition 3.1 *White noise (say) u is normally, independently, and identically distributed (NIID) with zero mean: $u \sim$NIID$(0, \sigma_u^2)$.*

This definition implies the following assumptions.

- Normally distributed simulation responses
- No use of CRN across the (say) n combinations of inputs (factors)
- Homogeneous variances; i.e., the variances of the simulation outputs remain constant across the n input combinations
- Valid metamodel so the estimated metamodel has zero mean residuals.

In this chapter, we shall try to answer the following questions:

1. How realistic are the classic assumptions in either deterministic or random simulation?
2. How can we test these assumptions if it is not obvious that these assumptions are violated (e.g., the use of CRN obviously violates the independence assumption)?
3. If an assumption is violated, can we then transform the simulation's input/output (I/O) data so that the assumption holds for the transformed data?
4. If we cannot find such a transformation, which statistical methods can we then apply?

The answers to these questions are scattered throughout the literature on statistics and simulation; in this chapter, we try to answer these questions in a coherent way. We focus on random simulation, but we also briefly discuss deterministic simulation.

3.2 Multivariate Output

In practice, simulation models often give multiple outputs. We shall discuss the consequences for deterministic simulation, in the last paragraph of Sect. 3.2.1. In the rest of this section we focus on random simulation with multivariate output.

Examples are *inventory simulations*; a classic inventory model is the (s, S) model defined in Definition 1.5. Practical inventory simulations often have the following two outputs:

1. the sum of the holding costs and the ordering costs, averaged over the simulated periods;

2. the service (or fill) rate, averaged over the same simulation periods.

The precise definitions of these two outputs vary in practice. For example, the holding costs may have fixed and variable components; the service rate may be the fraction of total demand per year that is delivered from the stock at hand. Academic inventory simulations often have a single output; namely, the total costs including out-of-stock costs. In practice, however, these out-of-stock costs are hard to quantify; e.g., what are the costs of loss-of-goodwill? Therefore, inventory simulations often have the two outputs listed above. Moreover, in practice the inventory managers may have to control hundreds or thousands of inventory items or "stock keeping units (SKUs)". Further discussion can be found in simulation textbooks such as Law (2015) and in many Management Science/Operations Research (MS/OR) textbooks.

A *case study* with multiple simulation responses is the decision support system (DSS) for the production planning of steel tubes based on a simulation model, presented in Kleijnen (1993). In the beginning of that study the simulation had a multitude of outputs; however, to support decision making it turned out that it suffices to consider only the following two (random) outputs:

1. the total production of steel tubes manufactured, which is of major interest to the production manager;

2. the 90 % quantile of delivery times, which is the sales manager's major concern.

In Eq. (2.6) we have already formulated the *general* black-box equation, which we repeat now:

$$\mathbf{w} = f_{\text{sim}}(d_1, \ldots, d_k, \mathbf{r}_0) = f_{\text{sim}}(\mathbf{d}, \mathbf{r}_0) \tag{3.1}$$

where $\mathbf{w} = (w_1, \ldots, w_r)'$ denotes the vector with the (say) r types of simulation outputs; $f_{\text{sim}}(.)$ denotes the mathematical function defined by the simulation computer code implementing the given simulation model; $d_j (j = 1, \ldots, k)$ is input j of the computer code; in deterministic simulation, $\mathbf{r}_0$ denoting the vector with the seeds of the pseudorandom number (PRN) streams vanishes; the k inputs are collected in the vector $\mathbf{d} = (d_1, \ldots, d_k)'$. We let $\mathbf{D} = (d_{i;j})$ denote the design matrix for the simulation experiment, with $i = 1, \ldots, n$ and n the number of input combinations in that experiment. For simplicity of notation and explanation we assume in this section that the number of replications is $m_i = 1$.

3.2.1 Linear Regression Metamodels

Analogous to Eq. (2.10) in the preceding chapter (with a single type of simulation output so $r = 1$), we now assume that the multivariate I/O function $f_{\text{sim}}(.)$ in Eq. (3.1) is approximated by r univariate linear regression metamodels (e.g., low-order polynomials):

$$\mathbf{y}_h = \mathbf{X}_h \boldsymbol{\beta}_h + \mathbf{e}_h \quad \text{with } h = 1, \ldots r \tag{3.2}$$

where $\mathbf{y}_h = (y_{1;h}, \ldots, y_{n;h})'$ denotes the n-dimensional vector with the dependent variable y_h corresponding with simulation output type h; n denotes the number of simulated input combinations; $\mathbf{X}_h = (\mathbf{x}_{i;j;h})$ denotes the $n \times q_h$ matrix of independent regression variables with $\mathbf{x}_{i;j;h}$ denoting the value of independent variable j in combination i for metamodel h $(i = 1, \ldots, n;\ j = 1, \ldots, q_h)$; $\boldsymbol{\beta}_h = (\beta_{1;h}, \ldots, \beta_{q_h;h})'$ denotes the vector with the q_h regression parameters for metamodel h; $\mathbf{e}_h = (e_{1;h}, \ldots, e_{n;h})'$ denotes the n-dimensional vector with the residuals of metamodel h, in the n combinations. Usually column 1 of $\mathbf{X}_h$ equals $\mathbf{1}_n = (1, \ldots, 1)'$, which denotes a vector with n ones corresponding with the intercept $\beta_{1;h}$ (often denoted as $\beta_{0;h}$) and columns 2 through $1 + k$ equal $d_{i;1}$ through $d_{i;k}$. For simplicity we might assume that all the r fitted regression metamodels are polynomials of the same order (e.g., second-order), so $\mathbf{X}_h = \mathbf{X}$ and $q_h = q$. Altogether, multivariate regression analysis requires a rather complicated notation.

The literature uses the following terminology: if $q > 1$ and the metamodel has an intercept, then the metamodel is called a *multiple* regression model; if $r > 1$, then the metamodel is called a *multivariate* regression model. A multivariate linear regression model violates the classic assumptions, as the following simplistic example illustrates.

Example 3.1 *Consider only two input combinations, so $n = 2$. Suppose further that each combination gives three outputs, so $r = 3$. Furthermore, suppose that the simulation is random and does not use CRN. Finally, suppose that the variances and covariances do not vary over the n combinations. These assumptions give the following covariance matrix, where we display only the elements on and above the main diagonal because covariance matrixes are symmetric:*

$$\mathbf{\Sigma_e} = \begin{bmatrix} \sigma_1^2 & \sigma_{1;2} & \sigma_{1;3} & 0 & 0 & 0 \\ & \sigma_2^2 & \sigma_{2;3} & 0 & 0 & 0 \\ & & \sigma_3^2 & 0 & 0 & 0 \\ & & & \sigma_1^2 & \sigma_{1;2} & \sigma_{1;3} \\ & & & & \sigma_2^2 & \sigma_{2;3} \\ & & & & & \sigma_3^2 \end{bmatrix}.$$

This example illustrates that multivariate residuals **e** have the following two *properties* in random simulation (we shall discuss deterministic simulation at the end of this subsection).

1. The residuals e_h have variances that may vary with the type of simulation output h so $\sigma_h^2 \neq \sigma^2$. Practical examples are simulation models that estimate inventory costs and service percentages; obviously, these two output types have different variances.
2. The residuals e_h and $e_{h'}$ are not independent for a given input combination i, because they are (different) transformations of the same PRN stream; so if $h \neq h'$, then $\sigma_{h;h';i} \neq 0$. Obviously, if these covariances (like the variances) do not vary with combination i, then this property may be written as $\sigma_{h;h';i} = \sigma_{h;h'} \neq 0$ for $h \neq h'$. For example, the seed vector $\mathbf{r}_0$ may give "unusual" PRN streams in a given combination i so the inventory costs are "relatively high"—that is, higher than expected—and the service percentage is also relatively high; obviously, in this example the two outputs are positively correlated so $\sigma_{h;h'} > 0$.

Because of these two properties ($\sigma_h^2 \neq \sigma^2$ and $\sigma_{h;h'} \neq 0$ for $h \neq h'$), the classic assumptions do not hold. Consequently, it might seem that in *multivariate regression* we need to replace classic ordinary least squares (OLS) by *generalized least squares* (GLS); see Khuri and Mukhopadhyay (2010). Such an approach tends to be rather complicated, because GLS involves the covariance matrix $\mathbf{\Sigma_e}$ so simulation analysts may be daunted; also see Gilbert and Zemčík (2006). Fortunately, Rao (1967) proves that if $\mathbf{X}_h = \mathbf{X}$, then GLS reduces to OLS computed per type of output variable; In this section we assume that $\mathbf{X}_h = \mathbf{X}$; e.g., we fit a (different) first-order polynomial per type of simulation output h. Consequently, the *best linear unbiased estimator* (BLUE) of $\boldsymbol{\beta}_h$ in Eq. (3.2) is

$$\widehat{\boldsymbol{\beta}}_h = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}_h \; (h = 1, \ldots, r). \tag{3.3}$$

References more recent than Rao (1967) are Markiewicz and Szczepańska (2007) and Ruud (2000, p. 703).

Given Eq. (3.3), we can easily obtain confidence intervals (CIs) and statistical tests for the regression parameters per output type; i.e., we may use the classic formulas presented in the preceding chapter.

Deterministic simulation gives residuals $\mathbf{e}_h$ in Eq. (3.2) that certainly violate the classic assumptions. Nevertheless, we may still fit the linear metamodel defined in Eq. (3.2). To fit this model, we may still apply OLS, using the mathematical distance measurement known as the L_2 norm. This norm is much more popular than the L_1 and L_∞ norms, because the L_2 norm results in a linear estimator $\widehat{\boldsymbol{\beta}}$ that is easy to compute and analyze (also see the discussion below Eq. (2.14)). Obviously, these mathematical norms differ from statistical norms such as the BLUE and the MLE (maximum likelihood estimator) criteria. These mathematical norms do not give CIs and statistical tests; nevertheless, we may evaluate the quality of the $\widehat{\boldsymbol{\beta}}_h$ estimates defined in Eq. (3.3) through cross-validation, as we shall see in Sect. 3.6.2.

3.2.2 Designs for Multivariate Simulation Output

To the best of our knowledge, there are no *general one-shot* (nonsequential) designs for multivariate output; see Khuri and Mukhopadhyay (2010). Let us consider a simple, artificial example that is inspired by Breukers (2006).

Example 3.2 *Suppose we are interested in two types of simulation output so* $r = 2$ *in Eq. (3.1), and the number of simulation inputs is 15 so* $k = 15$ *in Eq. (3.1). First, we try to estimate the first-order effects, so we use a resolution-III (R-III) design; namely, a* 2^{15-11} *design defined in Sect. 2.4. Suppose that after running this design, we find that the inputs labeled 1 through 7 have important first-order effects for response type 1, while the inputs labeled 6 through 15 have important first-order effects for response type 2. In the next stage of our investigation, we want to estimate the two-factor interactions between those inputs that have important first-order effects in the first stage; i.e., we use the "strong heredity" assumption, which states that if an input has no important first-order effect, then this input does not interact with any other input; see Wu and Hamada (2009). Because the number of possible two-factor interactions is* $k(k-1)/2$*, this number sharply increases with k. In this example it is therefore efficient to estimate the interactions in two separate experiments; namely, one experiment per type of simulation output. So we split the original group of* $k = 15$ *inputs into two subgroups; namely, one subgroup with* $k_0 = 7$ *inputs for the simulation response labeled 1 and* $k_1 = 10$ *inputs for the simulation response labeled 2 where the inputs labeled 6 and 7 are members of both subgroups. The original group with* 15 *inputs would require* $1 + 15 + 15 \times (15 - 1)/2 = 121$ *combinations at least (121 is a high number of combinations; moreover, classic resolution-V designs are often not saturated at all, so these designs require even more than 121 combinations; see the detailed discussion in Sect. 2.7). Now the first subgroup requires at least* $1 + 7 + 7 \times (7 - 1)/2 = 29$ *combinations, and the second subgroup requires at least* $1 + 10 + 10 \times (10 - 1)/2 = 56$ *combinations. So, together the two subgroups require at least* $29 + 56 = 85$ *instead of* 121 *combinations; i.e., a "divide and conquer" strategy turns out to pay off.*

3.3 Nonnormal Output

We repeat the comment in Sect. 3.2.1 (last paragraph); namely, OLS is a mathematical criterion so OLS does not assume a normal (Gaussian) distribution. Only if we require statistical properties—such as BLUEs, CIs, and tests—then we usually assume a normal distribution (alternative distributions corresponding to alternative criteria such as the L_1 and the L_∞ norms are discussed in Narula and Wellington, 2007). In this section we try to answer the following questions (already formulated more generally in Sect. 3.1):

1. How realistic is the normality assumption?

2. How can this assumption be tested?

3. How can the simulation output be transformed so that the normality assumption holds?

4. Which statistical methods that do not assume normality, can be applied?

3.3.1 Realistic Normality Assumption?

By definition, *deterministic* simulation models do not have a normally distributed output for a given input combination; actually, this output is a single fixed value. Nevertheless, we often assume a normal distribution for the *residuals* of the fitted metamodel. An example is the case study on coal mining, using deterministic "system dynamics" simulation, in Kleijnen (1995). Another case study examines global heating caused by the CO_2 greenhouse effect, using deterministic simulation, in Kleijnen et al. (1992). We also refer to our discussion of deterministic simulation in the chapter on Kriging (Chap. 5). Indeed, we might argue that so many things affect the residuals that the classic *central limit theorem* (CLT) applies, so a normal distribution is a good assumption for the residuals; we shall return to the CLT below (immediately after Definition 3.2).

In this subsection we again focus on *random* simulation models. We need the following definition, which uses notation $\sigma_{|t-t'|}$ such that $\sigma_{|0|} = \sigma^2$.

Definition 3.2 *The time series (say) w_t is a stationary process if it has a constant mean $E(w_t) = \mu$, a constant variance $Var(w_t) = \sigma^2$, and covariances that depend only on the so-called lag $|t - t'|$ so $cov(w_t, w_{t'}) = \sigma_{|t-t'|}$.*

In practical and academic simulation models, the normality assumption often holds *asymptotically*; i.e., if the "sample" size is large, then functions of the random simulation data—in particular the sample average of those data—give nearly normal output. Basic statistics books mention that the CLT explains why an average is often normally distributed. The

CLT assumes that this average has independent components. In simulation, however, the output of a simulation run is often an average computed over that run so the components are *autocorrelated* (serially correlated). Fortunately, there are (sophisticated) variations of the CLT that explain why and when this correlation does not destroy the normality of the average in many simulations. For example, Kim et al. (2007) discusses the *functional central limit theorem* (FCLT) and gives references including the classic textbook Billingsley (1968). Furthermore, the textbook Lehmann (1999, Chap. 2.8) implies that the average of a stationary process remains asymptotically normally distributed if the covariances tend to zero sufficiently fast for large lags.

We add that in inventory simulations the output is often the costs averaged over the simulated periods; this average is probably normally distributed. Another output of an inventory simulation may be the service percentage calculated as the fraction of demand delivered from on-hand stock per (say) week, so "the" output is the average per year computed from these 52 weekly averages. This yearly average may be normally distributed—unless the service goal is "close" to 100 %, so the average service rate is cut off at this threshold and the normal distribution is a bad approximation. *Quantiles* of correlated or uncorrelated observations may be very nonnormal, especially if they are rather extreme (such as the 99 % quantile). We point out that the t-statistic is quite insensitive to nonnormality, whereas the F-statistic is more sensitive to nonnormality; see the many references in the textbook Kleijnen (1987).

Example 3.3 *Kleijnen and Van Beers (2013) investigates an M/M/1 simulation model; such a model has already been discussed in Example 1.2. That investigation considers two outputs; namely, (i) the average, and (ii) the 90 % quantile of the waiting time, after simulating (say) T customers. So the model generates a time series of length T or a vector output $\mathbf{w} = (w_1, \ldots, w_T)'$ with positively correlated scalar components w_t. These components have variances $Var(w_t)$ that increase as t increases, until the steady state is reached (so $Var(w_t)$ becomes a constant that is independent of t). The two performance measures are nonlinear functions of the traffic rate $x = \lambda/\mu$ where λ and μ denote the arrival and the service rate, respectively. The simulation does not start with $w_1 = 0$ (the usual initialization) but with w_1 equal to the mean steady-state waiting time; i.e., the experiment "cheats" and uses the analytical solution for the mean steady-state waiting time. The reason for this initialization is that it accelerates the convergence of the sample average to the steady-state mean; see Law (2015, Figure 9.2). The two outputs of interest are (i) the steady-state mean waiting time $E(w_t \mid t \uparrow \infty) = \mu_w$; (ii) the steady-state 90 % quantile $w_{0.9}$ defined by $P(w_t \leq w_{0.9} | \; t \uparrow \infty) = 0.9$. The classic estimator of this mean is the time-series average $\overline{w} = \sum_{t=1}^{T} w_t/T$. To estimate the 90 % quantile, the (autocorrelated) time series $\mathbf{w}$ is sorted from low to high,* *which gives the (autocorrelated) order statistics $w_{(1)}$, ..., $w_{(T)}$ which give the classic point estimator $\widehat{w}_{0.9} = w_{(\lceil 0.9T \rceil)}$. To observe the sampling variability of $\overline{w}$ and $\widehat{w}_{0.9}$, the experiment uses m replications. So, replication r ($r = 1, \ldots, m$) gives the average waiting time $\overline{w}_r$ and the estimated quantile $\widehat{w}_{0.9;r}$. We expect these averages $\overline{w}_r$ to be normally distributed because of the FCLT. The quantile estimators $\widehat{w}_{0.9;r}$, however, are only asymptotically normally distributed; see Chen (2008) and Hong (2009). The experiment includes results for a "short" simulation run $T = 1{,}000$ and for a "long"run $T = 100{,}000$, with a time unit for the arrival and service times such that the service rate μ equals one. Furthermore, the experiment includes two traffic rates; namely, $\lambda/\mu = 0.5$ and $\lambda/\mu = 0.9$. A higher traffic rate gives stronger autocorrelation so we may then expect nonnormality. To obtain accurate estimates of the true behavior of the simulated outputs, the experiment has $m = 1{,}000$ replications. We shall continue this example in the next subsection, testing the normality of $\overline{w}$ and $\widehat{w}_{0.9}$.*

In summary, a limit theorem may explain why random simulation outputs are asymptotically normally distributed. Whether the actual simulation run is long enough, is always hard to know. Therefore it seems good practice to check whether the normality assumption holds—as we explain in the next subsection.

3.3.2 Testing the Normality Assumption

In this subsection we again focus on random simulation, but in Eq. (3.4) we shall consider the residuals of deterministic and random simulations. In general, to test whether a set of observations has a specific probability density function (PDF) (e.g., a Gaussian PDF), we may use various *residual plots* and *goodness-of-fit statistics* such as the chi-square, Kolmogorov-Smirnoff, Anderson-Darling, and Shapiro-Wilk statistics. These plots and statistics can also be generated through software that is available as an add-on to simulation or statistics software.

Note: For details we refer to basic statistics textbooks and articles such as Alba Fernández et al. (2014), Arcones and Wang (2006), Gel et al. (2007), Jimenez-Gamero and Kim (2015), and Jurečková and Picek (2007); we also refer to simulation textbooks such as Kleijnen (1987) and Law (2015) and articles such as Strang (2012) and Tofallis (2008).

Note: Instead of testing whether the output distribution is Gaussian, Montevechi et al. (2010) tests—through a chi-square statistic—whether the distribution is Poisson, because the output of interest is the number of units produced per month in a factory. Turner et al. (2013) considers several other outputs of interest; e.g., the sample variance s_w^2 which has the χ^2_{n-1} distribution if w has a Gaussian distribution; we notice that as n increases, the χ^2_{n-1} distribution converges to a Gaussian distribution.

Run-length T	1,000		100,000	
Traffic rate λ/μ	0.5	0.9	0.5	0.9
p for average	<0.01	<0.01	>0.15	0.11
p for 0.9 quantile	<0.01	<0.01	>0.15	0.116

TABLE 3.1. p-values for Kolmogorov-Smirnov test of normality

A basic assumption of goodness-of-fit tests is that the observations are identically and independently distributed (IID). We may therefore obtain "many" (say, $m = 100$) replications for a specific input combination (e.g., the base scenario) if the simulation is not computationally expensive. However, if a single simulation run takes relatively much computer time, then only "a few" (say, $2 \leq m \leq 10$) replications are feasible, so the plots are too rough and the goodness-of-fit tests lack power. (To obtain more observations on an expensive simulation in an inexpensive way, we may bootstrap a goodness-of-fit test; see Cheng (2006a) and Sect. 3.3.5 below.)

Example 3.4 *We continue Example 3.3, in which $m = 1{,}000$ replications are obtained for the M/M/1 simulation model. Kleijnen and Van Beers (2013) tests the goodness-of-fit through the chi-square and the Kolmogorov-Smirnov tests. Both tests give similar results. For the Kolmogorov-Smirnov test, Table 3.1 displays the resulting p-values; we discussed p-values below Eq. (2.21). The p-values reported in this table imply that the estimated average and quantile are not normally distributed if the simulation run is only $T = 1{,}000$—even for a traffic rate as low as $\lambda/\mu = 0.5$.*

Actually, the white-noise assumption concerns the metamodel's *residuals* e, not the simulation model's outputs w. The estimated residuals $\widehat{e}_i = \widehat{y}_i - w_i$ with $i = 1, \ldots, n$ were defined in Eq. (2.11), and an alternative definition $\widehat{\overline{e}}_i = \widehat{y}_i - \overline{w}_i$ was given in Eq. (2.25); these two definitions coincide if there are no replications—as is the case in deterministic simulation and passive observation of real systems (e.g., in econometrics). We, however, assume that we obtain at least a few replications for each input combination. For simplicity of presentation, we further assume that the number of replications is constant so $m_i = m$. If the simulation outputs w have a constant variance σ_w^2, then $\sigma_{\overline{w}}^2 = \sigma_w^2/m$ is also constant. Unfortunately, even if the average simulation outputs have a constant variance $\sigma_{\overline{w}}^2$ and are independent (no CRN), then it can be proven that the *estimated* residuals $\widehat{\overline{\mathbf{e}}}$ do not have a constant variance and are not independent; i.e., the covariance matrix of $\widehat{\overline{\mathbf{e}}}$ is given by

$$\mathbf{\Sigma}_{\widehat{\overline{\mathbf{e}}}} = [\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}']\sigma_{\overline{w}}^2 \tag{3.4}$$

where $\mathbf{X}$ is the $n \times q$ matrix of explanatory regression variables; also see Eq. (3.50) below. Eq. (3.4) uses the so-called *hat matrix* $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$, which we shall also use in Eq. (3.48).

Simulation examples of normality testing through *visual inspection* of residual plots are Ayanso et al. (2006) and Noguera and Watson (2006); such plots are standard output of many statistical packages.

3.3.3 Normalizing Transformations

We may transform a simulation output (say) w, in order to obtain (approximately) normally distributed output v; e.g., $v = \log(w)$ may be more normally distributed than the original simulation output w. Actually, this logarithmic transformation is a special case of the *Box-Cox power transformation* defined by

$$v = \frac{w^{\lambda} - 1}{\lambda} \text{ if } \lambda \neq 0; \text{ else } v = \ln(w) \tag{3.5}$$

where λ is estimated from the original simulation output data; λ is the classic symbol in this transformation, and has nothing to do with the arrival rate in M/M/1 models. A complication is that the metamodel now explains the behavior of the transformed output—not the original output! We shall return to the Box-Cox transformation when discussing transformations for variance stabilization in Sect. 3.4.3. For further discussion we refer to Atkinson and Riani (2000), Bekki et al. (2009), Cho and Loh (2006), Freeman and Modarres (2006) and Spöck and Pilz (2015).

If the actual distribution has "fatter" tails than the normal distribution has, then *outliers* may occur more frequently. We may then apply *robust regression analysis*; see again Atkinson and Riani (2000) and also Renaud and Victoria-Feser (2010). However, we are not aware of any applications of such an analysis in simulation.

3.3.4 Jackknifing

Jackknifing—or the jackknife—is a general statistical method for solving the following two types of problems:

1. *Confidence intervals* (CIs) for nonnormal responses
2. *Biased* estimators.

Examples of nonnormal simulation outputs (see problem type 1) are the estimated service rate close to 100 % in inventory models, and quantiles such as the 0.90 quantile in Table 3.1 and quantiles such as the 0.95 quantile in production planning detailed in Kleijnen et al. (2011). Examples of biased estimators (see problem type 2) will follow in Sect. 3.4; see Eq. (3.24).

Note: Jackknifing was proposed by Quenouille in 1949 for bias reduction, and by Tukey in 1969 for CI construction; see the classic review article on jackknifing, Miller (1974). A recent publication on jackknifing is Chen and Yu (2015).

To explain jackknifing, we use the following linear regression problem. Suppose we want CIs for the q individual regression coefficients in $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)'$ in case the simulation output has a very nonnormal distribution. The linear regression metamodel is still given by Eq. (2.10). For simplicity, we assume that each input combination i $(i = 1, \ldots, n)$ is replicated an equal number of times $m_i = m > 1$. The original OLS estimator (see again Eq. (2.13)) is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}}. \tag{3.6}$$

The jackknife deletes replication r $(r = 1, \ldots, m)$ for each input combination i, and computes the estimator

$$\widehat{\boldsymbol{\beta}}_{-r} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}}_{-r} \quad (r = 1, \ldots, m) \tag{3.7}$$

where the n-dimensional vector $\overline{\mathbf{w}}_{-r} = (\overline{w}_{1;-r}, \ldots, \overline{w}_{i;-r}, \ldots, \overline{w}_{n;-r})'$ has as component i the average of the $m - 1$ replications after deleting replication r; i.e.,

$$\overline{w}_{i;-r} = \frac{\sum_{r' \neq r}^{m} w_{i;r'}}{m-1} \tag{3.8}$$

where for the case $r = m$ the summation runs from 1 to $m - 1$ (not m) (a more elegant but more complicated mathematical notation is possible).

The m estimators $\widehat{\boldsymbol{\beta}}_{-1}$, …, $\widehat{\boldsymbol{\beta}}_{-m}$ in Eq. (3.7) are correlated because they share $m - 2$ elements. For ease of presentation, we focus on β_q (last element of $\boldsymbol{\beta}$). Jackknifing uses the *pseudovalue* usually denoted by J, which is defined as the following weighted average of $\widehat{\beta}_q$ (the original estimator) and $\widehat{\beta}_{q;-r}$ (element q of the jackknifed estimator $\widehat{\boldsymbol{\beta}}_{-r}$ defined in Eq. (3.7)) with the number of observations as weights:

$$J_r = m\widehat{\beta}_q - (m-1)\widehat{\beta}_{q;-r}. \tag{3.9}$$

In this example, both the original and the jackknifed estimators are unbiased, so the pseudovalues also remain unbiased estimators. Otherwise, it can be proven that the bias is reduced by the *jackknife point estimator*

$$\overline{J} = \frac{\sum_{r=1}^{m} J_r}{m}, \tag{3.10}$$

which is simply the average of the m pseudovalues J_r in Eq. (3.9).

To compute a CI, jackknifing treats the pseudovalues as if they were NIID:

$$P(\overline{J} - t_{m-1;1-\alpha/2}\widehat{\sigma}_{\overline{J}} < \beta_q < \overline{J} + t_{m-1;1-\alpha/2}\widehat{\sigma}_{\overline{J}}) = 1 - \alpha \tag{3.11}$$

where $t_{m-1;1-\alpha/2}$ denotes the $1-\alpha/2$ quantile (upper $\alpha/2$ point) of the distribution of the t-statistic with $m-1$ degrees of freedom, and

$$\widehat{\sigma}_{\overline{J}} = \sqrt{\frac{\sum_{r=1}^{m}(J_r - \overline{J})^2}{m(m-1)}}. \tag{3.12}$$

We can use the CI defined in Eq. (3.11) to test whether the true regression parameter is zero; see the null-hypothesis in Eq. (2.21).

Applications of jackknifing in simulation are numerous. For example, Gordy and Juneja (2010) applies jackknifing for the estimation of large loss probabilities in financial "portfolio risk management" simulation. Kleijnen et al. (1987) applies jackknifing to obtain a CI for a regression estimator that uses the estimated covariance matrix of the simulation output, so the estimator becomes nonlinear; see Eq. (3.24). Kleijnen et al. (1989) applies jackknifing to reduce the bias and compute CIs for a variance reduction technique (VRT) called "control variates" or "regression sampling". Jackknifing may also be applied in the renewal analysis of steady-state simulation; this analysis uses ratio estimators, which are known to be biased; see Kleijnen and Van Groenendaal (1992, pp. 202–203).

Exercise 3.1 *Apply jackknifing to derive a CI for the average waiting time of the first c customers arriving into an M/M/1 system with a traffic rate of* 0.8. *Vary c between* 10 *(terminating simulation) and* 10^7 *(steady-state simulation), and vary m (number of replications) between* 10 *and* 10^2. *Do these CIs cover the analytical steady-state value?*

Exercise 3.2 *Apply jackknifing to derive a CI for the slope* β_1 *in the simple regression model* $w_{ir} = \beta_0 + \beta_1 x_i + e_{ir}$ *where* e_{ir} *is nonnormally distributed (*$i = 1, \ldots, n$*;* $r = 1, \ldots, m$*), e.g.,* e_{ir} *has a lognormal distribution shifted such that* e_{ir} *has zero mean. Design a Monte Carlo experiment with* $\beta_0 = 0$ *and* $\beta_1 = 1$, $x_1 = 1$ *and* $x_2 = 2$ *(so* $n = 2$*),* $m = 5$ *and* $m = 25$, *respectively and 1,000 macroreplications; sample* e_{ir} *from a lognormal distribution with standard deviation* $\sigma_e = 0.1$ *and shifted such that* $E(e) = 0$.

Note: Jackknifing resembles cross-validation, in the sense that both methods drop observations; i.e., jackknifing deletes replication r ($r = 1, \ldots, m$), whereas leave-one-out cross-validation deletes I/O combination i from the complete set of n combinations which gives the remaining I/O data set $(\mathbf{X}_{-i}, \overline{\mathbf{w}}_{-i})$ (we shall detail cross-validation in Sect. 3.6.2).

Actually, the jackknife is a linear approximation of the bootstrap; see Efron and Tibshirani (1993). We discuss bootstrapping in the next subsection. Under the name "resampling techniques". MATLAB offers software for bootstrapping, jackknifing, and cross-validation.

3.3.5 Bootstrapping

Like jackknifing, bootstrapping—or the bootstrap—is a general statistical method that does not assume normality. Like jackknifing, bootstrapping has become popular since powerful and cheap computers have become available. Bootstrapping is well-suited for parallel computers, as we shall see. Moreover, special bootstrap software is available in many statistical software packages; e.g., "R boot" in the R package, BOOT in SAS, and the "bootstrap" command in S-Plus. For further discussion of such statistical software we refer to Novikov and Oberman (2007).

There are two types of bootstrapping; namely,

- distribution-free or nonparametric bootstrapping and
- parametric bootstrapping.

In this section we focus on distribution-free bootstrapping, because parametric bootstrapping is simply *Monte Carlo* sampling with the parameters of the assumed distribution being estimated from the available original data; we shall present several examples of parametric bootstrapping throughout this book.

Note: Efron (1982) is a famous monograph on jackknifing and bootstrapping. Efron and Tibshirani (1993) is the classic textbook on bootstrapping. Other textbooks on bootstrapping—a resampling method, as we shall see—are Chernick (2007), Davison and Hinkley (1997), Good (2005), Horowitz (2001), and Lunneborg (2000). Interesting articles on bootstrapping are Cheng (2006a,b), Davidson and MacKinnon (2007), Ghosh and Polansky (2014), Kreiss and Paparoditis (2011), Mammen and Nandi (2012), and Martínez-Camblor and Corral (2012). Furthermore, Efron (2011) discusses bootstrapping in Bayesian inference, as an alternative for the "Markov chain Monte Carlo" (MCMC) method; MCMC is also applied in Goldberg et al. (1998). More references will follow below.

Bootstrapping may be used to solve two types of problems:

1. Nonnormal distributions
2. Nonstandard statistics.

Sub 1: Nonnormal distributions

Let us consider the same example as we considered for jackknifing; i.e., we want a CI for the regression coefficients in case of nonnormal simulation output $\mathbf{w}$. Again we assume that each of the n input combinations is replicated an equal number of times, $m_i = m > 1$ $(i = 1, \ldots, n)$.

When we bootstrap, we distinguish between the *original observations* w and the *bootstrapped observations* w^*; the superscript $*$ is the standard notation for the bootstrapped observations. Standard bootstrapping assumes that the original observations are IID. In the example, there are $m_i = m$

IID original simulated observations for input combination i; namely, $w_{i;1}$, …, $w_{i;m}$. These observations give the average simulation output for combination i; namely, $\overline{w}_i$. These averages give the n-dimensional vector $\overline{\mathbf{w}}$, which determines the original OLS estimator in Eq. (3.6).

Most theory on bootstrapping assumes *univariate* output such as w, but we shall also discuss distribution-free and parametric bootstrapping for *multivariate* (vector) output $\mathbf{w}$ created through CRN; see Sect. 3.5 (and also Chap. 6).

Note: Bootstrapping of time series (which violates the IID assumption) is discussed in Cheng (2006b), Hsieh and Chen (2007), Lahiri (2003), Martin (2007), Park et al. (2001), and Psaradakis (2006).

In general, we obtain the bootstrap observations through *resampling with replacement* from the original observations. In the example, this resampling may result in the original observation $w_{i;1}$ being sampled m times, and—because the sample size is kept constant, at m—all the other $m-1$ original observations $w_{i;2}, \ldots, w_{i;m}$ being sampled zero times. Obviously, this specific sampling outcome has low probability, but it is not impossible. Resampling in this example implies that the bootstrapped observations $w^*_{i;1}$, …, $w^*_{i;m}$ occur with frequencies f_1, …, f_m such that $f_1 + \ldots + f_m = m$.

Note: These frequencies f_1, …, f_m follow the multinomial (or polynomial) distribution with parameters m and $p_1 = \ldots = p_m = 1/m$; the multinomial distribution is discussed in many statistics textbooks.

We do this resampling for each combination i $(i = 1, \ldots, n)$. The resulting bootstrapped outputs $w^*_{i;1}$, …, $w^*_{i;m}$ give the bootstrapped average simulation output $\overline{\mathbf{w}}^*$. Substitution of this $\overline{\mathbf{w}}^*$ into Eq. (3.6) gives the bootstrapped OLS estimator

$$\widehat{\boldsymbol{\beta}}^* = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}}^*. \tag{3.13}$$

To reduce sampling variation or "sampling error", we repeat this resampling (say) B times; B is known as the *bootstrap sample size.* A typical value for B is 100 or 1,000. This sample size gives $\widehat{\boldsymbol{\beta}}^*_1$, …, $\widehat{\boldsymbol{\beta}}^*_B$, which we may also denote as $\widehat{\boldsymbol{\beta}}^*_b$ with $b = 1, \ldots, B$. Obviously, bootstrapping is well-suited for parallel computers.

Let us again focus on the single regression parameter β_q, as we did in the jackknife example. In practice, the most popular CI uses the so-called *percentile method*:

$$P(\widehat{\beta}^*_{q(\lfloor B\alpha/2)\rfloor)} < \beta_q < \widehat{\beta}^*_{q(\lfloor B(1-\alpha/2)\rfloor)}) = 1 - \alpha \tag{3.14}$$

where the left endpoint of the interval $\widehat{\beta}^*_{q(\lfloor B\alpha/2)\rfloor)}$ is the (lower) $\alpha/2$ quantile of the empirical density function (EDF) of the bootstrap estimate $\widehat{\beta}^*_q$ so this EDF is obtained through sorting the B values of the bootstrap estimate $\widehat{\beta}^*_q$ from low to high; analogously, $\widehat{\beta}^*_{q(\lfloor B(1-\alpha/2)\rfloor)}$ is the upper limit of the interval. If we wish CIs that hold simultaneously for all q regression parameters, then we can apply Bonferroni's inequality replacing α by

α/q in Eq. (3.14). This inequality implies that the classic type-I error rate (in this case $\alpha/2$) is replaced by the same value divided by the number of CIs (in this case q)–resulting in the "experimentwise" or "familywise" type-I error rate α. Obviously, the qCIs are highly correlated. References on Bonferroni's inequality are given in Gordon (2007); alternatives for this conservative inequality are discussed in Döhler (2014).

Note: Besides the percentile method there are alternative procedures that are detailed in the bootstrap literature; see the double bootstrap in Cheng (2006b) and the smoothed and iterated bootstrap methods in Ghosh and Polansky (2014). If we wish simultaneous CIs for all q regression parameters, then we may replace Bonferroni's inequality and obtain tighter CIs through an algorithm detailed in Mandel and Betensky (2008). Furthermore, Ghosh and Polansky (2014) discusses ellipsoidal confidence regions for a vector of parameters, which are generalizations of symmetric CIs for a single parameter. Our description of bootstrapping explains why Godfrey (2006) describes bootstrapping as "an artificial bootstrap world is constructed, conditional on the observed data".

We can apply bootstrapping in many situations where classic statistics do not seem appropriate. For example, Kleijnen et al. (2001) applies bootstrapping to validate so-called "trace-driven" simulation models when the test statistic is the difference between the average outputs of the real system and the simulated system, and these two averages are not normally distributed. Turner et al. (2013) applies bootstrapping to estimate a CI for s_w^2 (sample variance of w) if w does not have a Gaussian distribution. Jimenez-Gamero and Kim (2015) applies bootstrapping to solve the problem of limited data in production control.

Exercise 3.3 *Analogously to Exercise 3.1, apply bootstrapping to derive a CI for the average waiting time of the first c customers arriving into the M/M/1 system with a traffic rate of* 0.8*. Vary c between* 10 *(terminating simulation) and* 10^7 *(steady-state simulation), and m (number of replications) between* 10 *and* 10^2*. Does this CI cover the analytical steady-state value?*

Exercise 3.4 *Analogously to Exercise 3.2, apply bootstrapping to derive a CI for the slope* β_1 *in the simple regression model* $y = \beta_0 + \beta_1 x + e$ *where e is nonnormally distributed (*$i = 1, \ldots, n$*;* $r = 1, \ldots, m$*), e.g., e has a lognormal distribution shifted such that e has zero mean. To evaluate this bootstrapping, design a Monte Carlo experiment with* $\beta_0 = 0$ *and* $\beta_1 = 1$*,* $x_1 = 1$ *and* $x_2 = 2$ *(so* $n = 2$*),* $m = 5$ *and* $m = 25$*, respectively and 1,000 macroreplications; sample e from a lognormal distribution with standard deviation* $\sigma_e = 0.1$ *and shifted such that* $E(e) = 0$*.*

In *expensive* simulation there may be so few replications (e.g., $m = 1$ or $m = 2$) that *distribution-free bootstrapping* does not work; i.e., resampling with replacement gives the same result "many" times. We may then apply *parametric bootstrapping*, as follows. We assume a specific type of distribution; e.g., a Poisson distribution. From the original data we estimate the parameters of this distribution; e.g., the parameter λ of the assumed Poisson distribution. Next we use PRNs to sample bootstrapped observations from the resulting distribution; e.g., the Poisson distribution with parameter $\widehat{\lambda}$. Altogether, we conclude that parametric bootstrapping is a Monte Carlo experiment with parameters estimated from the original data. Chang et al. (2010) discusses the comparison of several means through parametric bootstrapping. We shall give examples in Sects. 5.2 and 6.2.5.

Sub 2: Nonstandard statistics

Classic statistics such as t and F have tables with critical values that provide CIs, assuming Gaussian parent distributions; e.g., t_{n-1} is computed from normally distributed x_i with $i = 1, \ldots, n$. If this normality assumption does not hold, then we may apply bootstrapping as we explained sub 1. We may also be interested in statistics such as R^2, for which there are no tables with critical values. Kleijnen and Deflandre (2006) bootstraps R^2 to test the validity of regression metamodels in simulation. Sadooghi-Alvandi and Malekzadeh (2014) uses parametric bootstrapping for the ratios of the means of lognormal distributions; i.e., nonstandard statistics are considered for nonnormal distributions.

As we have already mentioned in our discussion of parametric bootstrapping for nonnormal distributions, in expensive simulation with only a few replications we may apply parametric bootstrapping. For example, in case of a nonstandard statistic we assume a specific type of distribution; e.g., a Gaussian distribution. From the original data we estimate the parameters of this distribution; e.g., the mean and the variance of the assumed Gaussian distribution. Next we sample bootstrapped observations from the resulting distribution. From these bootstrapped observations we compute the nonstandard statistic. We repeat this bootstrapping B times, etc.

Note: A special family of distributions is the "generalized gamma distribution". Bootstrapping such a distribution may give better results than distribution-free bootstrapping if we wish to estimate extreme quantiles; see Wang et al. (2010).

We emphasize that using bootstrapping to test a *null-hypothesis* (e.g., $H_0 : E(e) = 0$ or $H_0 : \beta_q = 0$) requires some more care than estimating a CI for some parameter (e.g., β_q). Indeed, Shao and Tu (1995, p. 189) warns: "bootstrap hypothesis testing . . . is not a well-developed topic."

Note: Further discussion of hypothesis testing versus CI estimation in bootstrapping is found in Martin (2007), Paparoditis and Politis (2005), and Racine and MacKinnon (2007). Examples of bootstrapping for testing the null-hypothesis of a valid simulation model or the null-hypothesis of a valid regression metamodel, are found in Kleijnen et al. (2001) and Kleijnen and Deflandre (2006).

In general, it is better not to bootstrap the original statistic of interest but the *pivotal* statistic, which (by definition) has a distribution that does not depend on unknown nuisance parameters. For example, the sample average $\overline{x}$ has the distribution $N(\mu, \sigma^2/n)$ with the unknown nuisance parameter σ, whereas the Studentized statistic $(\overline{x} - \mu)/(s/\sqrt{n})$ has a t_{n-1}-distribution, which does not depend on σ so the latter statistic is pivotal. Instead of bootstrapping $\widehat{\beta}_q$ in Eq. (3.14), it is better to bootstrap the studentized version $\widehat{\beta}_q \ /s(\widehat{\beta}_q)$; also see Eq. (2.19).

Note: Further discussion of bootstrapping pivotal statistics is found in Cheng (2006b), Paparoditis and Politis (2005), Racine and MacKinnon (2007), and Sadooghi-Alvandi and Malekzadeh (2014).

3.4 Heterogeneous Output Variances

A *deterministic* simulation model gives a single fixed value for a given input combination, so it has a zero output variance—given a fixed input combination. We often assume a normal distribution for the residuals of the metamodel fitted to the I/O data of the deterministic simulation model; also see the discussion at the beginning of Sect. 3.3.1. Usually, we then assume a normal distribution with a *constant* variance. Actually, we do not know a better assumption that works in practice, for deterministic simulation.

In the rest of this section we focus on *random* simulation models. We try to answer the following questions (formulated more generally in Sect. 3.1):

1. How *realistic* is the constant variance assumption?
2. How can this assumption be *tested*?
3. How can the simulation output be *transformed* such that the constant variance assumption holds?
4. Which statistical *analysis* methods can be applied that allow nonconstant variances?
5. Which statistical *design* methods can be applied that allow nonconstant output variances?

3.4.1 Realistic Constant Variance Assumption?

In practice, random simulation outputs usually do not have constant variances as input combinations change. For example, in the M/M/1 queueing simulation not only the expected value (first moment) of the steady-state waiting time changes as the traffic rate changes—the variance (central second moment) of this output changes even more; see Cheng and Kleijnen (1999) and Cheng et al. (2000). Variance heterogeneity is also discussed in Yang et al. (2007) for cycle time-throughput (CT-TH) curves, which quantify the relationship of long-run average cycle time to the throughput rate in manufacturing systems. Variance heterogeneity is discussed in Montevechi et al. (2010) for the output that follows a Poisson distribution; it is well-known that a Poisson distribution has a variance that changes as the mean changes.

3.4.2 Testing for Constant Variances

It may be a priori certain that the variances of the simulation outputs are not constant at all, as the previous subsection demonstrated; in some applications, however, we may hope that the variance heterogeneity is negligible. Unfortunately, the output variances are unknown so we must estimate them. If there are m_i replications, then the classic unbiased variance estimator s^2 of σ^2 follows from Eq. (2.27). We point out that this estimator itself has high variance; i.e., using the classic assumption of normally distributed output, any statistics textbook mentions that $(m-1)s^2$ has a chi-square or χ^2 distribution with $m-1$ degrees of freedom, and

$$\sigma^2_{s^2} = \frac{2\sigma^4}{m}.$$

Given the same assumption, we may compare two independent variance estimators (say) s_1^2 and s_2^2 through

$$F_{m_1-1.m_2-1} = \frac{(m_1-1)s_1^2}{(m_2-1)s_2^2}.$$

In practice, simulation experiments have n combinations of k inputs (with $k < n$), so we need to compare n variance estimators s_i^2 $(i = 1, \ldots, n)$. This problem may be solved in many different ways; e.g., Kleijnen (1987, p. 225) speaks of approximately 60 different tests. Here we mention only three of these tests.

1. Hartley (1950) presents the maximum F-ratio:

$$F_{max} = \frac{max\ (s_i^2)}{min\ (s_i^2)}. \tag{3.15}$$

2. Scheffé (1964) proposes analysis of variance (ANOVA), treating the data as an experiment with a single input and n levels (values). However, the s_i^2 have χ^2 distributions whereas ANOVA assumes normality. Therefore we may apply a normalizing transformation such as the Box-Cox transformation defined in Eq. (3.5). Details are given in Scheffé (1964).

3. Conover (1999) gives a distribution-free test; for details we again refer to that publication.

Exercise 3.5 *Apply bootstrapping to derive the distribution of Hartley's statistic defined in Eq. (3.15) for the following simple case:* $w_{i;r} \sim NID$ (μ_i, σ_i^2) $(i = 1, \ldots, n; r = 1, \ldots, m)$ *with* $n = 3$ *and homogeneous variances so* $\sigma_i^2 = \sigma^2$*. Design a Monte Carlo experiment with* $\mu_i = 0$ *and* $\sigma^2 = 1$, $m = 25$*, and 1,000 macroreplications. Repeat the experiment for heterogeneous variances* $(\sigma_i^2 \neq \sigma^2)$*. Repeat for nonnormally distributed* $w_{i;r}$*.*

3.4.3 Variance Stabilizing Transformations

The logarithmic transformation—which is a special case of the *Box-Cox* transformation in Eq. (3.5)—may be used not only to obtain Gaussian output but also to obtain outputs with constant variances. Montevechi et al. (2010) applies the so-called *Johnson* transformation to solve the problem of nonnormality and variance heterogeneity of the original output with a Poisson distribution; this transformation is detailed in Yeo and Johnson (2000). A problem may again be that the metamodel now explains the transformed output instead of the original output. To solve this problem, Irizarry et al. (2003) proposes the so-called "MLE-delta method", which gives asymptotically exact CIs for the original metamodel. We give no details on this method, because we prefer accepting heterogeneous variances and adapting our analysis—as we detail in the next subsection (Sect. 3.4.4).

3.4.4 Least Squares Estimators

In case of heterogeneous output variances, the LS criterion still gives an *unbiased* estimator $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$. To prove this lack of bias, it suffices to assume that the residuals have zero mean so $E(\mathbf{e}) = 0$; see again the solution of Exercise 2.2.

The *variance* of $\hat{\boldsymbol{\beta}}$, however, is no longer given by Eq. (2.18). Actually, this variance is given by the main diagonal of the covariance matrix that follows from Eq. (2.17):

$$\mathbf{\Sigma}_{\hat{\boldsymbol{\beta}}} = (\mathbf{X}_N'\mathbf{X}_N)^{-1}\mathbf{X}_N'\mathbf{\Sigma}_{\mathbf{w}}\mathbf{X}_N(\mathbf{X}_N'\mathbf{X}_N)^{-1} \tag{3.16}$$

where $\mathbf{X}_N$ is an $N \times q$ matrix with $N = \sum_{i=1}^{n} m$. We give the following comments on this equation.

- $\mathbf{\Sigma}_{\mathbf{w}}$ in the right-hand side of this equation is a *diagonal* matrix if the simulation outputs have different variances σ_i^2 $(i = 1, \ldots, n)$ but no CRN are used so these outputs are independent.

- If there are *no* replications, then $\mathbf{X}_N$ becomes the $n \times q$ matrix $\mathbf{X}_n$, and $\mathbf{\Sigma}_{\mathbf{w}}$ becomes an $n \times n$ matrix with element i on its main diagonal equal to σ_i^2 $(i = 1, \ldots, n)$.

- If input combination i is replicated m_i times, then $\mathbf{X}_N$ is $N \times q$ so $\boldsymbol{\Sigma}_{\mathbf{w}}$ is also an $N \times N$ matrix with the first m_1 elements on its main diagonal all equal to σ_1^2, …, the last m_n elements on its main diagonal equal to σ_n^2.
- If the number of replications is constant $(m_i = m)$, then

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}_n'\mathbf{X}_n)^{-1}\mathbf{X}_n'\overline{\mathbf{w}} \tag{3.17}$$

where $\overline{\mathbf{w}}$ denotes the vector with the n simulation outputs averaged over the m replications; see $\overline{w}_i$ in Eq. (2.28) with $m_i = m$.

Exercise 3.6 *Prove that Eq. (3.16) (general formula for the covariance matrix of the LS estimator) reduces to the classic formula in Eq. (2.17) if* $\boldsymbol{\Sigma}_{\mathbf{w}} = \sigma_w^2\mathbf{I}$.

Kleijnen (1992) examines CIs for the q individual OLS estimators in Eq. (3.17). Their standard errors follow from the main diagonal of the following corrected covariance matrix, which is the analogue of Eq. (3.16):

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}. \tag{3.18}$$

CIs may be computed through a t-statistic with $m-1$ degrees of freedom. We shall present an alternative method that does not require the estimation of $\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}$ in Eq. (3.18); see Eq. (3.33) below. One more alternative is presented in Wen et al. (2007).

Though the OLS estimator $\hat{\boldsymbol{\beta}}$ remains unbiased, it is no longer the BLUE. It can be proven that the BLUE is now the *weighted LS* (WLS) estimator, which we denote through a tilde instead of a hat:

$$\widetilde{\boldsymbol{\beta}} = (\mathbf{X}'{}_N^{\prime}\boldsymbol{\Sigma}_{\mathbf{w}}^{-1}\mathbf{X}_N)^{-1}\mathbf{X}_N'\boldsymbol{\Sigma}_{\mathbf{w}}^{-1}\mathbf{w}. \tag{3.19}$$

If the number of replications is constant such that $m_i = m$, then we may write analogously to Eq. (3.17):

$$\widetilde{\boldsymbol{\beta}} = (\mathbf{X}^{'}\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}^{-1}\mathbf{X})^{-1}\mathbf{X}'\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}^{-1}\overline{\mathbf{w}} \tag{3.20}$$

where $\mathbf{X}$ is an $n \times q$ matrix and $\boldsymbol{\Sigma}_{\overline{\mathbf{w}}} = \boldsymbol{\Sigma}_{\mathbf{w}}/m$ where $\boldsymbol{\Sigma}_{\mathbf{w}}$ is an $n \times n$ matrix The covariance matrix of the WLS estimator can be proven to be

$$\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{\beta}}} = (\mathbf{X}'\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}^{-1}\mathbf{X})^{-1}. \tag{3.21}$$

If we have access to classic OLS software only, then we may compute the WLS estimator through that software replacing the original I/O data $(x_{i;j}, w_i)$ by $(x_{i;j}/\sigma_i, w_i/\sigma_i)$ where σ_i denotes the standard deviation of w_i $(i = 1, \ldots, n$ and $j = 1, \ldots, q)$. Obviously, these transformed outputs have a constant variance, which has the value 1. It can be proven that WLS minimizes the sum of squared residuals weighted with $1/\sigma_i^2$.

In practice, $\boldsymbol{\Sigma}_{\mathbf{w}}$ is unknown so we must estimate this covariance matrix. We distinguish two types of situations (as we did in the preceding chapter):

1. passive observation of a real system or active experimentation with a deterministic simulation model: no replications
2. active experimentation with a real system or a random simulation model: replications.

In case of passive observation of a real system (included in type 1), we may estimate $\boldsymbol{\Sigma}_{\mathbf{w}}$ from the residuals; see any econometrics textbook or see again Godfrey (2006). In type-2 situations, we may estimate σ_i^2 through Eq. (2.27). In the rest of this subsection, we focus on the latter type of situations.

Substituting the estimated output variances s_i^2 into the main diagonal of $\boldsymbol{\Sigma}_{\mathbf{w}}$ gives $\widehat{\boldsymbol{\Sigma}}_{\mathbf{w}}$. Next we substitute this $\widehat{\boldsymbol{\Sigma}}_{\mathbf{w}}$ into the classic WLS estimation formula; namely, Eq. (3.19). This gives what we call the *estimated WLS* (EWLS), which is also known as the *Aitken* estimator. For a constant number of replications this EWLS estimator is

$$\widehat{\widehat{\boldsymbol{\beta}}} = (\mathbf{X}'\widehat{\boldsymbol{\Sigma}}_{\overline{w}}^{-1}\mathbf{X})^{-1}\mathbf{X}'\widehat{\boldsymbol{\Sigma}}_{\overline{w}}^{-1}\overline{\mathbf{w}}. \tag{3.22}$$

The EWLS defined in Eq. (3.22) is a *nonlinear* estimator. Consequently, the statistical analysis becomes more complicated. For example, the covariance matrix of the EWLS estimator does no longer follow from Eq. (2.17). The analogue of Eq. (3.21) holds only asymptotically:

$$\boldsymbol{\Sigma}_{\widehat{\widehat{\boldsymbol{\beta}}}} \approx (\mathbf{X}'\boldsymbol{\Sigma}_{\overline{\mathbf{w}}}^{-1}\mathbf{X})^{-1}; \tag{3.23}$$

see Arnold (1981), Godfrey (2006), and Kleijnen et al. (1985). CIs are no longer similar to Eq. (2.19). We have already presented relatively simple solutions for this type of problems; namely, jackknifing and bootstrapping (see the Sects. 3.3.4 and 3.3.5). For EWLS we may apply these two techniques as follows.

Jackknifed EWLS—or JEWLS—is detailed in Kleijnen et al. (1987), assuming a constant number of replications m. In JEWLS we delete replication r of the m replications, and recompute the estimator analogously to the jackknifed OLS estimator in Eq. (3.7):

$$\widehat{\widehat{\boldsymbol{\beta}}}_{-r} = (\mathbf{X}\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}};-r}^{-1}\mathbf{X})^{-1}\mathbf{X}'\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}};-r}^{-1}\overline{\mathbf{w}}_{-r} \quad (r = 1, \ldots, m) \tag{3.24}$$

where $\overline{\mathbf{w}}_{-r}$ is the vector with the n averages of the $m-1$ replications after deleting replication r, and $\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}};-r}$ is computed from the same replications. From these $\widehat{\widehat{\boldsymbol{\beta}}}_{-r}$ and the original $\widehat{\widehat{\boldsymbol{\beta}}}$ in Eq. (3.22) we compute the pseudovalues, which give the desired CI.

Bootstrapped EWLS (BEWLS) will be explained in the section on CRN, which leads to estimated generalized LS (EGLS). This BEWLS is applied in Kleijnen and Deflandre (2006); also see Godfrey (2006) and You and Chen (2006).

Altogether, we may compute both the OLS estimate and the EWLS estimate, analyze both estimates, and check whether these estimates give the same qualitative conclusions; e.g., do they suggest that the same inputs are important? We conjecture that in general EWLS gives estimates that are more significant, because EWLS gives smaller standard errors.

Note: Santos and Santos (2011) apply EWLS to estimate polynomials, not only to approximate the expected value but also the standard deviation of the simulation output. This standard deviation is important in robustness, as we shall see in Sect. 6.4.

3.4.5 Designs for Heterogeneous Output Variances

If the output variances σ_i^2 are not constant, then classic designs still give the *unbiased* OLS estimator $\hat{\boldsymbol{\beta}}$ and WLS estimator $\widetilde{\boldsymbol{\beta}}$. The DOE literature pays little attention to the derivation of alternative designs for cases with heterogeneous output variances. An exception is Ceranka et al. (2006), discussing A-optimal designs for heterogeneous variances; unfortunately, that article considers real-life chemical experiments instead of simulation experiments.

Kleijnen and Van Groenendaal (1995) investigates designs with a number of replications m_i of input combination i $(i = 1, \ldots, n)$ such that the estimated variances of the output averages per combination are approximately constant. We summarize that approach as follows. We defined the average $\overline{w}_i$ in Eq. (2.28), so

$$Var(\overline{w}_i) = \frac{\sigma_i^2}{m_i} \quad (i = 1,\ \ldots,\ n).$$

To ensure that $\mathrm{Var}(\overline{w}_i)$ does not vary with i, we select the number of replications such that

$$m_i = c_0 \sigma_i^2 \tag{3.25}$$

where c_0 is a common positive constant such that the m_i become integers and the common variance of the $\overline{w}_i$ becomes $\sigma^2 = 1/c_0$. In other words, the higher the variability of the output w_i is, the more replications we simulate. The allocation of the total number of simulation runs $N = \sum_{i=1}^{n} m_i$ through Eq. (3.25) is not necessarily optimal, but it simplifies the regression analysis and the design of the simulation experiment (an alternative allocation rule replaces the variances σ_i^2 by the standard deviations σ_i). Indeed, in the regression analysis we now apply OLS to the averages $\overline{w}_i$ to obtain the BLUE. In practice, however, we must estimate σ_i^2. A *two-stage* procedure takes a *pilot sample* of size (say) $m_0 \geq 2$ for each input combination, and estimates the variances σ_i^2 through

$$s_i^2(m_0) = \frac{\sum_{r=1}^{m_0} \left[w_{i;r} - \overline{w}_i(m_0)\right]^2}{m_0 - 1} \quad (i = 1,\ \ldots,\ n) \tag{3.26}$$

with

$$\overline{w}_i(m_0) = \frac{\sum_{r=1}^{m_0} w_{i;r}}{m_0}. \tag{3.27}$$

Combining Eqs. (3.26) and (3.25), we select a number of additional replications in the second stage; namely, $\widehat{m}_i - m_0$ with

$$\widehat{m}_i = m_0 \times nint\left[\frac{s_i^2(m_0)}{min\, s_i^2(m_0)}\right] \tag{3.28}$$

where $nint[x]$ denotes the integer closest to x. Obviously, in this second stage we do not obtain additional replications for the combination with the smallest estimated variance, which features in the denominator of Eq. (3.28). After the second stage, we use all $\widehat{m}_i$ replications to compute the average output and its variance. To these averages we apply OLS. We estimate $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}}$ through Eq. (3.16) with $\mathbf{\Sigma}_{\mathbf{w}}$ estimated through a diagonal matrix with diagonal elements $s_i^2(\widehat{m}_i)/\widehat{m}_i$. We base the CIs for the estimated regression parameters on the classic t-statistic with degrees of freedom equal to $m_0 - 1$.

We have the following comments on this design and analysis. The desired number of replications specified in Eq. (3.28) uses a ratio of random variables; in general, such ratios are known to be biased estimators of the true ratios. Moreover, the denominator of Eq. (3.28) is the minimum of n random variables; such a minimum (extreme) is also known to be hard to analyze. The final estimators of the average output and its variance are also ratio estimators, because their denominators involve the random variables $\widehat{m}_i$; see Eqs. (2.28) and (2.27) with m replaced by $\widehat{m}_i$. In general, the statistical literature uses asymptotic analysis for such problems; however, in expensive simulation the actual number of replications is relatively small. Therefore the simulation literature uses Monte Carlo experiments to quantify the performance of allocation rules such as Eq. (3.28).

After the second stage these variance estimates $s_i^2(\widehat{m}_i)/\widehat{m}_i$ may still differ considerably. Therefore, we may replace the two-stage approach by a purely *sequential* approach. In the latter approach we add one replication at a time, until the estimated variances of the average outputs have become practically constant. This sequential procedure may require fewer simulation outputs, but this procedure is also harder to understand, program, and implement.

Exercise 3.7 *Simulate the M/M/1 model, as follows. Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time. Select an experimental area; e.g., the traffic load is 0.3 and 0.5. Fit a first-order polynomial. Use m_i replicated simulation runs; each run should be "sufficiently long". Simulate more replications for the higher traffic rate, using Eq. (3.28). Do not apply CRN for different traffic rates. Now estimate the parameters of the metamodel and predict the simulation output at a 0.4 traffic load including a CI; does this CI cover the analytical solution?*

Hoad et al. (2010) states that there is little guidance for selecting the number of replications, m. That article, therefore, describes a heuristic for determining m in discrete-event simulation, in order to achieve a desired precision for the estimated mean output for an input combination. The same problem is discussed in Law (2015). The solution is based on the following well-known CI:

$$P\left[(\overline{x} - t_{m-1;1-\alpha/2}\frac{s_x}{\sqrt{m}} \leq \mu_x \leq \overline{x} + t_{m-1;1-\alpha/2}\frac{s_x}{\sqrt{m}}\right] = 1-\alpha \qquad (3.29)$$

with estimated mean $\overline{x}$ and estimated standard deviation s_x. The desired precision is the maximum relative error of $\overline{x}$ that we are prepared to tolerate, achieved with probability $1-\alpha$. We may compute the CI in Eq. (3.29) sequentially, and stop as soon as the CI reaches the desired length. Hoad et al. (2010) suggests to start the sequential heuristic with $m_0 = 5$. Test results show that the heuristic can indeed obtain the desired coverage. However, if $\overline{x}$ is close to zero, then the desired relative precision increases m drastically so the relative precision should be replaced by an absolute precision. In practice, a simulation model gives multiple types of output, so we should select m such that the desired precision holds for all these outputs. The heuristic is implemented within the SIMUL8 simulation package. Law (2015) gives the following formula for the desired number of replications $\widehat{m}$ when estimating the mean μ_w with a relative error of γ:

$$\widehat{m} = min\left[r \geq m : \frac{t_{r-1;1-\alpha/2}\sqrt{s_i^2(m)/i}}{|\overline{w}(m)|} \leq \frac{\gamma}{1+\gamma}\right] \qquad (3.30)$$

where the symbols follow from our Eq. (3.26).

Note: Hartung and Knapp (2010) also studies sample-size selection based on Eq. (3.29), but that article focuses on hypothesis testing with a desired power. Turner et al. (2013) uses Monte Carlo experiments to measure the sensitivity of the rule for selecting $\widehat{m}$ to kurtosis and skewness of the distribution of the output w; if the output is the sample mean $\overline{w}$ (as in Eq. (3.30)), then this sensitivity is low; if the output is the sample variance s_w^2, then this sensitivity is high. Brantley et al. (2014) selects the number of replications, assuming a second-order polynomial metamodel per subregion within the total experimental region; the context is simulation optimization through a multiple *ranking and selection* (R&S) procedure with *optimal computer budget allocation* (OCBA), within a Bayesian framework. Pasupathy et al. (2014) also considers a R&S procedure with OCBA. We shall discuss simulation optimization in Chap. 6.

However, heuristics such as Eq. (3.30) select $\widehat{m}$ for a single input combination and a specific performance measure; namely, the estimated mean performance. Consequently, such heuristics do not solve problems that arise in metamodeling. Notice that Turner et al. (2013) also states that $\widehat{m}$ for a single combination does not matter, as long as the metamodel is an accurate approximation. Actually, we think that in metamodeling, the *signal-noise ratio* should guide our selection of $\widehat{m}_i$, the desired number of replications for input combination i. Let us consider a simple example; namely, a first-order polynomial with two inputs; also see Eq. (2.7). If the noise $\sigma_e = \sigma_y = \sigma_w$ is much lower than the signals or first-order effects β_1 and β_2, then a relatively low $\widehat{m}_i$ suffices to estimate which input has the higher effect. Unfortunately, we do not know a general solution for selecting $\widehat{m}_i$ in linear regression metamodels of simulation models. Equation (3.28) only guides the *relative* number of replications $\widehat{m}_i/\widehat{m}_{i'}$, which should converge to $\sigma_i^2/\sigma_{i'}^2$ with i, $i' = 1, \ldots, n$. In practice we also have the additional problem that we do not know which linear regression model is a valid metamodel of the underlying simulation model. Therefore we should save part of our computational budget to test the validity of the estimated metamodel and—if necessary—to increase the number of simulated input combinations n to estimate and validate a more complicated metamodel. Because this problem is not yet solved, we recommend that a simple rule such as Eq. (3.30) should guide the selection of $\widehat{m}_i$. In Chap. 5 (on Kriging) we shall again discuss the choice of $\widehat{m}_i$ $(i = 1, \ldots, n)$ and n.

3.5 Common Random Numbers (CRN)

Deterministic simulation does not use CRN because this type of simulation does not use PRNs. Random simulation often uses CRN, because CRN is the default in software for discrete-event simulation. Intuitively speaking, CRN are meant to compare the outputs of different input combinations while all other "circumstances" are the same; e.g., the average waiting times are compared for one or two servers while the randomness in the customer arrivals is the same. Statistically speaking, CRN are meant to create correlation between $w_{i;r}$ and $w_{i';r}$; these two symbols denote the output of input combination i and i' $(i,\ i' = 1, \ldots, n)$ in the same replication r. If each combination has a common number of replications m, then $r = 1, \ldots, m$; else $r = 1, \ldots, m_0$ with $m_0 = \min\ m_i$. Obviously, we assume that two different replications use nonoverlapping PRN streams, so their outputs $w_{i;r}$ and $w_{i;r'}$ with $r \neq r'$ are independent; i.e., the two vectors of outputs $\mathbf{w}_r$ and $\mathbf{w}_{r'}$ with r, $r' = 1, \ldots, m_0$ are independent. If the number of replications is constant, then we may use the $n \times q$ matrix of explanatory variables $\mathbf{X}$ and the n-dimensional vector with average outputs $\overline{\mathbf{w}}_r = (\overline{w}_{1;r}, \ldots, \overline{w}_{n;r})'$; see Eq. (3.17). If the number of replications is not constant, then we use the $N \times q$ matrix of explanatory variables $\mathbf{X}$ with $N = \sum_{i=1}^{n} m_i$ and the N-dimensional vector with outputs $\mathbf{w}_r = (w_{1;r}, \ldots, w_{N;r})'$ and $r = 1, \ldots, m_i$; i.e., we replicate combination i of the explanatory variables m_i times, and we "staple" the N outputs.

The goal of CRN is to reduce the variance of the estimated regression effects; i.e., to decrease $\text{Var}(\widehat{\beta}_j)$ with $j = 1, \ldots, q$. Actually, CRN increase $\text{Var}(\widehat{\beta}_1)$, the variance of the intercept.

Exercise 3.8 *Prove that Var($\widehat{\beta}_1$) increases if CRN are used and β_1 denotes the intercept; assume that no replications are used so $m = 1$ and that CRN does "work"; i.e., the outputs w_i and $w_{i'}$ are positively correlated.*

So we may use CRN to better explain the input effects, as scenarios are compared under the "same circumstances". CRN are also useful to better predict the output of combinations not yet simulated, provided the lower accuracy of the estimated intercept is outweighed by the higher accuracy of all other estimated effects.

Because CRN violate the classic assumptions of regression analysis, we have two options that are analogous to the options in the case of heterogeneous output variances:

1. Continue to use OLS
2. Switch to GLS

Sub 1: If we continue to use OLS, then we should know that the variance of the OLS estimator $\widehat{\boldsymbol{\beta}}$ is given by Eq. (3.16) but now $\boldsymbol{\Sigma}_{\mathbf{w}}$ is not a diagonal matrix. Assuming a constant number of replications, we may estimate this $\boldsymbol{\Sigma}_{\mathbf{w}}$ analogously to Eq. (2.27):

$$\widehat{\sigma}_{i;i'} = \frac{\sum_{r=1}^{m}(w_{i;r} - \overline{w}_i)(w_{i';r} - \overline{w}_{i'})}{(m-1)}. \tag{3.31}$$

However, the resulting matrix $\widehat{\boldsymbol{\Sigma}}_{\mathbf{w}}$ is *singular* if the number of replications is "too small"; i.e., if $m \leq n$; see Dykstra (1970).

Note: Xu et al. (2014) discusses the estimation of a covariance matrix under a weighted quadratic loss function.

Kleijnen (1992) shows that we may compute CIs for the individual OLS estimators $\widehat{\beta}_j$ ($j = 1, \ldots, q$) from a t-statistic with $m-1$ degrees of freedom, provided $m > n$. In this t-statistic, the standard errors $s(\widehat{\beta}_j)$ are the square roots of the elements on the main diagonal of the estimated covariance matrix using Eqs. (3.16) and (3.31).

An *alternative* method does not require $\widehat{\boldsymbol{\Sigma}}_{\mathbf{w}}$, so it suffices that $m > 1$ (this method can also be found in Law (2015) and Schruben (2010)). This alternative does require that we compute the OLS estimate m times; i.e., using replication r, we estimate $\boldsymbol{\beta}$ through

$$\widehat{\boldsymbol{\beta}}_r = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w}_r \quad (r = 1, \ldots, m). \tag{3.32}$$

Obviously, the n elements of $\mathbf{w}_r$ are correlated because of CRN, and these elements may have different variances. The m estimators $\widehat{\beta}_{j;r}$ ($j = 1, \ldots, q$; $r = 1, \ldots, m$) of an individual regression parameter β_j are independent because these estimators use nonoverlapping PRN streams; these m estimators have a common standard deviation (say) $\sigma_{\widehat{\beta}_j}$. Therefore we replace Eq. (2.19) by

$$t_{m-1} = \frac{\overline{\widehat{\beta}}_j - \beta_j}{s(\overline{\widehat{\beta}}_j)} \quad \text{with } j = 1, \ldots, q \tag{3.33}$$

where the numerator includes $\overline{\widehat{\beta}}_j = \sum_{r=1}^{m} \widehat{\beta}_{j;r}/m$ and the denominator is given by

$$s(\overline{\widehat{\beta}}_j) = \sqrt{\frac{\sum_{r=1}^{m} (\widehat{\beta}_{j;r} - \overline{\widehat{\beta}}_j)^2}{m(m-1)}}.$$

However, we cannot apply this alternative method when we wish to estimate a quantile instead of an expected value. In case of an expected value, each replication gives an unbiased estimator of that value; in case of a quantile, many replications are needed to estimate this quantile. In case of a quantile, we recommend distribution-free bootstrapping; see Exercise 3.10 below, and also the example in Kleijnen et al. (2011).

Sub 2: We may decide to switch to GLS, because CRN imply that the BLUE is not the OLS but the GLS estimator, which is analogous to Eq. (3.19) with a nondiagonal $\mathbf{\Sigma_w}$. The covariance matrix of the GLS estimator is analogous to Eq. (3.21). In practice, $\mathbf{\Sigma_w}$ is unknown so we must estimate it. This estimator $\widehat{\mathbf{\Sigma}}_\mathbf{w}$ has the elements given by Eq. (3.31). This matrix is *singular* if the number of replications is "too small"; i.e., if $m \leq n$.

Substituting $\widehat{\mathbf{\Sigma}}_\mathbf{w}$ into the classic GLS estimation formula, we obtain *estimated GLS* (EGLS) which is analogous to EWLS in Eq. (3.22). The EGLS estimator can again be analyzed through jackknifing and bootstrapping. Kleijnen (1992), however, compares OLS and EGLS relying on the asymptotic covariance matrix of the EGLS estimator in Eq. (3.23) with a nondiagonal $\widehat{\mathbf{\Sigma}}_\mathbf{w}$. However, Davidson and MacKinnon (2007) states: "bootstrap tests ... yield more reliable inferences than asymptotic tests in a great many cases."

In conclusion, CRN with EGLS may give better point estimates of the input effects than CRN with OLS, but the EGLS estimate requires "many" replications—namely $m > n$—to obtain a nonsingular $\widehat{\mathbf{\Sigma}}_\mathbf{w}$.

Exercise 3.9 *Analogously to Exercise 3.7, simulate the M/M/1 model, as follows. Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time. Select an experimental area; e.g., the traffic load is 0.3 and 0.5. Each run should be "sufficiently long". Apply CRN for the different traffic rates. Use m replicated simulation runs; vary the number of replications between its minimum 2 and (say) 10. Fit a first-order polynomial. Now estimate the parameters of the metamodel, including a CI for the predicted output at a traffic rate of 0.4; does this CI cover the analytical solution?*

The literature pays no attention to the derivation of alternative designs for situations with CRN. Combining common and antithetic random numbers in classic designs is discussed in Schruben and Margolin (1978), and is extended in Chih (2013) and Song and Chiu (2007).

Equation (2.30) gave a lack-of-fit F-test assuming white noise. If we apply CRN, then we may apply the following variant of this test that is derived in Rao (1959):

$$F_{n-q;m-n+q} = \frac{m-n+q}{(n-q)(m-1)}(\overline{\mathbf{w}} - \widehat{\mathbf{y}})'\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}}}^{-1}(\overline{\mathbf{w}} - \widehat{\mathbf{y}}) \tag{3.34}$$

where $\hat{\mathbf{y}}$ denotes the EGLS estimator; obviously, $n > q$ and $m > n$.

Note: If the number of replications tends to infinity, then both the classic test and Rao's test converge in distribution to $\chi^2_{n-q}/(n-q)$ if the metamodel is valid.

Equation (3.34) also applies to EWLS instead of EGLS. Normality of the output $\mathbf{w}$ is an important assumption for both the classic lack-of-fit F-test and Rao's test; see Kleijnen (1992). In case of nonnormality, we may apply jackknifing or bootstrapping. We explain bootstrapping of EGLS as follows.

Suppose we apply CRN when simulating the n input combinations $\mathbf{x}_i$ $(i = 1, \ldots, n)$, and we obtain a fixed number m of replications for each $\mathbf{x}_i$. Obviously, the n elements w_i of the vector $\mathbf{w} = (w_1, \ldots, w_n)'$ are not IID; actually, we expect w_i and $w_{i'}$ $(i,\ i' = 1, \ldots, n)$ to be positively correlated. However, the m observations on this vector (say) $\mathbf{w}_r$ $(r = 1, \ldots, m)$ are IID, because they are transformations of nonoverlapping PRN streams. In distribution-free bootstrapping we resample—with replacement—the m multivariate outputs $\mathbf{w}_r$. This bootstrapping gives the bootstrapped averages $\overline{w}_i^* = \sum_{r=1}^{m} w_{i;r}^*/m$, which give the n-dimensional vector $\overline{\mathbf{w}}^* = (\overline{w}_1^*, \ldots, \overline{w}_n^*)'$. We can also compute the estimated covariance matrix of the bootstrapped averages $\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}}^*}$ from Eq. (3.31) with $w_{i;r}$ replaced by $w_{i;r}^*$ and $\overline{w}_i$ replaced by $\overline{w}_i^*$. Using the original $n \times q$ matrix $\mathbf{X}$ and the bootstrapped outputs $\overline{\mathbf{w}}^*$ and covariance matrix $\widehat{\boldsymbol{\Sigma}}_{\overline{\mathbf{w}}^*}$, we compute the bootstrapped EGLS $\widehat{\widehat{\boldsymbol{\beta}}}^*$. Repeating this bootstrapping B times gives $\widehat{\widehat{\boldsymbol{\beta}}}_b^*$ $(b = 1, \ldots, B)$, which enables us to compute the EDF for the regression parameters $\boldsymbol{\beta}$, and the EDF of Rao's statistic defined by Eq. (3.34). Kleijnen and Deflandre (2006) indeed bootstraps Rao's statistic (and also the classic R^2 statistic, which will be discussed in Sect. 3.6.1), under the null-hypothesis that the fitted metamodel is valid (bootstrapping under a null-hypothesis was discussed in Sect. 3.3.5).

Exercise 3.10 *Simulate the M/M/1 model, as follows. Pick a single (scalar) performance measure; e.g., the steady-state mean waiting time. Simulate* $n = 3$ *traffic rates. Apply CRN for these traffic rates. Fit a first-order polynomial, so* $q = 2$. *Use* $m = 25$ *replications; each run should be* *"sufficiently long". Use distribution-free bootstrapping to compute the EDF of the EGLS estimator of the regression parameters of the metamodel, and the EDF of Rao's lack-of-fit statistic.*

Note: CRN combined with Eq. (3.30) (Law (2015)'s formula for the desired number of replications $\widehat{m}$ when estimating the mean μ_w with relative error γ) gives unequal replication numbers, and is examined in Kleijnen et al. (2010)—albeit for Kriging instead of linear regression metamodels.

3.6 Validation of Metamodels

In this section we discuss the following questions (again, these questions were formulated more generally in Sect. 3.1):

1. How can we *test* the validity of the fitted linear regression metamodel?
2. If we find that this metamodel is not valid, can we then *transform* the simulation's I/O data such that a linear regression model becomes valid?
3. Which *alternative* metamodels can we apply, if the original metamodel turns out to be inadequate?

In practice, we do not know which type of metamodel gives a valid approximation—given the goals of the underlying simulation model; these goals were discussed in Sect. 1.2. For example—given a "small" experimental area—is the estimated first-order polynomial adequate to estimate the gradient (used to search for the optimum; see Chap. 6)? In Sect. 2.2 we have already discussed the classic lack-of-fit F-test assuming white noise, and in Eq. (3.34) we discussed Rao's variant. Now we present the following alternatives:

- two related coefficients of determination; namely, R^2 and R^2_{adj}
- cross-validation.

In practice, these alternatives have been applied to deterministic and random simulations, and to other metamodels than linear regression models; e.g., Kriging models. These alternatives may also be used to compare competing metamodels; e.g., a first-order polynomial versus a second-order polynomial, or a linear regression model versus a Kriging model. Validation of metamodels is also discussed in Bischl et al. (2012) and Santos and Santos (2011) and on. Pérez-Cruzado et al. (2015) also discusses several statistics for the validation of metamodels.

http://cran.r-project.org/web/packages/DiceEval/index.html
and
http://www.modelselection.org/.

3.6.1 The Coefficients of Determination R^2 and R^2_{adj}

R^2 is a very popular statistic in passive observation of real systems and in simulation models that are either deterministic or random. Whether or not replications are available, R^2 may be defined as follows (also see, e.g., Draper and Smith 1981, p. 33):

$$R^2 = \frac{\sum_{i=1}^{n} (\widehat{y}_i - \overline{\overline{w}})^2}{\sum_{i=1}^{n} (\overline{w}_i - \overline{\overline{w}})^2} = 1 - \frac{\sum_{i=1}^{n} (\widehat{y}_i - \overline{w}_i)^2}{\sum_{i=1}^{n} (\overline{w}_i - \overline{\overline{w}})^2} \tag{3.35}$$

where $\widehat{y}_i$ denotes the metamodel's predictor defined in Eq. (2.12), $\overline{w}_i$ denotes the simulation output of input combination i averaged over its $m_i \geq 1$ replications defined in Eq. (2.28), and $\overline{\overline{w}} = \sum_{i=1}^{n} \overline{w_i}/n$ denotes the "overall" average simulation output. The right-most equality in Eq. (3.35) shows that $R^2 = 1$ if $\widehat{y}_i = \overline{w}_i$ for all n values of i. The statistic R^2 measures how much of the variation in the simulation output is explained by the metamodel; see the denominator in Eq. (3.35), which is the numerator of the classic variance estimator computed over the n combinations—analogous to Eq. (2.27). We may also use R^2 in *deterministic* simulation, where we do not obtain any replications so in Eq. (3.35) $\overline{w}_i$ becomes w_i and $\overline{\overline{w}}$ becomes $\overline{w}$.

Renaud and Victoria-Feser (2010) points out that R^2 also equals the square of ρ—which is the classic symbol for *Pearson's correlation coefficient*—between w_i and $\widehat{y}_i$:

$$R^2 = \widehat{\rho}^2_{w;\widehat{y}} = \left(\frac{\widehat{\sigma}_{w;\widehat{y}}}{\widehat{\sigma}_w \widehat{\sigma}_{\widehat{y}}} \right)^2 = \left(\frac{\sum_{i=1}^{n} (w_i - \overline{w})(\widehat{y}_i - \overline{y})}{\sqrt{\sum_{i=1}^{n} (w_i - \overline{w})^2} \sqrt{\sum_{i=1}^{n} (\widehat{y}_i - \overline{\widehat{y}})^2}} \right)^2 \tag{3.36}$$

where we may replace $\widehat{\sigma}$ by s—as Eq. (2.27) demonstrates—and we use the definition $\overline{\widehat{y}} = \sum_{i=1}^{n} \widehat{y}_i/n$. In general, ρ quantifies the strength of the *linear* relationship between two random variables; e.g., w and $\widehat{y}$ in Eq. (3.36). Like R^2, the statistic ρ^2 ranges between 0 and 1. If $\widehat{\rho} = 1$, then w and $\widehat{y}$ are perfectly related by an increasing linear relationship; i.e., in the scatterplot all n pairs $(w_i, \widehat{y}_i)$ lie on a straight line with intercept 0 and slope 1 (45 °). Details on ρ are given in Kleijnen (2008, pp. 55–57) and Sun and Wong (2007); details on the numerical calculation and statistical tests are given in Press et al. (2007, pp. 745–748).

When ρ is computed from the ranked data or order statistics $(w_{(i)}, \widehat{y}_{(i)})$, the result is known as *Spearman's correlation coefficient*. Transformations such as this ranking will also be discussed in Sect. 3.6.3.

We do not define R^2 as a function of the *individual* outputs $w_{i;r}$, because we accept the metamodel as valid if it adequately predicts the *expected* output of the simulation model. Defining R^2 as a function of the individual outputs would give a lower R^2, because of the variability of the individual outputs per combination.

If $n = q$ (the design is saturated, so no degrees of freedom are left; see Chap. 2), then $R^2 = 1$—even if $E(e) \neq 0$. If $n > q$ and q increases, then R^2 increases—whatever the size of $|E(e)|$ is. Because of this problem of *overfitting*, the regression literature *adjusts* R^2 for the number of explanatory variables, as follows:

$$R^2_{\text{adj}} = 1 - \frac{n-1}{n-q}(1 - R^2). \tag{3.37}$$

So, if $q = 1$, then $R^2_{\text{adj}} = R^2$; if $q > 1$, then $R^2_{\text{adj}} < R^2$.

Lower *critical* values for either R^2 or R^2_{adj} are unknown, because these statistics do not have well-known distributions. We might use subjective lower thresholds. However, Kleijnen and Deflandre (2006) demonstrates how to estimate the distributions of these two statistics through *distribution-free bootstrapping* of the replicated outputs (bootstrapping was discussed in Sect. 3.3.5).

Note: Wang (2013) discusses a variant of R^2 that measures the ability of predicting a newly observed sample by using the fitted model. We do not further discuss this variant because we prefer cross-validation, which considers both the predictive use and the explanatory use of metamodels—as we shall see in Sect. 3.6.2. Furthermore, R^2 and R^2_{adj} are sensitive to outliers, which we briefly discussed in Sect. 3.3.3. Renaud and Victoria-Feser (2010) therefore discusses several robust variants of R^2, assuming a constant output variance σ^2_w. We do not further discuss these variants, because we prefer cross-validation over R^2.

3.6.2 Cross-Validation

Before we discuss cross-validation, we discuss the following algorithm that is often used for the validation of the predictive adequacy of any model, in any scientific discipline.

Algorithm 3.1

1. Use the model to compute a prediction $\widehat{y}$.
   Comment: This $\widehat{y}$ may be the outcome of the metamodel in DASE; in other areas, $\widehat{y}$ may be the outcome of a simulation model or some other model.

2. Observe the actual outcome w.
   Comment: This w is the simulation outcome in DASE; in other areas, w is the outcome of the real system.

3. Compare the two outcomes.
   Comment: This comparison checks whether these outcomes are close. This comparison may proceed as follows.

First we discuss this comparison of $\widehat{y}$ and w, in deterministic simulation, Next we discuss this comparison in random simulation, which is more complicated because of the random nature of w.

For *deterministic* simulation we assume that we compute $\widehat{y}_{n+1}$, the prediction for the "new" input combination $\mathbf{x}_{n+1}$ through a linear regression metamodel with parameters $\boldsymbol{\beta}$ estimated from the n "old" simulation outputs w_i of the combinations $\mathbf{x}_i$ $(i = 1, \ldots, n)$. Next we simply compute either the relative prediction error $\widehat{y}_{n+1}/w_{n+1}$ provided $w_{n+1} \neq 0$ or the absolute prediction error $|\widehat{y}_{n+1} - w_{n+1}|$. The relative error $\widehat{y}_{n+1}/w_{n+1}$ is scale-free; other validation measures are discussed below. Finally, we "eyeball" this prediction error, and decide whether the metamodel is acceptable for the goal of the simulation study; various goals are discussed in Sect. 1.2.

For *random* simulation we assume that we compute the prediction through a linear regression metamodel with parameters $\boldsymbol{\beta}$ estimated from the simulation output $w_{i;r}$ of input combination $\mathbf{x}_i$ where combination i is replicated m_i times $(i = 1, \ldots, n$ and $r = 1, \ldots, m_i)$. We use this metamodel to predict the actual simulation output for the new combination $\mathbf{x}_{n+1}$:

$$\widehat{y}_{n+1} = \mathbf{x}'_{n+1}\widehat{\boldsymbol{\beta}} \tag{3.38}$$

where for simplicity we assume that we use the OLS estimator $\widehat{\boldsymbol{\beta}}$ (a more complicated estimator would use EGLS; see Sect. 3.5). To estimate the expected simulation output for the same combination $\mathbf{x}_{n+1}$, we obtain m_{n+1} >1 replications and compute the average simulation output

$$\overline{w}_{n+1} = \frac{\sum_{r=1}^{m_{n+1}} w_{n+1;r}}{m_{n+1}}. \tag{3.39}$$

To compare the outcomes of Eqs. (3.38) and (3.39), we may use the scale-free Studentized statistic

$$t^{(i)}_{m-1} = \frac{\overline{w}_i - \widehat{y}_i}{\sqrt{s^2(\overline{w}_i) + s^2(\widehat{y}_{-i})}} \ (i = 1, ..., n) \tag{3.40}$$

where

$$s^2(\overline{w}_{n+1}) = \frac{\sum_{r=1}^{m_{n+1}}(w_{n+1;r} - \overline{w}_{n+1})^2}{m_{n+1}(m_{n+1} - 1)}$$

is the classic variance estimator, and

$$s^2(\widehat{y}_{n+1}) = \mathbf{x}'_{n+1}\widehat{\boldsymbol{\Sigma}}_{\widehat{\beta}}\mathbf{x}_{n+1} \tag{3.41}$$

follows from Eq. (2.17); the correct value for ν (degrees of freedom) in Eq. (3.40) is not so easy to determine, because $\overline{w}_{n+1}$ and $\widehat{y}_{n+1}$ have different variances: so-called *Behrens-Fisher problem.* A simple solution of this problem is

$$\nu = minm_{i'} - 1.$$

If the statistic in Eq. (3.40) is not significant, then we may accept the metamodel as being valid. Next, we may add the "new" simulation outputs $w_{n+1;r}$ $(r = 1, \ldots, m_{n+1})$ to the I/O data set. and *re-estimate* the regression parameters $\boldsymbol{\beta}$. We expect that the resulting new estimate does not deviate much from the old estimate—assuming the metamodel is indeed valid.

Actually, once we have added the new I/O data to the old data set, we may let the new and the old data change roles; e.g., we may replace $\mathbf{x}_1$ by $\mathbf{x}_{n+1}$ in the preceding equations. This idea leads to cross-validation.

Cross-validation is applied not only in linear regression analysis, but also in nonlinear regression analysis, Kriging, etc. The basic idea of cross-validation is quite old; see Stone (1974). Here we present so-called leave-one-out cross-validation, though Simpson et al. (2004) claims that more general "leave-k-out cross-validation" or "k-fold cross-validation" may be better; here k is the usual symbol for the subset of size k selected from the complete set of n I/O data $(\mathbf{x}_i, \mathbf{w}_i)$ with $i = 1, \ldots, n$ and $\mathbf{w}_i = (w_{i;1}, \ldots, w_{i;m_i})'$ (in the rest of this book k denotes the number of inputs). Software for cross-validation is available on

http://cran.r-project.org/web/packages/DiceEval/index.html.

For ease of presentation, we first assume that $\mathbf{X}$ has only n instead of $N = \sum_{i=1}^{n} m_i$ rows; i.e., we assume that the number of replications is constant, possibly one: $m_i = m \geq 1$. If m_i is a constant m higher than one $(m > 1)$, then we may replace the OLS estimate using $w_{i;r}$ (individual output for combination i) by $\overline{w}_i$ (average output for combination i), as we have already mentioned several times.

Note: If $m_i > 1$ and $m_i \neq m$ (different replication numbers), then the white noise assumption implies that the variance of the average output is σ_w^2/m_i; i.e., this variance is not constant. In case of such variance heterogeneity we should correct the OLS formulas; see again Sect. 3.4.

We present the following algorithm for *leave-one-out cross-validation*, which has five steps.

Algorithm 3.2

1. Delete I/O combination i from the complete set of n combinations, to obtain the remaining I/O data set $(\mathbf{X}_{-i}, \overline{\mathbf{w}}_{-i})$.
   Comment: We assume that this step results in a noncollinear matrix $\mathbf{X}_{-i}$ $(i = 1, \ldots, n)$; see Eq. (3.42) below. To satisfy this assumption, the original matrix $\mathbf{X}$ must satisfy the condition $n > q$. Counterexamples are saturated designs. A simple solution in case of a saturated design is to simulate one more combination, e.g., the center point if the original design is not a central composite design (CCD).

2. Use the data set resulting from step 1 to recompute the OLS estimator of the regression parameters:

$$\widehat{\boldsymbol{\beta}}_{-i} = (\mathbf{X}'_{-i}\mathbf{X}_{-i})^{-1}\mathbf{X}'_{-i}\overline{\mathbf{w}}_{-i}. \tag{3.42}$$

3. Use the estimator $\widehat{\boldsymbol{\beta}}_{-i}$ of step 2 to compute the regression prediction for the combination deleted in step 1:

$$\widehat{y}_{-i} = \mathbf{x}_i' \widehat{\boldsymbol{\beta}}_{-i}. \tag{3.43}$$

4. Repeat the preceding three steps until all n combinations have been processed, giving the n predictions $\widehat{y}_{-i}$.
   Comment: Obviously, these steps can also be executed simultaneously, so cross-validation suits parallel computers.

5. "Eyeball" a *scatterplot* with the n pairs $(\overline{w}_i, \widehat{y}_{-i})$, and decide whether the metamodel is valid.

Note: It would be wrong to proceed as follows, unless $m_i = 1$ (which is the case in deterministic simulation). Start with the $N \times q$ matrix $\mathbf{X}_N$ (instead of the $n \times q$ matrix $\mathbf{X}$), and the corresponding N-dimensional vector of outputs $\mathbf{w}$ (instead of $\overline{\mathbf{w}}$). Next, delete one row of this $\mathbf{X}_N$ and the corresponding $\mathbf{w}$ so $\mathbf{X}_N$ becomes $\mathbf{X}_{N-1}$. From the remaining I/O data, recompute the OLS estimator $\hat{\boldsymbol{\beta}}$ and the regression predictor $\widehat{y}$. We emphasize that this predictor uses $m_i - 1$ simulation outputs for combination i, so it does not challenge the metamodel to correctly predict the mean simulation output for this combination.

To illustrate this cross-validation, we return to Example 2.4.

Example 3.5 *Kleijnen and Standridge (1988) studies a deterministic simulation model of a flexible manufacturing system (FMS). A 2^{4-1} design is used, so n (number of combinations) is eight. This design gives I/O data, to which a first-order polynomial is fitted using OLS. Cross-validation suggests that the first-order polynomial is not adequate; Table 3.2 displays the relative prediction errors in percentages (e.g., deleting input combination 1 results in a prediction error of 10 %). Furthermore, cross-validation suggests that the effects of inputs z_1 and z_3 are negligible (not displayed in Table 3.2). So next, a first-order polynomial is fitted for the remaining two inputs z_2 and z_4 and their interaction. This metamodel is fitted to the "old" I/O data resulting from the 2^{4-1} design. Cross-validation of this new metamodel suggests that the resulting metamodel is valid; see Table 3.3. Furthermore, the estimated first-order effects of z_2 and z_4 and their interaction are found not to be very sensitive to the deletion of a combination; we do not display these data, but refer to the various tables in Kleijnen and Standridge (1988)*

.

Deleted combination i	1	2	3	4	5	6	7	8
Percentage prediction error	10	27	−19	−18	13	33	−38	−35

TABLE 3.2. Cross-validation of FMS example: relative predition error

Deleted combination i	1	2	3	4	5	6	7	8
Percentage prediction error	2	2	−1	1	−16	14	0	0

TABLE 3.3. Cross-validation of FMS example: relative predition error for new metamodel

Illustrations of cross-validation are also provided in the following two case studies:

- a deterministic spreadsheet simulation for the economic appraisal of natural gas projects; see Van Groenendaal (1998)
- a random simulation for the control of animal diseases; see Vonk Noordegraaf (2002).

Instead of the subjective judgment in step 5, Kleijnen (1983) proposes the following alternative that is inspired by Eq. (3.40). Compute

$$t_{m-1}^{(i)} = \frac{\overline{w}_i - \widehat{y}_i}{s(\overline{w}_i) + s(\widehat{y}_{-i})} \quad (i = 1, \ldots, n) \tag{3.44}$$

where $s(\overline{w}_i) = s(w_i)/\sqrt{m}$ and $s(w_i)$ follows from Eq. (2.27), and $s(\widehat{y}_{-i})$ follows from Eq. (3.43) and the analogue of Eq. (2.17) so

$$s(\widehat{y}_{-i}) = \sqrt{\mathbf{x}_i' \widehat{\boldsymbol{\Sigma}}_{\widehat{\boldsymbol{\beta}}_{-i}} \mathbf{x}_i} \tag{3.45}$$

where

$$\widehat{\boldsymbol{\Sigma}}_{\widehat{\boldsymbol{\beta}}_{-i}} = s^2(\overline{w}_i)(\mathbf{X}_{-i}'\mathbf{X}_{-i})^{-1}. \tag{3.46}$$

Because i runs from 1 through n, Eq. (3.44) gives n values. We reject the regression metamodel if

$$max_i \; |t_{m-1}^{(i)}| > t_{m-1;1-[\alpha/(2n)]} \tag{3.47}$$

where the right-hand side follows from *Bonferroni's inequality.* This inequality implies that the classic type-I error rate (in this case $\alpha/2$) is replaced by the same value divided by the number of tests (in this case n)—resulting in the "experimentwise" or "familywise" type-I error rate α. Obviously, the n statistics $t_{m-1}^{(i)}$ are highly correlated because they have many outputs in common. We also refer back to our discussion of Bonferroni's inequality below Eq. (3.14).

Note: We may replace the OLS estimator $\widehat{\boldsymbol{\beta}}$ by the EWLS or EGLS estimator $\widehat{\widehat{\boldsymbol{\beta}}}$. The t-statistic is less sensitive to nonnormality than the F-statistic; see the extensive Monte Carlo study in Kleijnen (1992).

There is a *shortcut* for the n computations in this cross-validation procedure; this shortcut is used in regression software, and is also discussed in Miller (1990, pp. 201–202). The shortcut uses the *hat matrix* $\mathbf{H}$, which we have already defined below Eq. (3.4) and we again define here for convenience:

$$\mathbf{H} = (\mathbf{h}_{i;i'}) = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \quad \text{with } i, i' = 1, \ldots, n. \tag{3.48}$$

This $\mathbf{H}$ is implicitly used in Eq. (2.12) where $\widehat{y}_i = \mathbf{x}'_i\widehat{\boldsymbol{\beta}}$, because this equation implies the vector $\widehat{\mathbf{y}} = (\widehat{y}_i) = \mathbf{X}\widehat{\boldsymbol{\beta}}$, which together with Eq. (2.13) gives

$$\widehat{\mathbf{y}} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\overline{\mathbf{w}} = \mathbf{H}\overline{\mathbf{w}}. \tag{3.49}$$

Geometrically speaking, $\mathbf{H}$ projects the vector of observations $\overline{\mathbf{w}}$ onto the subspace spanned by $\mathbf{X}$. Such a *projection matrix* is idempotent: $\mathbf{HH} = \mathbf{H}$. Obviously, $\mathbf{H}$ is an $n \times n$ matrix, so it assumes that the number of replications is constant, possibly one.

Note: If $m_i > 1$ and $m_i \neq m$ (different replication numbers), then the white noise assumption implies that the variance of the output average $\overline{w}_i$ is σ_w^2/m_i so this variance is not constant. Then a more complicated definition of the hat matrix becomes necessary for the shortcut; see Kleijnen and Van Groenendaal (1992, p. 157).

Equation (3.48) implies that element i on the main diagonal of $\mathbf{H}$ is h_{ii}. Atkinson and Riani (2000, pp. 18, 24) proves that the numerator of Eq. (3.44) may be written as

$$\overline{w}_i - \widehat{y}_{-i} = \frac{\overline{w}_i - \widehat{y}_i}{1 - h_{i;i}}$$

and Eq. (3.44) itself may be written as

$$t_{m_i - 1} = \frac{\overline{w}_i - \widehat{y}_i}{s(\overline{w}_i)\sqrt{1 - h_{i;i}}} \; (i = 1, \ldots, n) \tag{3.50}$$

so the cross-validation computations can be based solely on the *original* I/O data, $(\mathbf{X}, \mathbf{w})$, which give $\widehat{y}_i$ and $h_{i;i}$ (the subscripts involve i, not $-i$).

Below Eq. (2.21) we have already pointed out the difference between *significance* and *importance (*an input may be significant but not important, and vice versa). In situations with many simulation replications, a metamodel may give a predicted value that differs significantly from the simulation output, and yet the metamodel may adequately serve its purpose. For example, Breukers (2006) uses $m = 500$ replications when comparing the outcomes of a first-order polynomial metamodel and the original simulation for a new input combination, using Eq. (3.40). This comparison gives a significant difference. Yet the metamodel adequately helps identify the important inputs, even though the metamodel is not perfect.

Note: In deterministic simulation, we should not apply Eq. (3.47)—for the following reasons. Deterministic simulation implies that $s(\overline{w}_i) = 0$ in Eq. (3.44). We might compute $s(\widehat{y}_{-i})$ in Eq. (3.45), using Eq. (3.46) with $s^2(\overline{w}_i)$ now computed from the mean squared residuals (MSR) defined in Eq. (2.20). But the worse the metamodel fits, the bigger this MSR gets—so the smaller the test statistic in Eq. (3.44) becomes so the smaller the probability of rejecting this false metamodel becomes. Therefore we proposed to compute the relative prediction errors $\widehat{y}_{-i}/w_i$, and decide whether these errors are acceptable—practically speaking; see again Example 3.5. In other words, instead of Studentizing the prediction errors, we now standardize the prediction errors by using relative errors. An alternative remains the scatterplot described in Step 5 of the cross-validation procedure above.

Cross-validation affects not only the regression predictions $\widehat{y}_{-i} (i = 1, \ldots, n)$, but also the estimated q-dimensional vector with regression parameters $\widehat{\boldsymbol{\beta}}_{-i}$; see Eq. (3.42). So we may be interested not only in the predictive performance of the metamodel, but also in its *explanatory* performance. In Example 3.5 we mentioned that cross-validation gave stable first-order effects for z_2 and z_4 and for the interaction between these two inputs.

Related to Eq. (3.50) are several so-called *diagnostic* statistics that are proposed in the regression literature. Examples are DEFITS, DFBETAS, and Cook's D—also see Kleijnen and Van Groenendaal (1992, p. 157)—but the most popular diagnostic statistic is the *prediction sum of squares* (PRESS):

$$\text{PRESS} = \sqrt{\frac{\sum_{i=1}^{n} (\widehat{y}_{-i} - w_i)^2}{n}}$$

where we assume leave-one-out cross-validation. Viana et al. (2014) gives examples of PRESS in deterministic simulation, for various types of metamodel; e.g., Kriging, but not linear regression.

The simulation literature proposes validation measures that are related to the mean squared error (MSE); e.g., the *root MSE* (RMSE), the *average absolute error* (AAE) or *mean absolute error* (MAE), and the *average absolute relative error* (AARE). In the statistics literature, AAE is also known as the *mean absolute deviation* (MAD); also see Gorissen (2010, chapter 8) and Tofallis (2015). Instead of taking the mean (see the letter M in the preceding acronyms) or average (see the A in these acronyms), we may take the maximum; e.g., we may compute the maximum absolute error. The mean is relevant for risk-neutral users, whereas the maximum is for risk-averse users. For further discussion, we refer to Hamad (2011), Kleijnen and Sargent (2000), Lin et al. (2002), and

`http://cran.r-project.org/web/packages/DiceEval/index.html`.

We may apply bootstrapping to estimate the distribution of these validation statistics; for details we refer to Bischl et al. (2012), Cheng (2006a), Efron and Tibshirani (1993, pp. 247–255), and Harrell et al. (1996).

Exercise 3.11 *Simulate the M/M/1 model (also see Exercise 1.6). Pick a single (scalar) performance measure; e.g., the mean waiting time in the steady-state, or the mean waiting time of the first 100 or 1,000 customers. Select two different experimental areas; e.g., the traffic load* $\rho = \lambda/\mu$ *varies between 0.1 and 0.3 and between 0.5 and 0.8. Select these two areas such that a first-order polynomial seems to give good and bad fit, respectively; for "high" traffic rates, the first-order polynomial is not a valid metamodel. To select these areas, "cheat" as follows: draw a plot of the analytical steady-state mean against the traffic rate. Use* m_i *replications. Either ignore the variance heterogeneity within the experimental area or use more replications for the higher traffic rate; see Eq. (3.28). Use either a single PRN stream or two streams for arrival and service times. To simplify the analysis, do not apply CRN for different traffic rates. Now validate the fitted metamodel, using different techniques; e.g.,* R^2 *and cross-validation.*

Note: Besides quantitative tests, we may use graphical methods to judge the validity of a fitted metamodel. We have already discussed scatterplots in Step 5 of the cross-validation procedure above. Hamad (2011) mentions several other plots for judging the validity of metamodels. Viana et al. (2014) also emphasizes the importance of visualization. The following software may give various plots; namely, the residuals $\widehat{e}_i$ versus the index i, the residuals against the fitted values $\widehat{y}_i$, the EDF of e, and the so-called *normal Q-Q plot*:

`http://cran.r-project.org/web/packages/DiceEval/DiceEval.pdf`.

3.6.3 Transformations of Regression Variables

If the validation tests suggest important approximation errors in the fitted metamodel, then we may consider the following alternatives. In Eq. (2.9) we have already seen that a transformation combining two simulation inputs—namely, the arrival rate λ and the service rate μ—into a single independent regression variable—namely, the traffic rate $x = \lambda/\mu$—may give a better metamodel. In Eq. (2.7) we have seen another useful transformation,; namely, replace y, λ, and μ by $\log(y)$, $\log(\lambda)$, and $\log(\mu)$ so that the first-order polynomial approximates relative changes through elasticity coefficients.

Another simple transformation assumes that the I/O function of the underlying simulation model is *monotonic*. Then it makes sense to replace the dependent and independent variables by their ranks, which results in so-called *rank regression*; see Conover and Iman (1981) and also Saltelli et al. (2005), and Saltelli and Sobol (1995).

Note: Spearman's correlation coefficient also uses the rank transformation, but for only two correlated random variables. Kleijnen (2008, p. 57) and Kleijnen and Helton (1999) use Spearman's coefficient and rank regression to find the most important inputs in a random simulation model of nuclear waste disposal.

Note: Monotonic linear regression models are discussed in Tan (2015).

We may also apply transformations to make the simulation output (dependent regression variable) better satisfy the assumptions of normality and variance homogeneity; see again Sects. 3.3.3 and 3.4.3. Unfortunately, different goals of a transformation may conflict with each other; e.g., we may apply the logarithmic transformation to reduce nonnormality, but this transformation may give a metamodel with variables that are not of immediate interest.

3.6.4 Adding High-Order Terms

In the preceding chapter, we discussed designs for low-order polynomial metamodels. Resolution-III designs assume first-order polynomial metamodels; resolution-IV and resolution-V designs assume two-factor interactions; CCDs assume second-order polynomials. If these designs do not give valid metamodels, then we may look for *transformations,* as discussed in the preceding subsection. We do not recommend routinely adding higher-order terms to the metamodel, because these terms are hard to interpret. However, if the goal is not to better *explain* or *understand* the underlying simulation model but to better *predict* the output of an expensive simulation model, then we may add such high-order terms. Indeed, full factorial 2^k designs enable the estimation of all interactions, such as the interaction among all k inputs.

The regression literature calls the addition of more explanatory variables *stepwise regression.* That literature calls the elimination of nonsignificant variables *backwards elimination*; also see testing the significance of one or more variables in Eqs. (2.21) and (2.22).

Note: If we simulate more than two levels per input, then we may consider other types of metamodels; e.g., Kriging models (see Chap. 5). These alternatives may give better predictions than low-order polynomials do, but they are so complicated that they do not give much help when we try to better understand the underlying simulation model. Furthermore, these alternatives require alternative design types; e.g., Latin Hypercube Sampling (see Sect. 5.5.1).

3.7 Conclusions

In this chapter we discussed the assumptions of classic linear regression analysis and the concomitant statistical designs when these methods are applied in simulation practice. We pointed out that–given specific assumptions–multiple simulation outputs may still be analyzed through OLS per output type. We addressed possible nonnormality of the simulation output, including normality tests, normalizing transformations of the simulation output, and the distribution-free methods called jackknifing and bootstrapping. If the simulation outputs do not have a common variance, then we may apply alternative analysis and design methods. We discussed how to analyze simulation I/O data that use CRN, which make the simulation outputs correlated across different input combinations, within the same replication. We discussed the validation of linear regression metamodels, and transformations to improve the metamodel's validity.

Solutions of Exercises

Solution 3.1 *The jackknife results for this M/M/1 simulation depend on the PRN stream; see Kleijnen and Deflandre (2003) for examples.*

Solution 3.2 *The jackknife results for this Monte Carlo experiment depend on the PRN stream; see Kleijnen and Van Groenendaal (1992, pp. 141–146) and also Kleijnen (1992) and Kleijnen et al. (1987) for examples.*

Solution 3.3 *The bootstrap results for this M/M/1 simulation depend on the PRN stream; see Kleijnen and Deflandre (2003) for an example.*

Solution 3.4 *The bootstrap results for this Monte Carlo experiment depend on the PRN stream; see Kleijnen and Deflandre (2003) for examples.*

Solution 3.5 *See Sect. 3.3.5 on bootstrapping.*

Solution 3.6 *If* $\mathbf{\Sigma_w} = \sigma_w^2\mathbf{I}$, *then* $\mathbf{\Sigma}_{\widehat{\boldsymbol{\beta}}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Sigma_w}\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} = \sigma_w^2(\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1} = \sigma_w^2(\mathbf{X}'\mathbf{X})^{-1}$.

Solution 3.7 *The results for this M/M/1 simulation depend on the specific PRNs, etc.*

Solution 3.8 *Let the intercept be estimated through*

$$\widehat{\beta}_1 = \sum_{i=1}^{n} w_i/n = \mathbf{1}'_n\mathbf{w}/n$$

with $\mathbf{1}'_n$ *a vector with* n *ones. Then*

$$Var(\widehat{\beta}_1) = \mathbf{1}'\mathbf{\Sigma_w}\mathbf{1}/n^2 = \sum_{i=1}^{n}\sum_{i'=1}^{n}\sigma_{i;i'}/n^2$$

where $\sigma_{i;i'}$ *denotes the covariance between* w_i *and* $w_{i'}$ *if* $i \neq i'$ *and* $\sigma_{i;i} = \sigma_i^2$*; so* $Var(\widehat{\beta}_1)$ *increases if CRN "works" so* $\sigma_{i;i'} > 0$.

Solution 3.9 *The results for this M/M/1 simulation depend on the specific PRNs, etc.*

Solution 3.10 *The results for this M/M/1 simulation with CRN depend on the specific PRNs, etc.*

Solution 3.11 *The results for this M/M/1 simulation depend on the specific PRNs, etc.*

References

Alba Fernández MV, Jiménez Gamero MD, Castillo Gutiérrez S (2014) Approximating a class of goodness-of-fit test statistics. Math Comput Simul 102:24–38

Arcones MA, Wang Y (2006) Some new tests for normality based on U-processes. Stat Probab Lett 76(1):69–82

Arnold SF (1981) The theory of linear models and multivariate analysis. Wiley, New York

Atkinson A, Riani M (2000) Robust diagnostic regression analysis. Springer, New York

Ayanso A, Diaby M, Nair SK (2006) Inventory rationing via drop-shipping in Internet retailing: a sensitivity analysis. Eur J Oper Res 171(1):135–152

Bekki JM, Fowler JW, Mackulak GT, Kulahci M (2009) Simulation-based cycle-time quantile estimation in manufacturing settings employing non-FIFO dispatching policies. J Simul 3(2):69–128

Billingsley P (1968) Convergence of probability measures. Wiley, New York

Bischl B, Mersmann O, Trautmann H, Weihs C (2012) Resampling methods for meta-model validation with recommendations for evolutionary computation. Evol Comput 20(2):249–275

Brantley MW, Lee LH, Chen C-H, Xu J (2014) An efficient simulation budget allocation method incorporating regression for partitioned domains. Automatica 50(5):1391–1400

Breukers A (2006) Bio-economic modelling of brown rot in the Dutch potato production chain. Doctoral dissertation, Wageningen University, Wageningen

Ceranka B, Graczyk M, Katulska K (2006) A-optimal chemical balance weighing design with nonhomogeneity of variances of errors. Stat Probab Lett 76(7):653–665

Chang C-H, Pal N, Lim WK, Lin J-J (2010) Comparing several population means: a parametric bootstrap method, and its comparison with usual ANOVA F test as well as ANOM. Comput Stat 25:71–95

Chen EJ (2008) Some procedures of selecting the best designs with respect to quantile. Simulation 84:275–284

Chen Y, Yu J (2015) Optimal jackknife for unit root models. Stat Probab Lett 99:135–142

Cheng RCH (2006a) Validating and comparing simulation models using resampling. J Simul 1:53–63

Cheng RCH (2006b) Resampling methods. In: Henderson SG, Nelson BL (eds) Handbooks in operations research and management science, vol 13. North-Holland, Amsterdam, pp 415–453

Cheng RCH, Kleijnen JPC (1999) Improved design of queueing simulation experiments with highly heteroscedastic responses. Oper Res 47(5):762–777

Cheng RCH, Kleijnen JPC, Melas VB (2000) Optimal design of experiments with simulation models of nearly saturated queues. J Stat Plan Inference 85(1–2):19–26

Chernick MR (2007) Bootstrap methods; a practitioner's guide, 2nd edn. Wiley, New York

Chih M (2013) A more accurate second-order polynomial metamodel using a pseudo-random number assignment strategy. J Oper Res Soc 64:198–207

Cho K, Loh W-Y (2006) Bias and convergence rate of the coverage probability of prediction intervals in Box-Cox transformed linear models. J Stat Plan Inference 136(10):3614–3624

Conover WJ (1999) Practical nonparametric statistics, 3rd edn. Wiley, New York

Conover WJ, Iman RL (1981) Rank transformations as a bridge between parametric and nonparametric statistics. Am Stat 35(3):124–133

Davidson R, MacKinnon JG (2007) Improving the reliability of bootstrap tests with the fast double bootstrap. Comput Stat Data Anal 51(7):3259–3281

Davison AC, Hinkley DV (1997) Bootstrap methods and their application. Cambridge University Press, Cambridge

Döhler S (2014) A sufficient criterion for control of some generalized error rates in multiple testing. Stat Probab Lett 92:114–120

Draper NR, Smith H (1981) Applied regression analysis, 2nd edn. Wiley, New York

Dykstra RL (1970) Establishing the positive definiteness of the sample covariance matrix. Ann Math Stat 41(6):2153–2154

Efron B (1982) The jackknife, the bootstrap and other resampling plans. CBMS-NSF series. SIAM, Philadelphia

Efron B (2011) The bootstrap and Markov chain Monte Carlo. J Biopharm Stat 21(6):1052–1062

Efron B, Tibshirani RJ (1993) An introduction to the bootstrap. Chapman & Hall, New York

Freeman J, Modarres R (2006) Inverse Box-Cox: the power-normal distribution. Stat Probab Lett 76(8):764–772

Gel YR, Miao W, Gastwirth JL (2007) Robust directed tests of normality against heavy-tailed alternatives. Comput Stat Data Anal 51:2734–2746

Ghosh S, Polansky AM (2014) Smoothed and iterated bootstrap confidence regions for parameter vectors. J Multivar Anal 132:171–182

Gilbert S, Zemčík P (2006) Who's afraid of reduced-rank parameterizations of multivariate models? Theory and example. J Multivar Anal 97(4):925–945

Godfrey LG (2006) Tests for regression models with heteroskedasticity of unknown form. Comput Stat Data Anal 50(10):2715–2733

Goldberg PW, Williams CKI, Bishop CM (1998) Regression with input-dependent noise: a Gaussian process treatment. In: Jordan MI, Kearns MJ, Solla SA (eds) Advances in neural information processing systems, vol 10. MIT, Cambridge pp. 493–499

Good PI (2005) Resampling methods: a practical guide to data analysis, 3rd edn. Birkhäuser, Boston

Gordon AY (2007) Unimprovability of the Bonferroni procedure in the class of general step-up multiple testing procedures. Stat Probab Lett 77(2):117–122

Gordy MB, Juneja S (2010) Nested simulation in portfolio risk measurement. Manag Sci 56(11):1833–1848

Gorissen D (2010) Grid-enabled adaptive surrogate modeling for computer aided engineering. Ph.D. dissertation Ghent University, Ghent

Hamad H (2011) Validation of metamodels in simulation: a new metric. Eng Comput 27(4):309–317

Harrell FE, Lee KL, Mark DB (1996) Tutorial in biostatistics; multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy and measuring and reducing errors. Stat Med 15:361–387

Hartley HO (1950) The maximum F-ratio as a short-cut test for heterogeneity of variance. Biometrika 50:187–194

Hartung J, Knapp G (2010) Adaptive confidence intervals of desired length and power for normal means. J Stat Plan Inference 140:3317–3325

Hoad K, Robinson S, Davies R (2010) Automated selection of the number of replications for a discrete-event simulation. J Oper Res Soc 61(11):1632–1644

Hong LJ (2009) Estimating quantile sensitivities. Oper Res 57(1):118–130

Horowitz JL (2001) The bootstrap. Volume 5 of handbook of econometrics. North Holland, Oxford

Hsieh K-L, Chen Y-K (2007) Bootstrap confidence interval estimates of the bullwhip effect. Simul Model Pract Theory 15(8):908–917

Irizarry MA, Kuhl ME, Lada EK, Subramanian S, Wilson JR (2003) Analyzing transformation-based simulation metamodels. IIE Trans 35(3):271–283

Ivanescu C, Bertrand W, Fransoo J, Kleijnen JPC (2006) Bootstrapping to solve the limited data problem in production control: an application in batch processing industries. J Oper Res Soc 57(1):2–9

Jimenez-Gamero MD, Kim H-M (2015) Fast goodness-of-fit tests based on the characteristic function. Computational Statistics & Data Analysis, 89:172–191

Jurečková J, Picek J (2007) Shapiro–Wilk-type test of normality under nuisance regression and scale. Comput Stat Data Anal 51(10):5184–5191

Khuri AI, Mukhopadhyay S (2010) Response surface methodology. Wiley Interdiscip Rev Comput Stat 2:128–149

Kim S, Alexopoulos C, Tsui K, Wilson JR (2007) Distribution-free tabular CUSUM chart for autocorrelated data. IIE Trans 39:317–330

Kleijnen JPC (1983) Cross-validation using the t statistic. Eur J Oper Res 13(2):133–141

Kleijnen JPC (1987) Statistical tools for simulation practitioners. Marcel Dekker, New York

Kleijnen JPC (1992) Regression metamodels for simulation with common random numbers: comparison of validation tests and confidence intervals. Manag Sci 38(8):1164–1185

Kleijnen JPC (1993) Simulation and optimization in production planning: a case study. Decis Support Syst 9:269–280

Kleijnen JPC (1995) Sensitivity analysis and optimization of system dynamics models: regression analysis and statistical design of experiments. Syst Dyn Rev 11(4):275–288

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Kleijnen JPC, Cheng RCH, Bettonvil B (2001) Validation of trace-driven simulation models: bootstrapped tests. Manag Sci 47(11):1533–1538

Kleijnen JPC, Cremers P, van Belle F (1985) The power of weighted and ordinary least squares with estimated unequal variances in experimental designs. Commun Stat Simul Comput 14(1):85–102

Kleijnen JPC, Deflandre D (2003) Statistical analysis of random simulations: bootstrap tutorial. Simul News Europe issue 38/39:29–34

Kleijnen JPC, Deflandre D (2006) Validation of regression metamodels in simulation: bootstrap approach. Eur J Oper Res 170(1):120–131

Kleijnen JPC, Helton JC (1999) Statistical analyses of scatter plots to identify important factors in large-scale simulations, 1: review and comparison of techniques. Reliab Eng Syst Saf 65(2):147–185

Kleijnen JPC, Karremans PCA, Oortwijn WK, van Groenendaal WJH (1987) Jackknifing estimated weighted least squares: JEWLS. Commun Stat Theory Methods 16(3):747–764

Kleijnen JPC, Kriens J, Timmermans H, Van den Wildenberg H (1989) Regression sampling in statistical auditing: a practical survey and evaluation (including rejoinder). Statistica Neerlandica 43(4):193–207 (p 225)

Kleijnen JPC, Pierreval H, Zhang J (2011) Methodology for determining the acceptability of system designs in uncertain environments. Eur J Oper Res 209(2):176–183

Kleijnen JPC, Sargent RG (2000) A methodology for the fitting and validation of metamodels in simulation. Eur J Oper Res 120(1):14–29

Kleijnen JPC, Standridge C (1988) Experimental design and regression analysis: an FMS case study. Eur J Oper Res 33(3):257–261

Kleijnen JPC, Van Beers WCM (2013) Monotonicity-preserving bootstrapped Kriging metamodels for expensive simulations. J Oper Res Soc 64(5):708–717

Kleijnen JPC, Van Beers WCM, Van Nieuwenhuyse I (2010) Constrained optimization in simulation: a novel approach. Eur J Oper Res 202(1):164–174

Kleijnen JPC, Van Groenendaal W (1992) Simulation: a statistical perspective. Wiley, Chichester

Kleijnen JPC, Van Groenendaal W (1995) Two-stage versus sequential sample-size determination in regression analysis of simulation experiments. Am J Math Manag Sci 15(1&2):83–114

Kleijnen JPC, Van Ham G, Rotmans J (1992) Techniques for sensitivity analysis of simulation models: a case study of the CO_2 greenhouse effect. Simulation 58(6):410–417

Kreiss J-P, Paparoditis E (2011) Bootstrap methods for dependent data: a review (with discussion). J Korean Stat Soc 40:357–395

Lahiri SN (2003) Resampling methods for dependent data. Springer, New York

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Lehmann EL (1999) Elements of large-sample theory. Springer, New York

Lin Y, Mistree F, Tsui K-L, Allen JK (2002) Metamodel validation with deterministic computer experiments. In: 9th AIAA/ISSMO symposium on multidisciplinary analysis and optimization, Atlanta, 4–6 Sept 2002, Paper Number AIAA-2002-5425

Lunneborg CE (2000) Data analysis by resampling: concepts and applications. Duxbury Press, Pacific Grove

Mammen E, Nandi S (2012) Bootstrap and resampling. In: Gentle JE, Haerdle W, Mori Y (eds) Handbook of computational statistics, vol 1. Concepts and fundamentals, 2nd edn. Springer, Heidelberg, pp 499–527

Mandel M, Betensky RA (2008) Simultaneous confidence intervals based on the percentile bootstrap approach. Comput Stat Data Anal 52:2158–2165

Markiewicz A, Szczepańska A (2007) Optimal designs in multivariate linear models. Stat Probab Lett 77:426–430

Martin MA (2007) Bootstrap hypothesis testing for some common statistical problems: a critical evaluation of size and power properties Comput Stat Data Anal 51(12):6321–6342

Martínez-Camblor P, Corral N (2012) A general bootstrap algorithm for hypothesis testing. J Stat Plan Inference 142(2):589–600

Miller AJ (1990) Subset selection in regression. Chapman and Hall, London

Miller RG (1974) The jackknife—a review. Biometrika 61:1–15

Montevechi JAB, de Almeida Filho RG, Paiva AP, Costa RFS, Medeiros AL (2010) Sensitivity analysis in discrete-event simulation using fractional factorial designs. J Simul 4(2):128–142

Narula SC, Wellington JF (2007) Multiple criteria linear regression. Eur J Oper Res 181(2):767–772

Noguera JH, Watson EF (2006) Response surface analysis of a multi-product batch processing facility using a simulation metamodel. Int J Prod Econ 102(2):333–343

Novikov I, Oberman B (2007) Optimization of large simulations using statistical software. Comput Stat Data Anal 51(5):2747–2752

Pasupathy R, Hunter SR, Pujowidianto NA, Lee LH, Chen C-H (2014) Stochastically constrained ranking and selection via SCORE. ACMTrans Model Comput Simul 25(1):1:1–1:26

Paparoditis E, Politis DN (2005) Bootstrap hypothesis testing in regression models. Stat Probab Lett 74:356–365

Park DS, Kim YB, Shin KI, Willemain TR (2001) Simulation output analysis using the threshold bootstrap. Eur J Oper Res 134:17–28

Pérez-Cruzado C, Fehrmann L, Magdon P, Cañellas I, Sixto H, Kleinn C (2015) On the site-level suitability of biomass models. Environ Model Softw 73:14–26

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (2007) Numerical recipes: the art of scientific computing, third edition. Cambridge University Press

Psaradakis Z (2006) Blockwise bootstrap testing for stationarity. Stat Probab Lett 76(6):562–570

Racine JS, MacKinnon JG (2007) Inference via kernel smoothing of bootstrap values *P* values. Comput Stat Data Anal 51(12):5949–5957

Rao CR (1959) Some problems involving linear hypothesis in multivariate analysis. Biometrika 46:49–58

Rao CR (1967) Least squares theory using an estimated dispersion matrix and its application to measurement of signals. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Berkeley, vol I, pp 355–372

Renaud O, Victoria-Feser M-P (2010) A robust coefficient of determination for regression. J Stat Plan Inference 140(7):1852–1862

Ruud PA (2000) An introduction to classical econometric theory. Oxford University Press, New York

Sadooghi-Alvandi SM, Malekzadeh A (2014) Simultaneous confidence intervals for ratios of means of several lognormal distributions: a parametric bootstrap approach. Comput Stat Data Anal 69:133–140

Saltelli A, Ratto M, Tarantola S, Campolongo F (2005) Sensitivity analysis of chemical models. Chem Rev 105(7):2811–2827

Saltelli A, Sobol IM (1995) About the use of rank transformation in sensitivity analysis of model output. Reliab Eng Syst Saf 50:225–239

Santos MI, Santos PM (2011) Construction and validation of distribution-based regression simulation metamodels. J Oper Res Soc 62:1376–1384

Scheffé H (1964) The analysis of variance, 4th printing. Wiley, New York

Schruben LW (2010) Simulation modeling for analysis. ACM Trans Model Comput Simul 20(1):2.1–2.22

Schruben LW, Margolin BH (1978) Pseudorandom number assignment in statistically designed simulation and distribution sampling experiments. J Am Stat Assoc 73(363):504–525

Shao J, Tu D (1995) The jackknife and bootstrap. Springer, New York

Simpson TW, Booker AJ, Ghosh D, Giunta AA, Koch PN, Yang R-J (2004) Approximation methods in multidisciplinary analysis and optimization: a panel discussion. Struct Multidiscip Optim 27(5):302–313

Song WT, Chiu W (2007) A five-class variance swapping rule for simulation experiments: a correlated-blocks design. IIE Trans 39:713–722

Spöck G, Pilz J (2015) Incorporating covariance estimation uncertainty in spatial sampling design for prediction with trans-Gaussian random fields. Front Environ Sci 3(39):1–22

Stone M (1974) Cross-validatory choice and assessment of statistical predictions. J R Stat Soc Ser B 36(2):111–147

Strang KD (2012) Importance of verifying queue model assumptions before planning with simulation software. Eur J Oper Res 218(2):493–504

Sun Y, Wong ACM (2007) Interval estimation for the normal correlation coefficient. Stat Probab Lett 77(17):1652–1661

Tan MHY (2015) Monotonic quantile regression with Bernstein polynomials for stochastic simulation. Technometrics, (in press)

Tofallis C (2008) Selecting the best statistical distribution using multiple criteria. Comput Ind Eng 54(3):690–694

Tofallis C (2015) A better measure of relative prediction accuracy for model selection and model estimation. J Oper Res Soc 66:524

Turner AJ, Balestrini-Robinson S, Mavris D (2013) Heuristics for the regression of stochastic simulations. J Simul 7:229–239

Van Groenendaal WJH (1998) The economic appraisal of natural gas projects. Oxford University Press, Oxford

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690

Vonk Noordegraaf A (2002) Simulation modelling to support national policy making in the control of bovine herpes virus. Doctoral dissertation, Wageningen University, Wageningen

Wang B, Mishra SN, Mulekar MS, Mishra N, Huang K (2010) Comparison of bootstrap and generalized bootstrap methods for estimating high quantiles. J Stat Plan Inference 140(10):2926–2935

Wang Y (2013) On efficiency properties of an R-square coefficient based on final prediction error. Stat Probab Lett 83(10):2276–2281

Wen M-J, Chen S-Y, Chen HJ (2007) On testing a subset of regression parameters under heteroskedasticity. Comput Stat Data Anal 51(12):5958–5976

Xu J, Zhang S, Huang E, Chen C-H, Lee H, Celik N (2014) Efficient multi-fidelity simulation optimization. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, pp 3940–3951

Yang F, Ankenman B, Nelson B (2007) Efficient generation of cycle time-throughput curves through simulation and metamodeling. Nav Res Logist 54:78–93

Yeo I-K, Johnson R (2000) A new family of power transformations to improve normality or symmetry. Biometrika 87:954–959

You J, Chen G (2006) Wild bootstrap estimation in partially linear models with heteroscedasticity. Stat Probab Lett 76(4):340–348

4

Screening the Many Inputs of Realistic Simulation Models

This chapter is organized as follows. Section 4.1 introduces “screening” defined as searching for the really important inputs in experiments with simulation models that have “very many” inputs (say, hundreds of inputs); this section also gives an overview of several screening methods. Section 4.2 explains a screening method called *sequential bifurcation* (SB); for simplicity, this section assumes deterministic simulation and first-order polynomial metamodels. Section 4.3 explains SB for deterministic simulations and second-order polynomial metamodels that satisfy the “heredity” assumption; this assumption states that if a specific input has no first-order effect, then this input has no second-order effects either. Section 4.4 explains SB for random simulations with a fixed number of replications per input combination. Section 4.5 explains SB for random simulations with a variable number of replications determined through Wald’s sequential probability ratio test (SPRT). Section 4.6 discusses *multiresponse sequential bifurcation* (MSB), which extends SB to problems with multiple types of simulation responses (multivariate output). Section 4.7 discusses validation of the SB and MSB assumptions. Section 4.8 summarizes the major conclusions of this chapter. The chapter ends with solutions for the exercises, and a list with references for further study.

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_4

4.1 Introduction

Screening or *factor screening* means searching for the really important factors or inputs among the many inputs that can be varied in an experiment with a given simulation model or real system. We assume that effects are *sparse*; i.e., only a few inputs among these many inputs are really important. Many simulation publications speak of *the curse of dimensionality*; see the extensive and recent survey in Viana et al. (2014) and also Singh et al. (2014). In business and economics, the *Pareto* principle or *20–80* rule states that only a few inputs (namely, 20 %) are really important—or "active", as some authors say. In philosophy, the *parsimony* principle or *Occam's razor* implies that a simpler explanation is preferred to a more complex explanation—all other things being equal. In Sect. 2.4.1 we have already mentioned the psychological argument—originally formulated in Miller (1956)—stating that human capacity for processing information is limited to seven plus or minus two inputs. So we conclude that there is really a need for screening in the design and analysis of simulation experiments (DASE).

Note: Compared with mathematical programming models, simulation models have relatively few inputs—even if we speak of "many" simulation inputs; see Tenne and Goh (2010) for further discussion.

To illustrate the need for screening, we now summarize two practical simulation models with many inputs; one example is deterministic and one example is random.

Example 4.1 *"The Netherlands National Institute for Public Health and the Environment"—abbreviated to RIVM in Dutch—is a research institute of the Ministry of Health, Welfare and Sport; this institute developed a deterministic simulation model (called "IMAGE") that explains the worldwide increase of temperatures known as the "greenhouse phenomenon". In a submodel of this simulation model, Bettonvil and Kleijnen (1997) varies 281 inputs. After simulating only 154 input combinations (scenarios), a shortlist with 15 inputs is presented; this list includes some inputs that the ecological experts had not expected to be important! Next, this shortlist was used to support national policy makers in their decision-making. It is also important to know which inputs are unimportant so decision-makers are not bothered by details about these inputs.*

Example 4.2 *Persson and Olhager (2002) develops a random supply-chain simulation for the Ericsson company in Sweden, simulating only nine input combinations. Kleijnen et al. (2006) also uses this simulation model, but considers 92 inputs. Even an experiment with the minimum number of values per input—namely, two values—would require* $2^{92} \approx 5 \times 10^{27}$. *Changing one input at a time would still require 93 input combinations; moreover, such an approach does not enable the estimation of any fac-*

tor interactions (also see Sect. 2.3.2). In Sect. 4.4.2 we shall show how we actually simulate only 21 combinations—each combination replicated five times—to identify a shortlist with the 11 most important inputs among the original 92 inputs. Note that in this case study a single replication takes 40 minutes, after modifying the simulation code that originally took 3 hours per replication.

We emphasize that the importance of inputs depends on the *experimental domain*, which is also called the *experimental area* or the *experimental frame* (also see Sect. 2.3). Information on this domain should be given by the users of the given simulation model, including realistic ranges of the individual inputs and limits on the admissible input combinations (e.g., some input values must add up to 100 %; also see the mixture designs in Sect. 2.10.2). So, in practice, user involvement is crucial for the application of screening.

To solve the screening problem, several types of screening designs may be used. We focus on designs that treat the simulation as a *black box*; i.e., only the inputs and outputs of the simulation model are observed (also see Definition 2.1). We use the symbol n to denote the number of input combinations actually simulated, and the symbol k to denote the number of inputs changed in the simulation experiment. We summarize the following four types of screening designs.

- *Resolution-III* (R-III) designs require $n \approx k$ input combinations to estimate the k first-order effects (see Sect. 2.4), and are often called screening designs in the literature on the classic design of experiments (DOE). By definition, R-III designs give unbiased estimators of these first-order effects if there are no higher-order effects. Definition 2.12 implies that a *saturated* design has a number of input combinations equal to the number of parameters to be estimated; so, if a R–III design is saturated, then $n = 1 + k$. Actually, R-III designs are either saturated or nearly saturated (see again Sect. 2.4). Related to these R-III designs are so-called "definitive screening" designs, which are resolution-IV (R-IV) designs and require $n \approx 2k$ combinations; see Jones and Nachtsheim (2015).

- *Supersaturated* designs have fewer input combinations than inputs, by definition: $n < \ k$. These designs assume that the designs are not sequential; by definition, sequential designs enable us to learn about the input/output (I/O) behavior of the simulated system as we collect data on this system before we decide on the next input combination to be simulated. Obviously, nonsequential or *one-shot* designs are less efficient. Kleijnen (1975) compares supersaturated, R-III, and group-screening designs; group-screening will be discussed under the next bullet. A bibliography on supersaturated designs is given on the following website maintained by the Indian Agricultural Statistics Research Institute (IASRI):

http://www.iasri.res.in/design/Supersaturated_Design/SSD/Supersaturated.html.

We also refer to some more recent articles than Kleijnen (1975); namely, Claeys-Bruno et al. (2011), Draguljić et al. (2014), Edwards and Mee (2011), Holcomb et al. (2007), Koukouvinos et al. (2011), Phoa et al. (2015), Sarkar et al. (2009), and Xing et al. (2013).

- *Group-screening* designs aggregate (or confound) individual inputs into groups so that the k original individual inputs may be evaluated in less than k input combinations. Consequently, these designs are supersaturated—but they are sequential; i.e., they are executed in two or more steps or stages. There are several types of group-screening designs. Examples are one-factor-at-a-time (OAT), Morris's OAT, Cotter's design, Andres's iterated fractional factorial design (IFFD), multi-stage group screening, and sequential bifurcation (SB); see Borgonovo and Plischke (2015) Boukouvalas et al. (2014), Campolongo et al. (2007), Campolongo et al. (2000), De Vos et al. (2006), Fédou and Rendas (2015), Huang et al. (2015), Khare et al. (2015), Kleijnen (1975, 2008, pp. 159–160), Martin et al. (2016), Morris (2006), Pujol (2009), Schonlau and Welch (2006), Shen et al. (2010), and Van der Sluijs et al. (2005).

  Note: Originally, group screening was developed in Dorfman (1943), to detect syphilis among men called up for induction and subjected to a blood test; also see Xiong and Ding (2015). Watson (1961) extends this screening to the screening of inputs in experiments with real systems.

- *Frequency domain experiments* (FDE) oscillate the input values (levels) during a simulation run, whereas all other types of designs keep these values constant during the simulation run. More precisely, each input has its own carefully chosen oscillation frequency. FDE require only $n = 2$ input combinations; namely, one combination with all k inputs kept constant during the simulation run, and one combination that is run while each input oscillates at its own frequency. FDE try to find which input oscillations significantly affect observed output oscillations. For this analysis FDE use Fourier spectral analysis. Originally, Schruben and Cogliano (1987) proposed this approach. Sanchez et al. (2006) applies FDE for second-order polynomial metamodels, including an example of a simulation model with $k = 34$ inputs. Sohi et al. (2012) proposes an alternative test statistic to control the error rates of type-I and type-II.

The preceding types of screening designs are based on different mathematical assumptions concerning the *smoothness* of the I/O function implied by the underlying simulation model, possible *monotonicity* of this function, etc.; e.g., Moon et al. (2010) assumes a Kriging metamodel, Rosen and Guharay (2013) assumes a neural network metamodel, and

Shih et al. (2014) assumes multivariate adaptive regression splines (MARS). In this chapter we focus on SB, because SB is very efficient and effective if its assumptions are satisfied, as we shall see below.

Note: Mathematically speaking, SB resembles binary search, which is a well-known procedure in computer science. SB, however, not only estimates *which* inputs are important, but also estimates the *magnitudes* of the effects of the important inputs.

We repeat that the assumption of a *fixed* sample size in R-III and supersaturated designs does not hold in *sequential* designs, which select the next input combination after analyzing the preceding I/O data. Such an analysis may also give designs that are not purely sequential, but are multi-stage or two-stage. Moreover, these designs are *customized*; i.e., they account for the specific simulation model.

4.2 Sequential Bifurcation (SB) for Deterministic Simulations and First-Order Polynomial Metamodels

Originally, SB was developed in Bettonvil (1990), a doctoral dissertation.

Note: This dissertation is summarized in Bettonvil and Kleijnen (1997). Later on, other authors extended SB; see Frazier et al. (2012), Kleijnen (2008, p. 160, 2009), Sanchez et al. (2009), and Shen and Wan (2009). Some specific extensions will be mentioned below.

In this section we explain the basic idea of SB, assuming *deterministic* simulation so the I/O function is not disturbed by noise. Furthermore, we assume that this I/O function can be adequately approximated through a *first-order polynomial* metamodel—which is the simplest metamodel that can still reflect input effects (a zero-order polynomial implies that not a single input affects the output; a second-order polynomial will be discussed in later sections). So the metamodel with the response (output) y, the k inputs z_j $(j = 1, \ldots, k)$ measured on the original scale (not coded, scaled, or standardized), and the approximation error (or fitting error) e is

$$y = \gamma_0 + \gamma_1 z_1 + \ldots + \gamma_k z_k + e. \tag{4.1}$$

Finally, we assume that the *signs* of these first-order effects γ_j are known so that we may define the lower and upper bounds l_j and u_j of the input z_j such that all k first-order effects are nonnegative: $\gamma_j \geq 0$ (we follow the notation of Sect. 2.3.1).

The metamodel in Eq. (4.1) and the known signs $\gamma_j \geq 0$ imply that the inputs may be ranked (sorted) by their first-order effects; i.e., the most important inputs are the ones with the largest first-order effects; the least important inputs are the ones with the effects closest to zero. If the metamodel is valid (or "adequate"), then by definition the approximation error has zero expected value: $E(e) = 0$.

SB is illustrated in Fig. 4.1, which is adapted from Bettonvil and Kleijnen (1997). This plot applies SB to an example in Jacoby and Harrison (1962). The example has $k = 128$ inputs, but only three important inputs; namely,the inputs labeled 68, 113, and 120. The symbols $\leftarrow$ and $\rightarrow$ show which simulation outputs estimate group effects; $\downarrow$ means that a group is split into two subgroups; $\uparrow$ refers to the individual input that SB finally identifies as important. Furthermore, w_j denotes the simulation output when the first j inputs are at their high levels h_j and the remaining inputs are at their low levels l_j; this symbol w_j is also used in Shi et al. (2014a), whereas $w_{(j)}$ is used in Bettonvil (1990) and other publications. Finally, $\gamma_{j'-j}$ denotes the sum of the first-order effects of inputs j' through j; to simplify the notation in this plot, we do not display the hat in $\widehat{\gamma}_{j'-j}$ that denotes an estimator.

In general, SB is *sequential;* i.e., SB consists of a sequence of steps. In the first (initial) step, SB aggregates all k inputs into a single group, and checks whether or not that group of inputs has an important effect. Input j is called *important* if $\gamma_j > \Delta$ where $\Delta \geq 0$ is determined by the users; obviously, Δ depends on the problem. So in this step, SB obtains the simulation output w when all k simulation inputs are "low"; this output may be denoted by $w(\mathbf{z} = \mathbf{l})$ where $\mathbf{z} = (z_1, \ldots, z_k)'$ and $\mathbf{l} = (l_1, \ldots, l_k)'$. In this step, SB also obtains w when all inputs are "high", denoted by $w(\mathbf{z} = \mathbf{h})$ where $\mathbf{h} = (h_1, \ldots, h_k)'$. Obviously, if all inputs have zero effects so $\gamma_j = 0$ $(j = 1, \ldots, k)$, then the values of these two outputs are the same: $w(\mathbf{z} = \mathbf{l}) = w(\mathbf{z} = \mathbf{h})$. However, if one or more inputs have positive effects (so $\exists j$: $\gamma_j > 0$), then these two outputs differ: $w(\mathbf{z} = \mathbf{l}) < w(\mathbf{z} = \mathbf{h})$. In practice, not all k inputs have zero effects. We point out that it may happen that all effects are unimportant so $0 \leq \gamma_j < \Delta$, but that $w(\mathbf{z} = \mathbf{h}) - w(\mathbf{z} = \mathbf{l}) > \Delta$. If SB finds that the group has an important effect, then the next step of SB splits the group into two subgroups, which explains the term *bifurcation.* Let k_1 denote the size of subgroup 1 and k_2 the size of subgroup 2, so $k_1 + k_2 = k$. "Good" values for k_1 and k_2 will be discussed below. For the time being, we may suppose that the two subgroups have "approximately" the same size; e.g., Fig. 4.1 shows $k_1 = k_2 = 64$.

In this next step, SB obtains the simulation output w when all k_1 simulation inputs within subgroup 1 are "high". So in this step, SB obtains w_{k_1}. SB compares this w_{k_1} with $w_0 = w(\mathbf{z} = \mathbf{l})$; if $w_{k_1} - w_0 < \Delta$, then none of the individual inputs in subgroup 1 is important and SB eliminates this subgroup from further experimentation.

SB also compares this w_{k_1} with $w_k = w(\mathbf{z} = \mathbf{h})$; if $w_k - w_{k_1} < \Delta$, then none of the individual inputs in subgroup 2 is important and SB eliminates this subgroup from further experimentation. However, in the example of Fig. 4.1 the result $w_k - w_{k_1} < \Delta$ is impossible, because at least one input is important and this input is labeled such that this input is a member of subgroup 2.

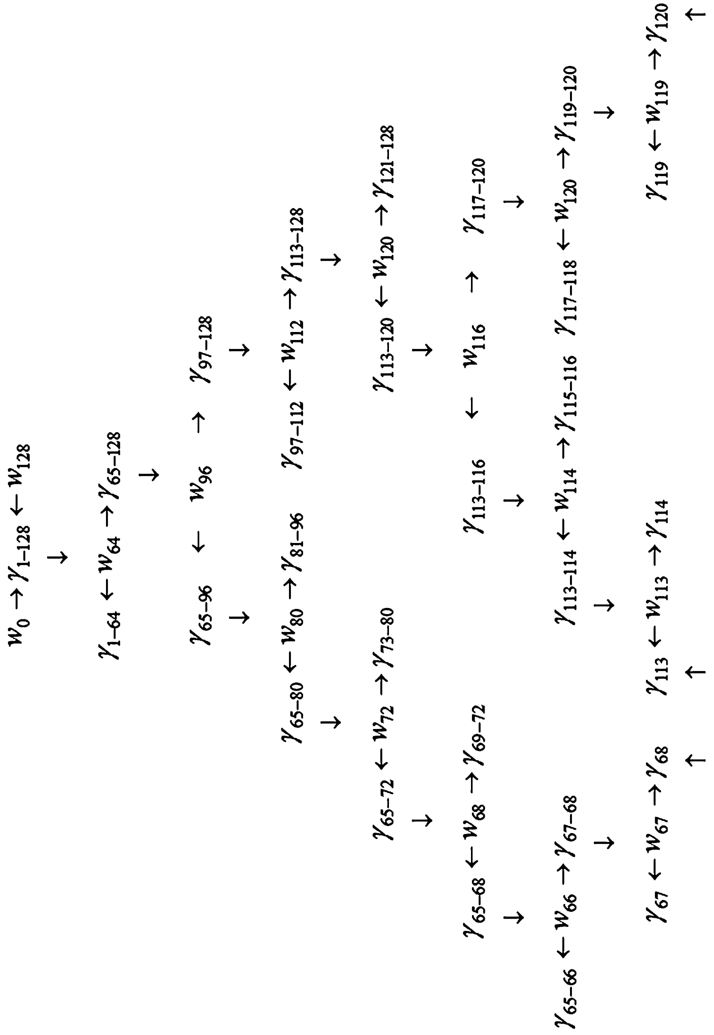

$w_0 \rightarrow y_{1-128} \leftarrow w_{128}$
$y_{1-64} \leftarrow w_{64} \rightarrow y_{65-128}$
$y_{65-96} \leftarrow w_{96} \rightarrow y_{97-128}$
$y_{65-80} \leftarrow w_{80} \rightarrow y_{81-96}$
$y_{97-112} \leftarrow w_{112} \rightarrow y_{113-128}$
$y_{65-72} \leftarrow w_{72} \rightarrow y_{73-80}$
$y_{113-120} \leftarrow w_{120} \rightarrow y_{121-128}$
$y_{65-68} \leftarrow w_{68} \rightarrow y_{69-72}$
$y_{113-116} \leftarrow w_{116} \rightarrow y_{117-120}$
$y_{65-66} \leftarrow w_{66} \rightarrow y_{67-68}$
$y_{113-114} \leftarrow w_{114} \rightarrow y_{115-116}$
$y_{117-118} \leftarrow w_{120} \rightarrow y_{119-120}$
$y_{67} \leftarrow w_{67} \rightarrow y_{68}$
$y_{113} \leftarrow w_{113} \rightarrow y_{114}$
$y_{119} \leftarrow w_{119} \rightarrow y_{120}$

FIGURE 4.1. SB example with 128 inputs including only three important inputs

SB continues splitting important subgroups into smaller subgroups, and discards unimportant subgroups. It may happen that SB finds both subgroups to be important; e.g., w_{96} in Fig. 4.1 leads to further experimentation with two important subgroups. Finally, SB identifies and estimates all individual inputs that are not in subgroups identified as unimportant; see the symbol $\uparrow$ at the bottom of Fig. 4.1.

In this section we standardize (scale, code) the inputs such that the high values of the original inputs correspond with the value 1 for the standardized inputs, but the low values of the original inputs correspond with the value 0 (instead of -1) for the standardized inputs (also see Sect. 2.3.1):

$$x_j = a_j + b_j z_j \quad \text{with} \quad a_j = \frac{-l_j}{u_j - l_j};\ b_j = \frac{1}{u_j - l_j};\ j = 1, \ldots, k. \tag{4.2}$$

If an original input is qualitative, then we randomly associate its levels with the standardized values 0 and 1. So Eq. (4.1) for the original inputs z implies the following metamodel for the standardized inputs x:

$$y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + e. \tag{4.3}$$

To *estimate* β_j in Eq. (4.3), it is most efficient to experiment with only two levels (values) per input (see again Chap. 2). In practice, it is important that these levels are realistic extreme values; i.e., the users of the underlying simulation model should provide these values. We also refer to the discussion of scaling in Wan et al. (2006).

Below Eq. (4.1) we assumed $\gamma_j \geq 0$ for the original inputs, so now we assume $\beta_j \geq 0$ for the scaled inputs. Without this assumption first-order effects might cancel each other within a group. Part (a) of Fig. 4.2 illustrates that the "known signs" assumption is related to the "monotonicity" of the I/O function, defined as follows.

Definition 4.1 *The function $w = f(x)$ is called monotonically increasing if $w(x = x_1) \leq w(x = x_2)$ if $x_1 \leq x_2$.*

Obviously, we can define the inputs such that if the function is monotonically decreasing in the original inputs z_j, then this function becomes monotonically increasing in the standardized inputs x_j.

Our experience shows that in practice the users often do know the *signs*. One example is the RIVM case study in Example 4.1, in which the ecological experts could specify the signs of all 281 inputs. Another example is the random simulation in Example 4.2, in which some inputs refer to transportation speeds so the higher these speeds, the lower the "work in process (WIP)" and hence the lower the cost; this cost is the output of interest in this SB experiment.

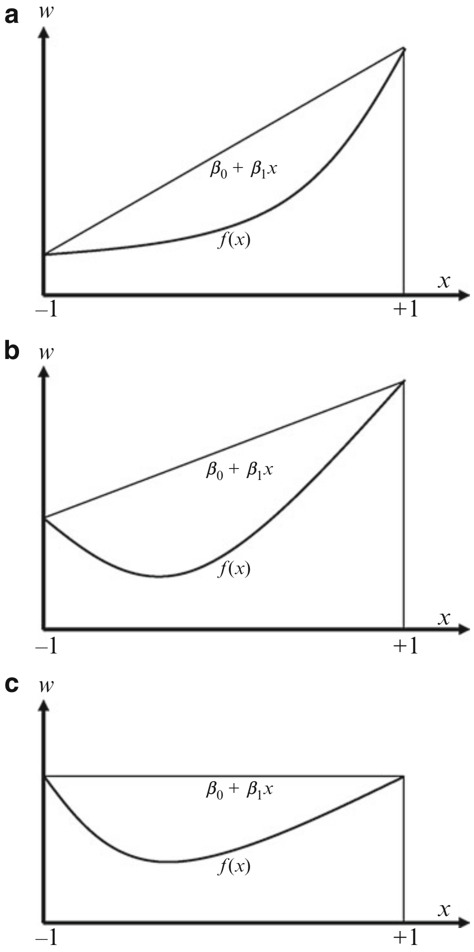


FIGURE 4.2. Known signs of I/O function: (**a**) monotonic (**b**) nonmonotonic (**c**) nonmonotonic with misleading sign

Note: More examples of known monotonic I/O functions are given in other publications, including publications on so-called *isotonic regression*, which assumes that the mean function is monotone increasing (or decreasing); see Antoniadis et al. (2007), Draguljić et al. (2014), Kleijnen (2008, p. 162), Lim and Glynn (2006), Murray et al. (2013), Siem et al. (2008), Tan (2015), Wang and Xue (2015), and Wu et al. (2015). However, if the signs are unknown, then SB may be preceded by a saturated or nearly saturated R-III design; see Oh et al. (2009) and Sanchez et al. (2009), detailing this so-called fractional factorial controlled sequential bifurcation (FFCSB) and several variants.

Part (b) of Fig. 4.2 illustrates a *nonmonotonic* I/O function. Nonetheless, switching the standardized input from −1 to +1 increases the output so SB will find that this input is important.

Part (c) of Fig. 4.2 gives a ("pathological"?) counterexample; i.e., the I/O function is not monotonic and happens to give the same output values for the two observed input levels −1 and +1 so the input effect seems to be zero and SB will eliminate this input.

Nevertheless, if in a particular case study it is hard to specify the signs of a few specific inputs, then we should treat these inputs *individually*; i.e., we should not group these inputs with other inputs in SB. For example, De Vos et al. (2006) creates some subgroups of size one in a multi-stage group-screening design; this design is less efficient than SB, but (like SB) it also uses aggregation. Treating such inputs individually is safer than assuming a negligible probability of cancellation within a subgroup.

The *efficiency* of SB—measured by the number of simulated input combinations (and hence computer time)—improves if the individual inputs are labeled such that inputs are placed in increasing order of importance, as proven in Bettonvil (1990, p. 44). This labeling implies that the important inputs are clustered. To realize this efficiency gain, it is crucial to utilize prior knowledge of users and analysts about the real system being simulated; e.g., in the Ericsson case-study of Example 4.2, the input "demand" is placed at the very end of the list with 92 individual inputs. The efficiency further improves when placing similar inputs within the same subgroup; e.g., in the Ericsson case-study, all "test yield" inputs are placed together, because the conjecture is that if one yield input is unimportant, then all yield inputs are unimportant too. Finally, the efficiency increases if input subgroups are split such that the number of inputs for the first new subgroup is a power of two; e.g., split a group of 48 inputs into a subgroup of 32 (= 2^5) inputs and a subgroup of the remaining 16 inputs so the important inputs are placed in the smallest subgroup—assuming the inputs are sorted from unimportant to most important. However, we do not recommend such splitting if it implies splitting a group of related inputs. Anyhow, we conclude that splitting a group into subgroups of *equal* size—like some authors do—is not necessarily optimal. For further discussion we refer to Bettonvil (1990, pp. 40–43).

Analogously to the symbol w_j, we let the symbol w_{-j} denote the observed simulation output with the inputs 1 through j set to their low levels and the remaining inputs set to their high levels. Notice that $w_0 = w_{-k}$. Combining w_j and w_{-j} with the scaling in Eq. (4.2) and the metamodel in Eq. (4.3), we obtain

$$E(w_j) = \beta_0 + \sum_{h=1}^{j} \beta_h \tag{4.4}$$

and

$$E(w_{-j}) = \beta_0 - \sum_{h=1}^{j} \beta_h. \tag{4.5}$$

Combining Eqs. (4.4) and (4.5), we obtain

$$E(w_j) - E(w_{-j}) = 2\sum_{h=1}^{j} \beta_h. \tag{4.6}$$

Let $\beta_{j'-j}$ denote the sum of the first-order effects of the standardized inputs j' through j:

$$\beta_{j'-j} = \sum_{h=j'}^{j} \beta_h. \tag{4.7}$$

A simple unbiased estimator of this group effect $\beta_{j'-j}$ follows from Eq. (4.6):

$$\widehat{\beta}_{j'-j} = \frac{w_j - w_{j'-1}}{2}. \tag{4.8}$$

Consequently, the *individual* first-order effect of input j may be estimated through the analogue of Eq. (4.8):

$$\widehat{\beta}_j = \frac{w_j - w_{j-1}}{2}. \tag{4.9}$$

Note: Ankenman et al. (2006) derives a more complicated estimator that uses the ordinary least squares (OLS) criterion with the k additional constraints $\widehat{\beta}_j \geq 0$ $(j = 1, \ldots, k)$, assuming random simulation (deterministic simulation, which is the focus of this section, is a limiting case of random simulation). This so-called “polytope” method requires fewer combinations to be simulated, but it is also more complicated because it requires the solution of a linear programming (LP) problem after each additional observation; this LP problem arises because the method computes the OLS estimate—so it minimizes the sum of squared residuals, SSR, defined in Eq. (2.11)—under the constraints stipulating that all regression coefficients be nonnegative. Moreover this method assumes a first-order polynomial, whereas we shall also present simple estimators like Eq. (4.8) for second-order polynomials (see Sect. 4.3).

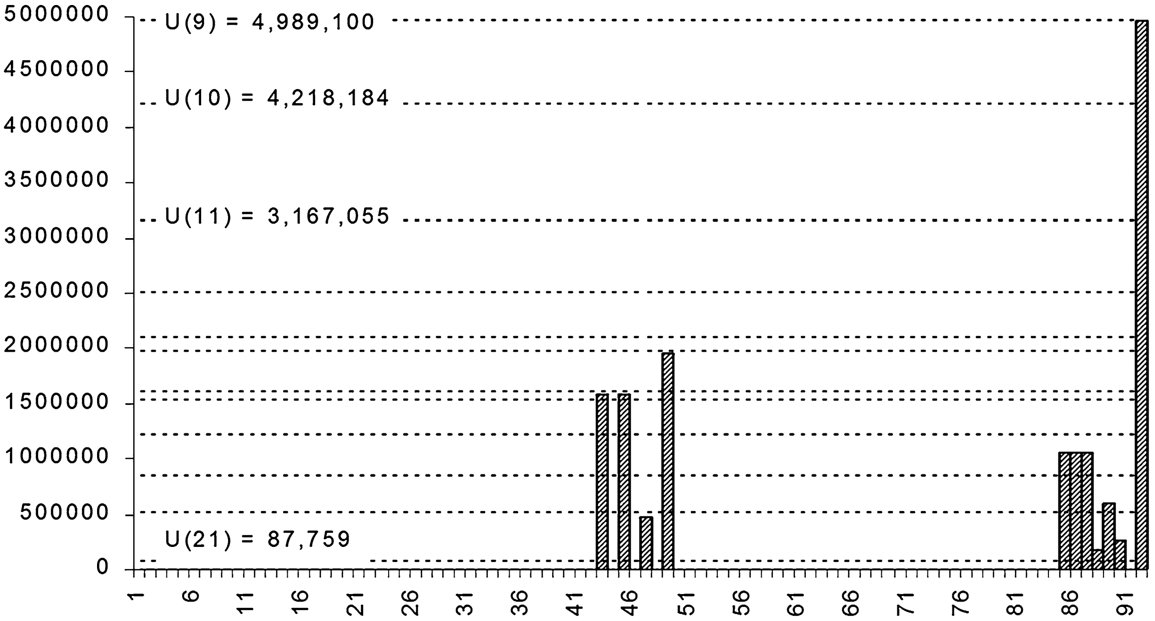


FIGURE 4.3. Upper limit $U(i)$ after step i $(i = 9, \ldots, 21)$ and estimated individual main effects (*shaded bars*) versus input label j $(j = 1, \ldots, 92)$ in the Ericsson supply-chain simulation

The way SB proceeds may be interpreted through the following *metaphor* that is inspired by Japanese zero-inventory management; also see Cachon and Terwiesch (2006, Fig. 10.6). Figure 4.3 reproduced from Kleijnen et al. (2006) suggests a lake with murky water that is controlled through a dam. The goal of this control is to identify the highest (most important) rocks (actually, SB not only identifies, but also measures the height of these "rocks"). The dam is controlled in such a way that the level of the water slowly drops. Obviously, the highest rock first emerges from the water. The most-important-but-one rock turns up next. And the water level continues to decrease .... Actually, SB may stop when we feel that all the "important" inputs are identified; once we stop, we know that all remaining (unidentified) inputs have effects that are smaller than the effects of the inputs that have been identified so far.

Moreover, the aggregated effect of a given (sub)group is an upper limit for the value of any individual first-order effect within that group; see U in Fig. 4.3 (actually, this plot illustrates a more complicated case-study; namely, the Ericsson random simulation in Example 4.2, assuming a second-order polynomial). If we must terminate SB prematurely (e.g., because our computer breaks down or our users get impatient), then SB still allows identification of the inputs with first-order effects larger than the current upper limit U. For example, Fig. 4.3 shows that if SB is terminated after Step 11, then the most important input—namely, the input labelled 92, which is demand for the product—has already been identified, and its first-order effect has been estimated; none of the other inputs has a first-order effect exceeding that of the input 92.

4.3 SB for Deterministic Simulations and Second-Order Polynomial Metamodels

In this section we assume that a valid metamodel is a second-order polynomial plus approximation error e with zero mean so $E(e) = 0$:

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j x_j + \sum_{j=1}^{k-1} \sum_{j'=j+1}^{k} \beta_{j;j'} x_j x_{j'} + \sum_{j=1}^{k} \beta_{j;j} x_j^2 + e, \tag{4.10}$$

which we have already discussed in Sect. 2.8 assuming a relatively small number of inputs so classic designs could be applied. Actually, Bettonvil (1990) and Bettonvil and Kleijnen (1997) ignore the purely quadratic effects $\beta_{j;j}$.

In this section (unlike the preceding section, which includes Eq. (4.2)) we standardize the original inputs z such that the standardized inputs x are either -1 or 1 in the experiment (also see Sect. 2.3.1):

$$x_j = a_j + b_j z_j \quad \text{with } a_j = \frac{l_j + u_j}{l_j - u_j} \text{ and } b_j = \frac{2}{u_j - l_j}. \tag{4.11}$$

Moreover, we assume that if input j has no first-order effect so $\beta_j = 0$, then this input has no second-order effects either so $\beta^2_{j;j} = 0$ and $\beta_{j;j'} = 0$ $(j' \neq j)$. This assumption is the analogue of the *heredity* assumption in Wu and Hamada (2009).

Note: Heredity is related to "functional marginality" discussed in Tsai et al. (2007). Heredity is questioned in Draguljić et al. (2014), and Rosen and Guharay (2013); also see Archer et al. (1997).

In Theorem 2.1 we have already presented the *foldover* principle, used to construct resolution-IV (R-IV) designs from R-III designs. This principle implies that we simulate the "mirror" input combination besides the original combination; i.e., −1 and 1 in the original design become 1 and −1 in the mirror design. Likewise, SB enables the estimation of first-order effects unbiased by second-order effects if SB simulates the mirror input of the original input in its sequential design. Obviously, SB now doubles the number of simulated combinations compared with SB assuming a first-order polynomial.

More specifically, the second-order polynomial in Eq. (4.10) gives the analogue of Eq. (4.4):

$$E(w_j) = \beta_0 + \sum_{h=1}^{j} \beta_h - \sum_{h=j+1}^{k} \beta_h + \sum_{h=1}^{k} \beta^2_{h;h} + \sum_{h=1}^{j-1} \sum_{h'=h+1}^{j} \beta_{h;h'} + \sum_{h=j+1}^{k-1} \sum_{h'=h+1}^{k} \beta_{h;h'} - \sum_{h=1}^{j} \sum_{h'=j+1}^{k} \beta_{h;h'}. \quad (4.12)$$

For example, if in Eq. (4.12) $k = 92$ (as in Example 4.2) and (say) $j = 49$, then

$$\begin{aligned} E(w_{49}) = \beta_0 &+ (\beta_1 + \ldots + \beta_{49}) - (\beta_{50} + \ldots + \beta_{92}) \\ &+ (\beta_{1;1} + \ldots + \beta_{92;92}) + (\beta_{1;2} + \ldots + \beta_{48;49}) \\ &+ (\beta_{50;51} + \ldots + \beta_{91;92}) - (\beta_{1;50} + \ldots + \beta_{49;92}). \end{aligned}$$

Likewise, the metamodel in Eq. (4.10) gives the analogue of Eq. (4.5):

$$E(w_{-j}) = \beta_0 - \sum_{h=1}^{j} \beta_h + \sum_{h=j+1}^{k} \beta_h + \sum_{h=1}^{k} \beta^2_{h;h} + \sum_{h=1}^{j-1} \sum_{h'=h+1}^{j} \beta_{h;h'} + \sum_{h=j+1}^{k-1} \sum_{h'=h+1}^{k} \beta_{h;h'} - \sum_{h=1}^{j} \sum_{h'=j+1}^{k} \beta_{h;h'}. \quad (4.13)$$

For example, $k = 92$ and $j = 49$ give

$$\begin{aligned} E(w_{-49}) = \beta_0 &- (\beta_1 + \ldots + \beta_{49}) + (\beta_{50} + \ldots + \beta_{92}) \\ &+ (\beta_{1;1} + \ldots + \beta_{92;92}) + (\beta_{1;2} + \ldots + \beta_{48;49}) \\ &+ (\beta_{50;51} + \ldots + \beta_{91;92}) - (\beta_{1;50} + \ldots + \beta_{49;92}). \end{aligned}$$

Subtracting Eq. (4.13) from Eq. (4.12) cancels all second-order effects.

An unbiased estimator of the first-order group effect is the analogue of Eq. (4.8):

$$\widehat{\beta}_{j'-j} = \frac{(w_j - w_{-j}) - (w_{j'-1} - w_{-(j'-1)})}{4}. \tag{4.14}$$

An unbiased estimator of the individual effect is the analogue of Eq. (4.9):

$$\widehat{\beta}_j = \frac{(w_j - w_{-j}) - (w_{j-1} - w_{-(j-1)})}{4}. \tag{4.15}$$

Exercise 4.1 *What is the mirror scenario of the extreme input combination that has all k inputs at their low levels?*

If we suspect that the heredity assumption is violated for a specific input, then we should not use SB to investigate that particular input, but we should investigate that input after the screening phase.

Note: SB with mirror scenarios does not enable estimation of *individual* interactions, but it does show whether interactions are important—as follows. First we estimate the first-order effects from the original input combinations ignoring the mirror combinations. Next we compute the first-order effects from the outputs for the mirror combinations. If these two types of estimators give roughly the same values, then we may conclude that interactions are unimportant. An example is the ecological simulation in Example 4.1. In that example, the input values change relatively little (because larger changes would give unrealistic simulation output); because of these small changes a first-order polynomial is adequate. However, the Ericsson case-study in Example 4.2 gives interactions that turn out to be important. In a follow-up experiment with the inputs declared to be important in SB, we may estimate the individual interactions through a resolution-V (R-V) design (these designs are discussed in Sect. 2.7). SB with mirror observations may give a different path through the list of individual inputs; e.g., the path in Fig. 4.4 (displayed in the next section) may change.

4.4 SB for Random Simulations and Constant Number of Replications

In Sect. 4.4.1 we present the SB method for a constant number of replications m per input combination. In Sect. 4.4.2 we present a case study; namely, Ericsson's supply-chain simulation.

4.4.1 The SB Method

We extend our notation such that $w_{j;r}$ denotes replication r of the simulation output with the inputs 1 through j at their high levels and the remaining inputs at their low levels; we assume a fixed number of replications m per simulated input combination so $r = 1, \ldots, m$.

Assuming a *first-order* polynomial, we obtain the following unbiased estimator of the group effect based on replication r:

$$\widehat{\beta}_{(j'-j);r} = \frac{w_{j;r} - w_{(j'-1);r}}{2},$$

which is the analogue of Eq. (4.8). The analogous estimator of an individual first-order effect is

$$\widehat{\beta}_{j;r} = \frac{w_{j;r} - w_{(j-1);r}}{2}.$$

Assuming a *second-order* polynomial metamodel, we obtain the following unbiased estimator of the first-order group effect based on replication r that is the analogue of Eq. (4.14):

$$\widehat{\beta}_{(j'-j);r} = \frac{(w_{j;r} - w_{(-j);r}) - (w_{(j'-1);r} - w_{-(j'-1);r})}{4}. \tag{4.16}$$

And the analogous estimator of an individual effect is

$$\widehat{\beta}_{j;r} = \frac{(w_{j;r} - w_{(-j);r}) - (w_{(j-1);r} - w_{-(j-1);r})}{4}. \tag{4.17}$$

Whether we assume a first-order or a second-order polynomial, the m replications enable us to estimate the mean and the variance for each aggregated and individual estimated effect; also see Eq. (3.33). For example, Eq. (4.17) gives

$$\overline{\widehat{\beta}}_{j} = \frac{\sum_{r=1}^{m} \widehat{\beta}_{j;r}}{m} \text{ and } s(\overline{\widehat{\beta}}_{j}) = \sqrt{\frac{\sum_{r=1}^{m} (\widehat{\beta}_{j;r} - \overline{\widehat{\beta}}_{j})^2}{m(m-1)}}. \tag{4.18}$$

This variance estimator allows unequal output variances and common random numbers (CRN). Consequently, the individual estimated regression parameters $\widehat{\beta}_j$ may be tested through the t-statistic with $m - 1$ degrees of freedom:

$$t_{m-1} = \frac{\overline{\widehat{\beta}}_{j} - \beta_j}{s(\overline{\widehat{\beta}}_{j})}. \tag{4.19}$$

In SB we apply a one-sided test because all individual first-order effects are assumed to be nonnegative; our "favorite" or "null" hypothesis (H_0) is that the SB assumption holds:

$$H_0 : \beta_j > 0 \quad \text{versus} \quad H_1 : \beta_j = 0. \tag{4.20}$$

4.4.2 Case Study: Ericsson's Supply Chain

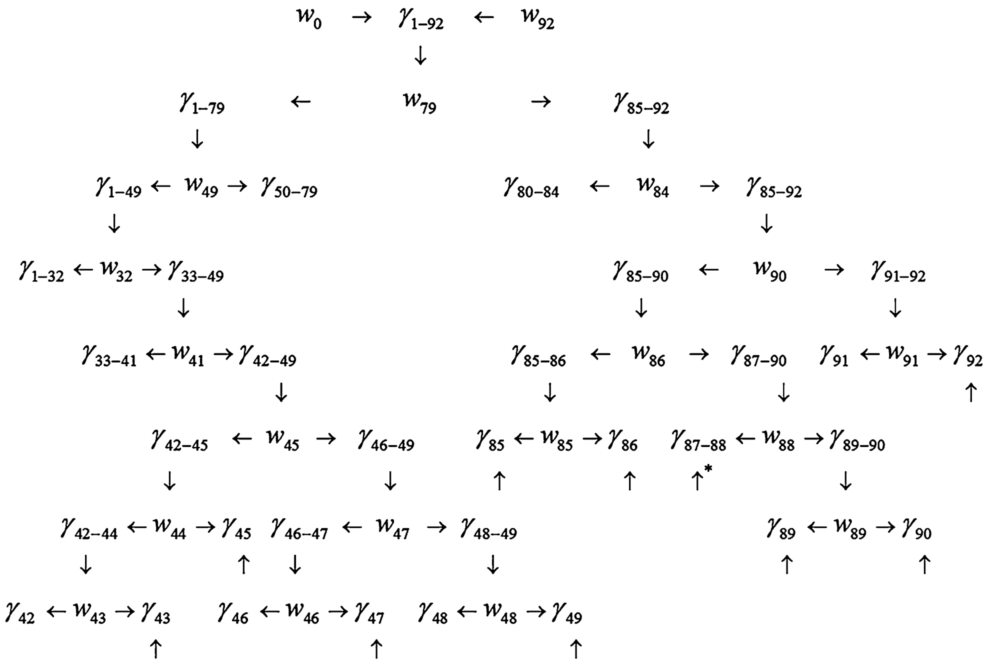
We return to the Ericsson simulation model in Example 4.2. For this model, Kleijnen et al. (2006) examines $k = 92$ inputs and obtains $m = 5$ replications for each simulated input combination. Table 4.1 gives the simulation outputs for each replication of the two extreme combinations,which have the first $j = 0$ and $j = 92$ inputs at their high levels. The next-to-last row in this table displays the average output w_0 = 3,981,627 and w_{92} = 34,013,832. Combining these averages with Eqs. (4.7), (4.6), and (4.18) gives the estimated group effect of all 92 inputs; namely, $\widehat{\overline{\beta}}_{1-92} = (34{,}013{,}832 - 3{,}983{,}627)/2 = 15{,}016{,}102$. Moreover, the last row in the table combined with Eq. (4.18) gives the standard error of this estimated group effect; namely, $s(\widehat{\overline{\beta}}_{1-92}) = 94{,}029.3/\sqrt{5} = 42{,}051$. So Eq. (4.19) gives $t_4 = 15{,}016{,}102/42{,}051 = 357.09$; i.e., this effect is very significant. In hindsight, these two extreme combinations require fewer replications than $m = 5$; e.g., $m = 2$ replications would have shown that this group effect is important (also see the next exercise).

Exercise 4.2 *Compute the t-statistic using only the first two replications in Table 4.1.*

Given the simulation outputs in Table 4.1 for the first two extreme input combinations, SB continues its search for important inputs. Figure 4.4 shows the successive SB steps; this plot uses the same symbols as Fig. 4.1 does. For example, after the initial step with its two extreme input combinations, SB divides the total group of 92 inputs into two subgroups; namely, the subgroup in the left-hand side of the plot that aggregates all the 79 "decision" inputs, and the other subgroup that aggregates all 13 "environmental" inputs (controllable and environmental inputs will be discussed in Sect. 6.4). We expect that simulation of this (less extreme) combination gives an average output between the average outputs of the preceding two extreme combinations; these values are not displayed. Comparison of $\overline{w}_{79}$ and $\overline{w}_0$ gives $\widehat{\overline{\gamma}}_{1-79}$. Similarly, comparison of $\overline{w}_{92}$ and $\overline{w}_{79}$ gives $\widehat{\overline{\gamma}}_{80-92}$.

Replication r	$w_{0;r}$	$w_{92;r}$	$\widehat{\beta}_{(1-92);r}$
1	3,954,024	34,206,800	15,126,388.0
2	3,975,052	33,874,390	14,949,669.0
3	3,991,679	33,775,326	14,891,823.5
4	4,003,475	34,101,251	15,048,888.0
5	3,983,905	34,111,392	15,063,743.5
Average	3,981,627	34,013,832	15,016,102.4
Standard error	18,633	180,780	94,029.3

TABLE 4.1. Five replications for the two extreme combinations in the Ericsson supply-chain model


$w_0 \rightarrow \gamma_{1-92} \leftarrow w_{92}$
$\gamma_{1-79} \leftarrow w_{79} \rightarrow \gamma_{85-92}$
$\gamma_{1-49} \leftarrow w_{49} \rightarrow \gamma_{50-79}$
$\gamma_{80-84} \leftarrow w_{84} \rightarrow \gamma_{85-92}$
$\gamma_{1-32} \leftarrow w_{32} \rightarrow \gamma_{33-49}$
$\gamma_{85-90} \leftarrow w_{90} \rightarrow \gamma_{91-92}$
$\gamma_{33-41} \leftarrow w_{41} \rightarrow \gamma_{42-49}$
$\gamma_{85-86} \leftarrow w_{86} \rightarrow \gamma_{87-90}$
$\gamma_{91} \leftarrow w_{91} \rightarrow \gamma_{92}$
$\gamma_{42-45} \leftarrow w_{45} \rightarrow \gamma_{46-49}$
$\gamma_{85} \leftarrow w_{85} \rightarrow \gamma_{86}$
$\gamma_{87-88} \leftarrow w_{88} \rightarrow \gamma_{89-90}$
*
$\gamma_{42-44} \leftarrow w_{44} \rightarrow \gamma_{45}$
$\gamma_{46-47} \leftarrow w_{47} \rightarrow \gamma_{48-49}$
$\gamma_{89} \leftarrow w_{89} \rightarrow \gamma_{90}$
$\gamma_{42} \leftarrow w_{43} \rightarrow \gamma_{43}$
$\gamma_{46} \leftarrow w_{46} \rightarrow \gamma_{47}$
$\gamma_{48} \leftarrow w_{48} \rightarrow \gamma_{49}$

* Input 87 is a dummy input

FIGURE 4.4. SB steps in Ericsson case study

So, this step splits the total effect $\overline{\overline{\gamma}}_{1-92}$ into its two additive components. This step decreases the upper limit U for any individual effect in the first subgroup and the second subgroup, respectively; see again Fig. 4.3.

SB does not split a subgroup any further when its estimated aggregated first-order effect is not significantly positive. Note that if the estimate were significantly negative in a two-sided t-test, then the assumption of known signs would be rejected. For example, the estimated aggregated first-order effect of inputs 50 through 79 turns out to be negative but not significant, so this group is not further split.

The input labeled 87 is a "dummy" input in SB that does not occur in the simulation model itself, so this input is known to have zero effect; also see the discussion leading to Exercise 4.3 below.

In this case study, SB stops after 21 steps. The upper limit, $U(21)$, for the first-order effect of any remaining individual input is then reduced to 87,759; see again Fig. 4.3. The shortlist has 11 inputs; the most important input has label 92 (and is "demand"). We have already pointed out that the efficiency of SB improves if the inputs are labeled from least important to most important; we now conclude that input 92 is indeed the most important input and that no input labelled smaller than 43 is declared to be important. This plot also shows that the most important individual input (namely, input 92) has already been identified and estimated after only ten steps; the next important input (input 49) is identified after 16 observations.

More details are given in Kleijnen et al. (2006). These details include the programming and validation of the simulation model, steady-state analysis including estimation of a warm-up period, and the role of two-factor interactions and dummy inputs; see the next exercise.

Exercise 4.3 *The Ericsson model concerns three successive variants of the supply chain such that the oldest variant has more inputs (namely, 92) than the latest variant (which has 78 inputs). Hence, applying SB to the latest variant uses 14 dummy inputs. Will the group effect after simulating the two extreme input combinations for the latest variant be smaller or larger than for the old variant?*

Note: Kleijnen et al. (2006) also discusses the need for *software* that implements sequential screening in simulation experiments. That software should generate an input file, once a particular design type (e.g., SB) has been chosen. Such a file can then be executed sequentially and efficiently, in batch mode; i.e., no human intervention is required while the computer executes the sequential design including rules for selecting the next input combination based on all preceding observations. Good computer programming avoids fixing the inputs at specific numerical values within the code; instead, the computer reads input values so that the program can be run for many combinations of these values. Of course, the computer should check whether these values are admissible; i.e., are these combinations within the experimental domain? Such a practice can automatically provide a long list of potential inputs.

4.5 SB for Random Simulations and Variable Number of Replications

Whereas the t-statistic in Eq. (4.19) assumes a fixed m (number of replications) and a "favorite" null-hypothesis (namely, H_0) in Eq. (4.20), the SPRT in Wan et al. (2010) selects the number of replications such that it improves the control over the type-I or α error rate ("false positive") and has no favorite null-hypothesis but considers two comparable hypotheses—as we shall see in this section.

Note: The origin of the SPRT is Wald (1945). False positives are also discussed in Draguljiċ et al. (2014) and Shih et al. (2014).

Ankenman et al. (2015) derives an alternative for the SPRT in Wan et al. (2010), and shows how to save on simulation observations. That publication also considers situations with the dispersion instead of the mean response as "the" output of interest.

To define this SPRT, we use the symbols in Wan et al. (2010) as much as possible. This SPRT is meant to control the type-I error probability through the whole procedure and to hold the type-II or β error probability at each step. We claim that ideally SB should also control the type II error probability over the whole procedure with its sequence of steps (also see De and Baron (2015) and Döhler (2014) for interesting discussions of so-called familywise error probabilities). We therefore consider SB to be no more than a heuristic, but this heuristic is better than apriori assuming that the majority of potentially important individual inputs are unimportant, and experimenting with a small group of inputs that are subjectively assumed to be important. The numerical results for this heuristic—published in Wan et al. (2010) and Shi et al. (2014a)—look very promising.

Wan et al. (2010) assumes a second-order polynomial metamodel (such a metamodel requires mirror combinations; fewer combinations would be required when assuming a first-order polynomial metamodel; however, a second-order polynomial may require fewer replications per combination). Simulation outputs are assumed to have (marginal) Gaussian distributions. Variances may vary with the input combination $\mathbf{x}$. Moreover, the four input combinations including mirror observations—such as Eq. (4.16)—may use CRN. If CRN are indeed applied, then the number of replications for the four input combinations used to estimate $\beta_{j'-j}$ are equal, before beginning the SPRT (see Wan et al. (2010) near the end of their Section 3.2).

In general, an SPRT adds one replication at a time, and terminates as soon as a conclusion can be reached; also see Kleijnen (1987, pp. 54–55, 108–109). Wan et al. (2010) applies a novel SPRT each time when testing either a group effect (in the early stages) or an individual effect.

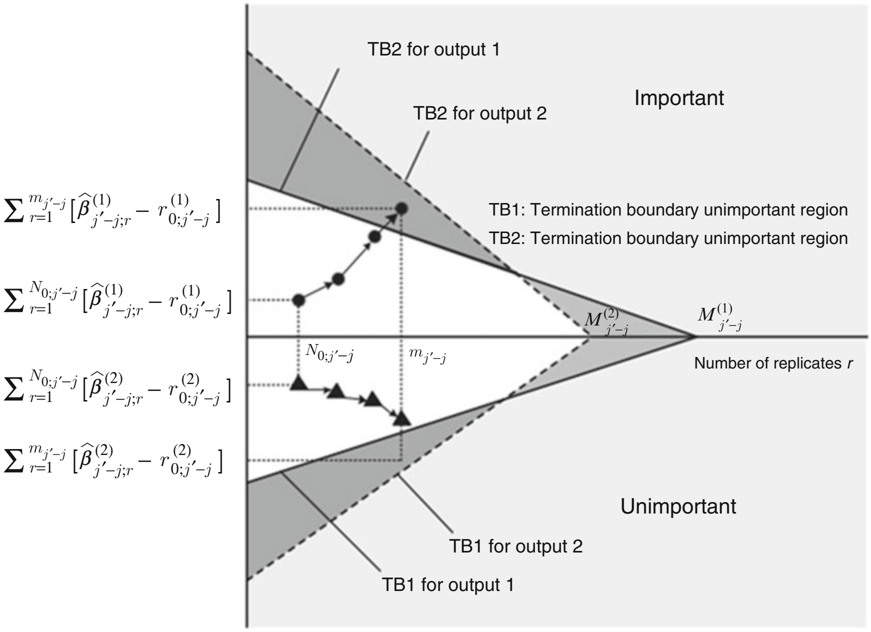

TB2 for output 1
TB2 for output 2
Important
$\sum_{r=1}^{m_{j'-j}} [\widehat{\beta}^{(1)}_{j'-j;r} - r^{(1)}_{0;j'-j}]$
TB1: Termination boundary unimportant region
TB2: Termination boundary unimportant region
$\sum_{r=1}^{N_{0;j'-j}} [\widehat{\beta}^{(1)}_{j'-j;r} - r^{(1)}_{0;j'-j}]$
$M^{(2)}_{j'-j}$
$M^{(1)}_{j'-j}$
$N_{0;j'-j}$
$m_{j'-j}$
Number of replicates r
$\sum_{r=1}^{N_{0;j'-j}} [\widehat{\beta}^{(2)}_{j'-j;r} - r^{(2)}_{0;j'-j}]$
$\sum_{r=1}^{m_{j'-j}} [\widehat{\beta}^{(2)}_{j'-j;r} - r^{(2)}_{0;j'-j}]$
Unimportant
TB1 for output 2
TB1 for output 1

FIGURE 4.5. SPRT example

Wan et al. (2010) classifies inputs with $\beta_j \leq \Delta_0$ as *unimportant* and inputs with $\beta_j \geq \Delta_1$ as *important* where Δ_0 and Δ_1 are determined by the users (Δ was determined analogously, in Sect. 4.2). For these unimportant inputs, the type-I error probability is controlled such that it does not exceed α; for important inputs, the statistical power of the test should be at least γ (the symbol γ used in Wan et al. should not be confused with γ in the preceding sections, where γ denotes the effects of the original, nonstandardized inputs). For *intermediate* inputs—which have $\Delta_0 < \beta_j < \Delta_1$—the power should be "reasonable"; also see Table 4.2 below.

The *initial* number of replications when estimating $\beta_{j'-j}$ is $N_{0;j'-j}$. Wan et al. (2010) selects a value for $N_{0;j'-j}$ that remains constant over all the SB stages; e.g., $N_{0;j'-j} = N_0 = 25$. We, however, expect that $N_{0;j'-j}$ may be smaller in the early stages, because those stages estimate the sum of the positive first-order effects of bigger groups so the signal-noise ratio is larger; see again Exercise 4.2. These $N_{0;j'-j}$ replications are used to estimate the variance $S^2_{j'-j}$ of $\widehat{\beta}_{(j'-j);r}$ (where $\widehat{\beta}_{(j'-j);r}$ denotes the estimator of $\beta_{j'-j}$ computed in replication r with $r = 1, \ldots, N_{0;j'-j}$):

$$S_{j'-j} = \frac{\sum_{r=1}^{N_{0;j'-j}} (\widehat{\beta}_{(j'-j);r} - \overline{\widehat{\beta}}_{j'-j})^2}{N_{0;j'-j} - 1} \quad \text{with } \overline{\widehat{\beta}}_{j'-j} = \frac{\sum_{r=1}^{N_{0;j'-j}} \widehat{\beta}_{(j'-j);r}}{N_{0;j'-j}}. \tag{4.21}$$

The remainder of this section gives statistical details on the SPRT derived in Wan et al. (2010). Hence, some readers may wish to skip these details, and proceed to the Monte Carlo experiment with this SPRT in Sect. 4.5.1.

Wan et al. (2010) uses a SPRT based on

$$\sum_{r=1}^{m_{j'-j}} [\widehat{\beta}_{(j'-j);r} - r_{0;j'-j}] \tag{4.22}$$

where $m_{j'-j}$ denotes the *current* number of replications (so $m_{j'-j} \geq N_{0;j'-j}$) and $r_{0;j'-j}$ denotes the so-called *drift parameter* defined below (because we want to stick close to the symbols in Wan et al. (2010), we do not write $[\sum_{r=1}^{m_{j'-j}} \widehat{\beta}_{(j'-j);r}] - r_{0;j'-j}$, where the new place of the square brackets would emphasize that $r_{0;j'-j}$ does not change as r changes). Shi et al. (2014a) illustrates this SPRT for two types of outputs, denoted through the superscripts (1) and (2) in Fig. 4.5 (SB for multiple output types will be detailed in Sect. 4.6). This plot shows that each type of output has its own *continuation region* that turns out to be a triangle; see the solid lines for type 1 and the dotted lines for type 2. For these two outputs, the symbols "●" and "▲" represent the observed values of the SPRT statistic defined in Eq. (4.22) as functions of the number of replications r. The test sequentially checks whether the statistic in Eq. (4.22) crosses a *termination boundary* (TB), defined such that TB1 denotes the TB of the region in which the effect is declared to be unimportant and TB2 is defined analogously for important effects. This SPRT does not go on for ever; i.e., it ends with a *maximum* number of observations that is one more than $M_{j'-j}$ which denotes the value at which the solid or dotted lines cross the horizontal axis r. Actually, the final number of replications for estimating $\beta_{j'-j}$ is $m_{j'-j}$; the plot displays $N_{0;j'-j} < m_{j'-j} < M_{j'-j}$. The two triangular regions are defined by the two slopes $\pm\lambda$ with

$$\pm\lambda = \pm(\Delta_1 - \Delta_0)/4.$$

Notice that the slopes of the triangles increase as Δ_1 increases; consequently, fewer replications are needed when estimating bigger effects. These regions have the two intercepts $\pm a_{j'-j}$ with

$$\pm a_{j'-j} = \pm a_{0;j'-j} S_{j'-j},$$

where $S^2_{j'-j}$ was defined in Eq. (4.21), and the constant $a_{0;j'-j}$ and the drift $r_{0;j'-j}$ in Eq. (4.22) are the solutions of rather complicated equations specified in Wan et al. (2010, Eq. 5 and Eq. 6) and the Matlab code in Appendix C in the Online Supplement.

However, Shi et al. (2014a) corrects an error in this code (adding the SumInt function that is missing in the Online Supplement). Notice that the intercept $a_{j'-j}$ increases as $S^2_{j'-j}$ increases; consequently, more replications are needed if the simulation outputs—and consequently the estimated input effects—have more noise. Further details of the SPRT algorithm are given in Wan et al. (2010, Fig. 3).

4.5.1 Monte Carlo Experiment with SPRT

The advantage of Monte Carlo experiments is that we can guarantee that all SB assumptions are satisfied, whereas we cannot do so in case studies. Moreover, we know which inputs are truly important in Monte Carlo experiments. Finally, experiments with practical simulation models may be very expensive; i.e., a single simulation run may take hours or days, whereas Monte Carlo experiments may generate an output in (micro)seconds (depending on the computer hardware and software).

Wan et al. (2010) uses a Monte Carlo experiment to evaluate SB combined with the SPRT, considering only $k = 10$ inputs, and selecting $\Delta_0 = 2$ and $\Delta_1 = 4$; the noise e in the metamodel is normally distributed with mean zero and a standard deviation equal to $1 + |E(w)|$. Obviously, these rather large standard deviations require many replications (as we shall illustrate in Table 4.2 and Fig. 4.6). Furthermore, Wan et al. (2010) selects for the type-I error probability $\alpha = 0.05$ and for the power $\gamma = 0.90$. In this Monte Carlo experiment with additive noise e, CRN would generate a linear correlation coefficient with value 1; therefore no CRN are used in this Monte Carlo experiment. The values for the two-factor interactions are resampled in the 1,000 macroreplications. The performance measure of the Monte Carlo experiment is $\widehat{\Pr}(\text{DI})$, which denotes the probability of declaring an individual input to be important; this $\widehat{\Pr}(\text{DI})$ is estimated from the 1,000 macroreplications.

Wan et al. (2010) examines several cases, including the following case. There are five first-order effects between Δ_0 and Δ_1 (remember that $\Delta_0 = 2$ and $\Delta_1 = 4$) and five first-order effects exceeding Δ_1 but not exceeding the value 6. More specifically, these effects have the values 2.00, 2.44, 2.88, 3.32, 3.76, 4.20, 4.64, 5.08, 5.52, and 6.00; see the columns 1 and 2 in Table 4.2. Shi et al. (2014a) presents results that we reproduce in Table 4.2 with a fixed initial sample size N_0 equal to either 5 or 25 and (see the last four columns) N_0 that is either 5 or 25 in the first stage and either 25 % or 50 % of the final number of replications in the immediately preceding stage. This table shows that the selection of N_0 does not seriously affect the SB performance quantified through $\widehat{\Pr}(\text{DI})$. This table does show in its last line that a fixed N_0 (columns 3 and 4) requires more replications than a variable N_0; the number of replications are added over all stages. Note that it is well known that in general the selection of the initial sample size in sequential procedures is difficult; maybe there is an "optimal" value for N_0 and maybe $N_0 = 25$ is closer to that optimal value than $N_0 = 5$ is. Our conclusion is that the rule that makes N_0 vary from stage to stage increases the efficiency of SB.

Some details of this sample-size selection are shown in Fig. 4.6. This plot displays initial sample sizes that are not fixed—except in the very first stage where $k = 10$ and $N_0 = 5$—but are 25 % of the final number of replications in the immediately preceding stage. Results are displayed

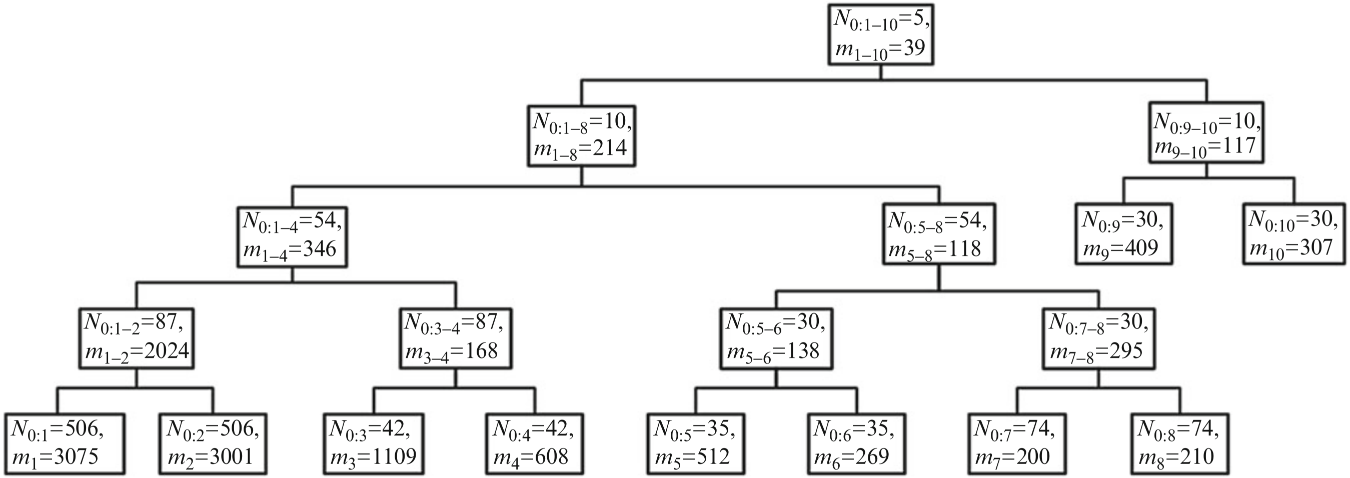


FIGURE 4.6. Initial and final number of replications in macroreplication 1 when $N_{0;1-10}$ is 5 in the first stage and 25 % of m in the preceding stage

TABLE 4.2. $\widehat{\Pr}(\mathrm{DI})$ and number of replications for a constant N_0 and a variable $N_{0;j'-j}$ with $\Delta_0 = 2$ and $\Delta_1 = 4$

Input	Effect β_j	$N_0 = 5$	$N_0 = 25$	(5,25 %)	(5,50 %)	(25,25 %)	(25,50 %)
1	2.00	0.01	0.03	0.04	0.05	0.02	0.03
2	2.44	0.15	0.14	0.17	0.13	0.14	0.12
3	2.88	0.38	0.34	0.38	0.40	0.36	0.38
4	3.32	0.65	0.61	0.69	0.69	0.68	0.57
5	3.76	0.83	0.89	0.84	0.91	0.80	0.76
6	4.20	0.96	0.94	0.96	0.95	0.97	0.96
7	4.64	0.98	0.98	0.97	0.98	0.98	0.99
8	5.08	1.00	1.00	1.00	1.00	1.00	1.00
9	5.52	1.00	1.00	1.00	1.00	1.00	1.00
10	6.00	1.00	1.00	1.00	1.00	1.00	1.00
# replications		21,798	22,008	15,203	14,875	16,860	15,048

for the first macroreplication. So, the initial number in the first stage is $N_{0;1-10} = 5$ and this stage ends with $m_{1-10} = 39$. In the next stage, SB tries to increases its efficiency, so SB splits the total group of 10 inputs into a subgroup with $2^3 = 8$ inputs and a subgroup with the remaining $10 - 8 = 2$ inputs. The initial number of replications in this stage is 25 % of 39, so rounding to the next integer makes this number equal to 10. This stage ends with 214 replications for the first subgroup. We note that the final number of replications tends to increase as the group size decreases so the signal-noise ratio decreases.

Shi et al. (2014a) also studies SB with this SPRT in a Monte Carlo experiment, but this experiment has $k = 100$ inputs, two types of simulation outputs, and several other problem characteristics. Moreover, that article includes a case study concerning a Chinese third-party logistics (TPL) company with a just-in-time (JIT) system for its customer, a car manufacturer. We shall summarize these experiments in the next section.

4.6 Multiresponse SB: MSB

In practice, simulation models have *multiple response types*, which in random simulation may be called *multivariate output*. Only recently Shi et al. (2014a) extended SB to multivariate output, calling the resulting method *multiresponse SB* (MSB). This MSB selects groups of inputs such that within a group all inputs have the same sign for a specific type of output, so no cancellation of first-order effects occurs. MSB also applies the SPRT derived in Wan et al. (2010) to select the number of replications. The performance of MSB is examined through extensive Monte Carlo experiments and a case study concerning a logistic system in China; this performance is very promising, as we shall see.

To define MSB, we use the symbols in Shi et al. (2014a) as much as possible. A basic MSB rule is to declare a group of inputs to be important if that group is important for at least one of the multiple outputs. Moreover, MSB estimates the effects of groups for all outputs, while minimizing the experimental effort compared with SB applied per output type. MSB uses the same *assumptions* as SB does. Specifically, MSB assumes that an adequate metamodel for simulation output l with $l = 1, \ldots, n$ (in this section, n does not denote the number of input combinations) is a second-order polynomial:

$$y^{(l)} = \beta_0^{(l)} + \sum_{j=1}^{k} \beta_j^{(l)} x_j + \sum_{j=1}^{k} \sum_{j' \geq j}^{k} \beta_{j;j'}^{(l)} x_j x_{j'} + e^{(l)} \ (l = 1, \ldots, n). \quad (4.23)$$

Furthermore, MSB assumes that the signs of all first-order effects are known; i.e., it is known that either $\beta_j^{(l)} \geq 0$ or $\beta_j^{(l)} \leq 0$ for given j and l. Finally, MSB assumes the heredity property (like SB does).

By definition, changing the level of input j from $L_j^{(l)}$ to $H_j^{(l)}$ increases output l. This change, however, may decreases output $l' \neq l$. So, $L_j^{(l)}$ equals either $L_j^{(l')}$ or $H_j^{(l')}$; e.g., $L_j^{(l)} = H_j^{(l')}$ if input j has opposite effects on the outputs l and l'. An example is given in Table 4.3. This example has k simulation inputs and $n = 2$ simulation outputs. Columns 4 and 5 show that in example (a) the inputs 1 through k_1 have the same signs, whereas inputs $k_1 + 1$ through k have opposite signs; i.e., changing from $L_j^{(l)}$ to $H_j^{(l)}$ with $l = 1, 2$ and $1 \leq j \leq k_1$ increases both outputs—as the + signs denote—whereas changing from $L_j^{(l)}$ to $H_j^{(l)}$ with $l = 1$ and $k_1 + 1 \leq j \leq k$ increases output 1 but decreases output 2—as the − signs denote). In example (b) we wish to increase output 2 for all k inputs. Therefore, for $1 \leq j \leq k_1$ we have $L_j^{(1)} = L_j^{(2)}$ and $H_j^{(1)} = H_j^{(2)}$, but for $k_1 \leq j \leq k$ we have $L_j^{(1)} = H_j^{(2)}$ and $H_j^{(1)} = L_j^{(2)}$.

The remainder of this section gives statistical details on MSB, so some readers may wish to skip these details, and proceed to the Monte Carlo experiment with MSB and SB in Sect. 4.6.1. First we need some extra

TABLE 4.3. Input values for two output types, in examples (a) and (b)

	(a) Input values for $w^{(1)}$				(b) Input values for $w^{(2)}$			
Input	Low level for $w^{(1)}$	High level for $w^{(1)}$	$w^{(1)}$	$w^{(2)}$	Low level for $w^{(2)}$	High level for $w^{(2)}$	$w^{(1)}$	$w^{(2)}$
1	$L_1^{(1)}$	$H_1^{(1)}$	+	+	$L_1^{(2)}$	$H_1^{(2)}$	+	+
2	$L_2^{(1)}$	$H_2^{(1)}$	+	+	$L_2^{(2)}$	$H_2^{(2)}$	+	+
...	...	...	+	+	...	...	+	+
k_1	$L_{k_1}^{(1)}$	$H_{k_1}^{(1)}$	+	+	$L_{k_1}^{(2)}$	$H_{k_1}^{(2)}$	+	+
$k_1 + 1$	$L_{k_1+1}^{(1)}$	$H_{k_1+1}^{(1)}$	+	−	$L_{k_1+1}^{(2)}$	$H_{k_1+1}^{(2)}$	−	+
...	...	...	...	...	...	...	...	...
k	$L_k^{(1)}$	$H_k^{(1)}$	+	−	$L_k^{(2)}$	$H_k^{(2)}$	−	+

symbols for MSB compared with SB. We let $w_j^{(l)}$ denote output l when inputs 1 though j are at their high levels $H^{(l)}$ and the remaining inputs ($j+1$ through k) are at their low levels $L^{(l)}$. Analogously, we let $w_{-j}^{(l)}$ denote output l when inputs 1 though j are at $L^{(l)}$ and the remaining inputs ($j+1$ through k) are at $H^{(l)}$. We let the symbol "$l \rightarrow l'$" in a superscript denote that output l' is observed "for free"; i.e., running the simulation model to observe output l also gives an observation on the other output l'. For example, $w_k^{(1\rightarrow 2)}$ denotes output 2 when all k inputs are at $H_j^{(1)}$; analogously, $w_{-k}^{(1\rightarrow 2)}$ denotes output 2 when all k inputs are at $L_j^{(1)}$ ($j = 1, \ldots, k$). Therefore, $w_k^{(1\rightarrow 2)}$ and $w_k^{(1)}$ are observed for the same input combination $H_{1-k}^{(1)}$. This gives the following definitions: $w_j^{(l\rightarrow l')}$ denotes output l' when inputs 1 through j are at $H^{(l)}$ and the remaining inputs are at $L^{(l)}$; likewise, the mirror output $w_{-j}^{(l\rightarrow l')}$ denotes output l' when inputs 1 through j are at $L^{(l)}$ and the remaining inputs are at $H^{(l)}$.

Next we define an *input group* as a group of inputs with no cancellation of individual effects within the group. Table 4.3 included example (a), which has two groups with group 1 containing inputs 1 through k_1 so both outputs increase and group 2 containing inputs $k_1 + 1$ through k so output 1 increases and output 2 decreases. In general, an input group has either all n output types increase or all n output types decrease when changing all the individual inputs in this group from -1 to 1.

Shi et al. (2014a) proves that the estimators of group effects and individual effects are analogous to Eq. (4.16). If inputs j' through j are in the same group and they have the *same* signs for outputs l and l', then unbiased group estimators of the first-order group effects for outputs l and l' are

$$\widehat{\beta}_{j'-j}^{(l)} = \frac{[w_j^{(l)} - w_{-j}^{(l)}] - [w_{j'-1}^{(l)} - w_{-(j'-1)}^{(l)}]}{4} \tag{4.24}$$

and

$$\widehat{\beta}_{j'-j}^{(l')} = \frac{[w_j^{(l\rightarrow l')} - w_{-j}^{(l\rightarrow l')}] - [w_{j'-1}^{(l\rightarrow l')} - w_{-(j'-1)}^{(l\rightarrow l')}]}{4} \tag{4.25}$$

where $j' \leq j$ and corresponding terms in Eqs. (4.24) and (4.25) are observed for the same input combination. Obviously, for the individual effects the subscript $j' - j$ is replaced by j. If the inputs j' through j are in the same group, but they have *opposite* signs for outputs l and l', then

$$\widehat{\beta}_{j'-j}^{(l)} = \frac{[w_j^{(l)} - w_{-j}^{(l)}] - [w_{j'-1}^{(l)} - w_{-(j'-1)}^{(l)}]}{4} \tag{4.26}$$

and

$$\widehat{\beta}_{j'-j}^{(l')} = -\frac{[w_j^{(l\rightarrow l')} - w_{-j}^{(l\rightarrow l')}] - [w_{j'-1}^{(l\rightarrow l')} - w_{-(j'-1)}^{(l\rightarrow l')}]}{4} \tag{4.27}$$

where the last equation has a minus sign immediately after the equality sign.

To select the number of replications, MSB uses the SPRT discussed in Sect. 4.5 for SB. In MSB the symbols obviously need the superscript (l), so the symbols become $\Delta_0^{(l)}$, $\Delta_1^{(l)}$, $[S_{j'-j}^{(l)}]^2$, $r_{0;j'-j}^{(l)}$, $a_{j'-j}^{(l)}$, $\lambda^{(l)}$, etc. MSB uses *Bonferroni's inequality* so α is replaced by α/n and $(1-\gamma)$ by $(1-\gamma)/n$. This change implies bigger triangles in Fig. 4.5, in which sampling is continued until either the group of inputs is declared unimportant for all output types or the group is declared important for one or more output types.

Note: MSB is conservative, because it controls the type-I and type-II error rates through Bonferroni's inequality and ignores information on correlations among simulation outputs. Wang and Wan (2014) details two procedures based on a SPRT, assuming a first-order polynomial metamodel estimated through a fractional factorial design and assuming a multivariate normal distribution for the simulation outputs. These two procedures require the solution of (classic) quadratic programming problems, and perform better than MSB.

4.6.1 Monte Carlo Experiments with MSB and SB

Shi et al. (2014a) studies problems with $n = 2$ outputs and $k = 100$ inputs; these inputs can be arranged into two "input groups" (these groups were

TABLE 4.4. Monte Carlo experiments ($i = 1, \ldots, 16$) with four problem characteristics, and resulting number of replications in MSB versus SB

Combinations			Replications	
i	σ	Other three characteristics[a]	MSB	SB
1	5	Inputs (1,2,99,100) = $(2, 5, \pm2, \pm5)$	135	242
2	5	Inputs (1,10,91,100) = $(2, 5, \pm2, \pm5)$	313	600
3	5	Inputs (1,2,99,100) = $(5, 5, \pm2, \pm5)$	119	218
4	5	Inputs (1,10,91,100) = $(2, 5, \pm2, \pm5)$	233	463
5	10	Inputs (1,2,99,100) = $(2, 5, \pm2, \pm5)$	350	656
6	10	Inputs (1,10,91,100) = $(2, 5, \pm2, \pm5)$	933	1,607
7	10	Inputs (1,2,99,100) = $(5, 5, \pm2, \pm5)$	250	470
8	10	Inputs (1,10,91,100) = $(2, 5, \pm2, \pm5)$	641	1,112
9	5	Inputs (1,2,3,4,97,98,99,100) = (2,3,4,5,±2,±3,±4,±5)	178	354
10	5	Inputs (1,10,20,30,71,81,91,100) = (2,3,4,5,±2,±3,±4,±5)	536	1,058
11	5	Inputs (1,2,3,4,97,98,99,100) = (5,5,5,5,±5,±5,±5,±5)	145	290
12	5	Inputs (1,10,20,30,71,81,91,100) = (5,5,5,5,±5,±5,±5,±5)	410	818
13	10	Inputs (1,2,3,4,97,98,99,100) = (2,3,4,5,±2,±3,±4,±5)	464	922
14	10	Inputs (1,10,20,30,71,81,91,100) = (2,3,4,5,±2,±3,±4,±5)	1,713	3,233
15	10	Inputs (1,2,3,4,97,98,99,100) = (5,5,5,5,±5,±5,±5,±5)	319	620
16	10	Inputs (1,10,20,30,71,81,91,100)=(5,5,5,5,±5,±5,±5,±5)	1,126	2,248

[a] Symbol "+" means positive effect on output 1; i.e., $\beta^{(1)} > 0$
Symbol "–" means negative effect on output 2; i.e., $\beta^{(2)} < 0$

defined in the discussion of Table 4.3). MSB is compared with SB when SB is applied per output type. The same thresholds are selected as in Wan et al. (2010); namely, $\Delta_0^{(l)} = 2$ and $\Delta_1^{(l)} = 4$ $(l = 1, 2)$. The initial number of replications in the first stage is $N_{0;1-100} = 5$, and the initial number of replications in the next stages are 25 % of the final number of replications in the immediately preceding stage (again see Fig. 4.6 with $k = 10$). The prespecified type-I error rate is $\alpha = 0.05$ and the prespecified power is $\gamma = 0.9$. Because there are two output types, application of Bonferroni's inequality replaces α by $\alpha/2 = 0.025$ and $1 - \gamma$ by $(1 - \gamma)/2 = 0.05$. The following four *problem characteristics*—with each characteristic having two levels (values)—are studied; also see Table 4.4 (the last two columns will be discussed below).

- *Sparsity* of effects; i.e., either four or eight of the hundred first-order effects are "important". Actually, rows 1 through 8 of Table 4.4 display four important effects, and rows 9 through 16 display eight important effects.

- *Signal-noise ratio*; the higher the noise is, the more replications should be obtained. The standard deviation σ of e_l is either five or ten; see column 2.

- *Variability* of effects; i.e., either all important first-order effects have the same value $|\beta_j^{(l)}| = 5$ (see rows 11, 12, 15, 16) or all these effects are different; namely, -5, -2, 2, 5 (so $|\beta_j^{(l)}|$ is either two or five) when there are four important inputs (see characteristic 1), and $|\beta_j^{(l)}| = 2$, 3, 4, 5 when there are eight important inputs.

- *Clustering* of effects; the more clustered the individual important effects are, the more efficient SB and MSB are expected to be. When there are four important inputs and they are clustered, then the important inputs are 1, 2, 99, and 100 (see, e.g., row 1), and the nonclustered inputs are 1, 10, 91, and 100 (see, e.g., row 2); when there are eight important inputs, then the clustered inputs are 1, 2, 3, 4, 97, 98, 99, 100, and the nonclustered inputs are 1, 10, 20, 30, 71, 81, 91, and 100.

Experimenting with two levels per characteristic gives 16 combinations; see the first three columns of Table 4.4. As footnote [a] at the bottom of the table shows, in all these 16 combinations there are two inputs groups: the ± signs mean that all important first-order effects are positive for output 1 and some important first-order effects are negative for output 2; e.g., in combination 1 the four important inputs 1, 2, 99, and 100 have positive effects for output 1 but the inputs 99 and 100 have negative effects for output 2.

These Monte Carlo experiments use 1,000 macroreplications. The last two columns of Table 4.4 display the *average number of replications* per stage, for MSB and SB; these columns quantify the *efficiency*. Both MSB and SB require more replications for a higher number of important inputs, more variability of effects, more noise of outputs, and lack of clustering of important inputs. For example, combination 3—with $\sigma = 5$, only four important inputs and much clustering—requires the minimum number of replications; namely, 119 for MSB. However, combination 14—with $\sigma = 10$, eight important inputs with different values and even spread—requires the maximum number of replications; namely, 1,723 for MSB. In general, MSB requires only approximately half the number of replications needed by SB; our explanation is that input combinations used to estimate effects for one output in MSB are also used for the other output. Altogether, MSB requires fewer input combinations and replications than SB does.

Efficacy is quantified through $\widehat{\Pr}(\mathrm{DI})$, which was also used in Table 4.2. Figure 4.7 displays $\widehat{\Pr}(\mathrm{DI})$ only for the four combinations numbered 2, 7, 9, and 14, because similar results are found for the remaining 12 combinations. The x-axis of this plot gives $|\beta_j^{(l)}|$ and the y-axis gives $\widehat{\Pr}(\mathrm{DI})$; e.g., $|\beta_j^{(l)}| = 0$, 2, and 5 in combination 2, and $|\beta_j^{(l)}| = 0$, 2, 3, 4, and 5 in combination 9. Because $|\beta_j^{(1)}| = |\beta_j^{(2)}|$, we display $\widehat{\Pr}(\mathrm{DI})$ for output 1 only. This plot suggests that $|\beta_j^{(l)}|$ has an important positive effect on $\widehat{\Pr}(\mathrm{DI})$; we may indeed expect that the power of the method for screening inputs increases as the input has a higher effect. In all combinations, $\widehat{\Pr}(\mathrm{DI}) = 0$ when $|\beta_j^{(l)}| = 0$ and $\widehat{\Pr}(\mathrm{DI}) = 1$ when $|\beta_j^{(l)}| = 5$; $\widehat{\Pr}(\mathrm{DI})$ lies in the interval $[0.025, 0.95]$ when $\Delta_0^{(l)} = 2 \leq |\beta_j^{(l)}| \leq \Delta_1^{(l)} = 4$; in combination 9 the type-I error rate is virtually the same for MSB and SB, and in combination 14 this rate is not significantly higher than the rate for SB. So, both MSB and SB give appropriate results for their type-I error rates and power. However, $\widehat{\Pr}(\mathrm{DI})$ in MSB exceeds $\widehat{\Pr}(\mathrm{DI})$ in SB when $\Delta_0^{(l)} \leq |\beta_j^{(l)}| \leq \Delta_1^{(l)}$; e.g., in combination 9 (south-west corner of the plot), $\widehat{\Pr}(\mathrm{DI})$ for MSB is 0.7 and $\widehat{\Pr}(\mathrm{DI})$ for SB is only 0.38 and 0.43 when $|\beta_j^{(l)}| = 3$ (also see combination 14 in the south-east corner). The explanation may be that an input that is unimportant for one output has a chance to be important for the other output, so the probability of declaring this input to be important increases.

Shi et al. (2014a) conducts another experiment in which the two outputs do not have effects with the same magnitudes; i.e., the magnitudes for output 1 are double those for output 2. It is realistic that the thresholds for the two outputs also differ; i.e., $\Delta_0^{(1)} \neq \Delta_0^{(2)}$, and $\Delta_1^{(1)} \neq \Delta_1^{(2)}$. Actually, $\Delta_0^{(1)} = 2\Delta_0^{(2)}$ and $\Delta_1^{(1)} = 2\Delta_1^{(2)}$; i.e., $\Delta_0^{(1)} = 4 \neq \Delta_0^{(2)} = 2$, $\Delta_1^{(1)} = 8 \neq \Delta_1^{(2)} = 4$. The results for this experiments turn out to be very similar to the results for the former experiments. The conclusion of these Monte Carlo experiments is that MSB is more efficient and effective than SB.

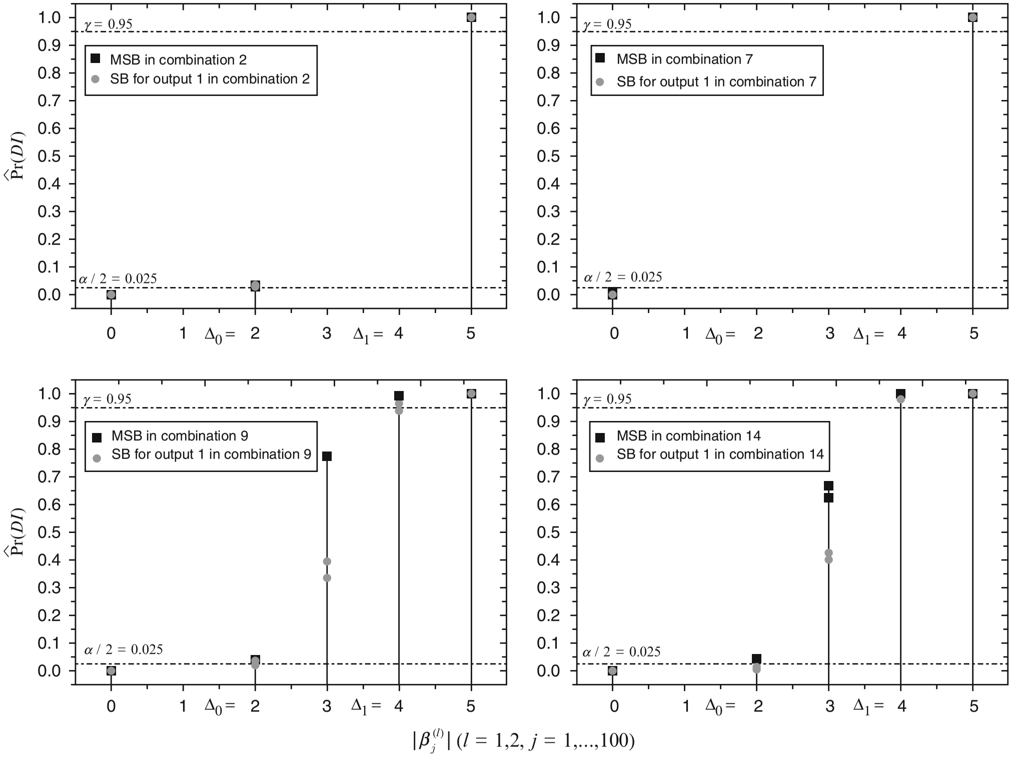


FIGURE 4.7. Estimated probability of declaring an input to be important $\widehat{\Pr}(\text{DI})$ versus magnitude of input effect $|\beta_j^{(l)}|$

4.6.2 Case Study: Chinese Supply-Chain

Shi et al. (2014a) presents a case study concerning a Chinese third-party logistics (TPL) company that wants to improve the just-in-time (JIT) system for its customer, a car manufacturer; more details are given in Shi et al. (2014b). The discrete-event simulation model represents a flow of parts, truck scheduling, etc. The Chinese car market is expected to grow by ten to fifteen percent over the next decade. To satisfy this growing demand, the TPL customer expects to open another assembly plant. When this new plant becomes operational, the current TPL capacity will not meet the logistic needs. Management wants to maintain the current logistic performance, measured through the *average cycle time* (CT) of a part and the *number of throughput* (NT) per "month" or 30-day period. A high CT conflicts with the JIT philosophy. NT is the sum of the shipments collected at the part suppliers and delivered to the assembly plants within a production cycle of 30 days. The goal of this case study is to identify the inputs that are important for one or both outputs (CT, NT).

The simulation model has 26 inputs that may affect CT or NT. These inputs and their low and high values are detailed in Shi et al. (2014a). All inputs are quantitative, except for one input; namely, the queueing discipline. Inputs 1 through 5 are known to have the same signs for both outputs, so two input groups are formed; namely, group 1 with inputs 1 though 5, and group 2 with the remaining inputs labeled 6 through 26. (Shi et al. (2014b) considers only 21 instead of 26 inputs, and uses a fixed number of replications—namely $m = 5$—and applies SB per output.)

The SPRT uses $\Delta_1^{(\mathrm{CT})} = 5$ and $\Delta_1^{(\mathrm{NT})} = 3{,}000$ as the performance improvement not to be missed, and $\Delta_0^{(\mathrm{CT})} = 2.5$ and $\Delta_0^{(\mathrm{NT})} = 2{,}000$ as the minimum critical values. The initial number of replications in the first stage is inspired by Fig. 4.6 (the Monte Carlo experiment); for input group 1 $N_{0;1-5} = 5$ and for group 2 $N_{0;6-26} = 5$ too; the initial number of replications in the next stages is 25 % of the final number of replications in the immediately preceding stage, but not smaller than 5. Because there are two outputs, applying Bonferroni's inequality implies that $\alpha = 0.05$ and $1 - \gamma = 0.1$ are replaced by $\alpha/2 = 0.025$ and $(1 - \gamma)/2 = 0.05$. Figure 4.8 displays the MSB results per stage, where shaded blocks denote individual inputs declared to be important. Altogether, MSB requires 233 replications—namely, $m_{1-5} + m_{6-26} + \ldots + m_{21}$—to identify five important inputs—namely, inputs 4, 5, 14, 17, and 20; the inputs 4 and 5 are in input group 1 (see the first left bifurcation) and inputs 14, 17, and 20 are in input group 2 (see the first right bifurcation).

Shi et al. (2014a) shows that SB requires 238 and 117 replications for CT and NT, respectively. So, altogether SB requires 355 replications, whereas MSB requires only 233 replications. SB and MSB declare the same inputs to be important; SB identifies the inputs 4, 5, 14, 17, and 20 for CT and

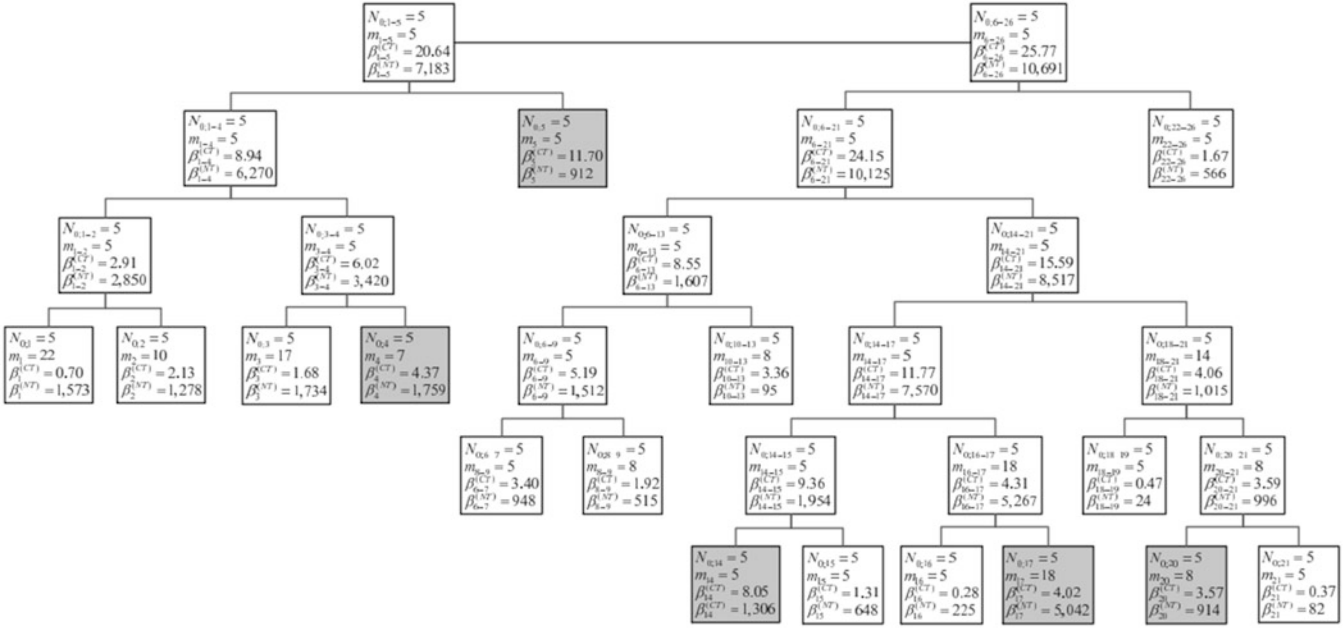

$N_{0;1-5} = 5$, $m_{1-5} = 5$, $\beta_{1-5}^{(CT)} = 20.64$, $\beta_{1-5}^{(NT)} = 7{,}183$
$N_{0;6-26} = 5$, $m_{6-26} = 5$, $\beta_{6-26}^{(CT)} = 25.77$, $\beta_{6-26}^{(NT)} = 10{,}691$
$N_{0;1-4} = 5$, $m_{1-4} = 5$, $\beta_{1-4}^{(CT)} = 8.94$, $\beta_{1-4}^{(NT)} = 6{,}270$
$N_{0;5} = 5$, $m_5 = 5$, $\beta_5^{(CT)} = 11.70$, $\beta_5^{(NT)} = 912$
$N_{0;6-21} = 5$, $m_{6-21} = 5$, $\beta_{6-21}^{(CT)} = 24.15$, $\beta_{6-21}^{(NT)} = 10{,}125$
$N_{0;22-26} = 5$, $m_{22-26} = 5$, $\beta_{22-26}^{(CT)} = 1.67$, $\beta_{22-26}^{(NT)} = 566$
$N_{0;1-2} = 5$, $m_{1-2} = 5$, $\beta_{1-2}^{(CT)} = 2.91$, $\beta_{1-2}^{(NT)} = 2{,}850$
$N_{0;3-4} = 5$, $m_{3-4} = 5$, $\beta_{3-4}^{(CT)} = 6.02$, $\beta_{3-4}^{(NT)} = 3{,}420$
$N_{0;6-13} = 5$, $m_{6-13} = 5$, $\beta_{6-13}^{(CT)} = 8.55$, $\beta_{6-13}^{(NT)} = 1{,}607$
$N_{0;14-21} = 5$, $m_{14-21} = 5$, $\beta_{14-21}^{(CT)} = 15.59$, $\beta_{14-21}^{(NT)} = 8{,}517$
$N_{0;1} = 5$, $m_1 = 22$, $\beta_1^{(CT)} = 0.70$, $\beta_1^{(NT)} = 1{,}573$
$N_{0;2} = 5$, $m_2 = 10$, $\beta_2^{(CT)} = 2.13$, $\beta_2^{(NT)} = 1{,}278$
$N_{0;3} = 5$, $m_3 = 17$, $\beta_3^{(CT)} = 1.68$, $\beta_3^{(NT)} = 1{,}734$
$N_{0;4} = 5$, $m_4 = 7$, $\beta_4^{(CT)} = 4.37$, $\beta_4^{(NT)} = 1{,}759$
$N_{0;6-9} = 5$, $m_{6-9} = 5$, $\beta_{6-9}^{(CT)} = 5.19$, $\beta_{6-9}^{(NT)} = 1{,}512$
$N_{0;10-13} = 5$, $m_{10-13} = 8$, $\beta_{10-13}^{(CT)} = 3.36$, $\beta_{10-13}^{(NT)} = 95$
$N_{0;14-17} = 5$, $m_{14-17} = 5$, $\beta_{14-17}^{(CT)} = 11.77$, $\beta_{14-17}^{(NT)} = 7{,}570$
$N_{0;18-21} = 5$, $m_{18-21} = 14$, $\beta_{18-21}^{(CT)} = 4.06$, $\beta_{18-21}^{(NT)} = 1{,}015$
$N_{0;6\ 7} = 5$, $m_{6-9} = 5$, $\beta_{6-7}^{(CT)} = 3.40$, $\beta_{6-7}^{(NT)} = 948$
$N_{0;8\ 9} = 5$, $m_{8-9} = 8$, $\beta_{8-9}^{(CT)} = 1.92$, $\beta_{8-9}^{(NT)} = 515$
$N_{0;14-15} = 5$, $m_{14-15} = 5$, $\beta_{14-15}^{(CT)} = 9.36$, $\beta_{14-15}^{(NT)} = 1{,}954$
$N_{0;16-17} = 5$, $m_{16-17} = 18$, $\beta_{16-17}^{(CT)} = 4.31$, $\beta_{16-17}^{(NT)} = 5{,}267$
$N_{0;18\ 19} = 5$, $m_{18-19} = 5$, $\beta_{18-19}^{(CT)} = 0.47$, $\beta_{18-19}^{(NT)} = 24$
$N_{0;20\ 21} = 5$, $m_{20-21} = 8$, $\beta_{20-21}^{(CT)} = 3.59$, $\beta_{20-21}^{(NT)} = 996$
$N_{0;14} = 5$, $m_{14} = 5$, $\beta_{14}^{(CT)} = 8.05$, $\beta_{14}^{(CT)} = 1{,}306$
$N_{0;15} = 5$, $m_{15} = 5$, $\beta_{15}^{(CT)} = 1.31$, $\beta_{15}^{(NT)} = 648$
$N_{0;16} = 5$, $m_{16} = 5$, $\beta_{16}^{(CT)} = 0.28$, $\beta_{16}^{(NT)} = 225$
$N_{0;17} = 5$, $m_{17} = 18$, $\beta_{17}^{(CT)} = 4.02$, $\beta_{17}^{(NT)} = 5{,}042$
$N_{0;20} = 5$, $m_{20} = 8$, $\beta_{20}^{(CT)} = 3.57$, $\beta_{20}^{(NT)} = 914$
$N_{0;21} = 5$, $m_{21} = 5$, $\beta_{21}^{(CT)} = 0.37$, $\beta_{21}^{(NT)} = 82$

FIGURE 4.8. MSB for Chinese case study

input 17 for NT. It turns out that MSB and SB do not use the same input combinations in every stage.

We emphasize that Fig. 4.8 implies that MSB requires fewer combinations than there are inputs, so MSB is supersaturated. Moreover, the number of replications increases as the group size decreases, so selecting a fixed number of replications may give misleading results.

4.7 Validating the SB and MSB Assumptions

We now present a method for the validation of the assumptions that are the basis of SB and MSB. By definition, "screening" means that the number of inputs k is too big to enable the estimation of all the individual effects of a second-order polynomial. Actually, this number of effects is $1 + k + k + k(k-1)/2$; e.g., the case study in Sect. 4.6.2 is a relatively small screening example with $k = 26$ inputs, and yet the number of effects is 378. To validate the screening results for SB in random simulation, Wan et al. (2010) uses a central composite design (CCD) based on a R-V design for all k inputs. Our method, however, is inspired by Shi and Kleijnen (2015), and is more efficient and—we expect—is still effective. We focus on random simulation, but conjecture that our approach may easily be adapted to deterministic simulation.

The three assumptions of SB and MSB are:

1. a *second-order polynomial* per output is an adequate approximation (a valid metamodel) of the implicit I/O function of the underlying simulation model;

2. the *signs* of the first-order effects are known (so the first-order polynomial approximation per output is monotonic);

3. *heredity* applies; i.e., if an input has no important first-order effect, then this input has no important second-order effects.

We denote the number of *unimportant inputs* identified through SB or MSB by k_{U} where the subscript "U"stands for unimportant; likewise, we denote the number of important inputs by k_{I} where "I"stands for important (obviously $k_{\mathrm{U}} + k_{\mathrm{I}} = k$). Each of the k_{U} unimportant inputs has nearly the same magnitude for its estimated first-order effect for output type l; namely virtually zero; see the threshold $\Delta_0^{(l)}$ $(l = 1, \ldots, n)$ in the SPRT discussed in Sect. 4.5. So we do not need to estimate the many individual—first order and second order—effects of the unimportant inputs; it suffices to test that these k_{U} inputs have virtually no effects. We therefore test the effects of the k_{U} unimportant inputs through the simulation of only a few *extreme* combinations of these inputs. First we explain our method for a simulation model with a single output type ($n = 1$) so SB suffices; next we explain this method for MSB with two output types (as in the Chinese case study).

In SB with a single output type, we simulate only the following two extreme combinations of the unimportant inputs:

(a) All k_{U} inputs declared to be unimportant are at their *low* levels (coded -1), while we fix (freeze) all inputs declared to be important; e.g., we fix these important inputs at their base levels (so the important "factors" become "constants").

(b) All these k_{U} inputs are at their *high* levels (coded 1), while we fix the k_{I} remaining inputs at the same values as in combination (a).

To simplify our explanation, we assume that these k_{I} inputs are *quantitative* and are fixed at their coded values 0. Furthermore, we relabel the k inputs such that the first k_{U} inputs are declared to be unimportant. Finally, we let $\mathbf{x}_{\mathrm{U}}$ denote the k_{U}-dimensional vector with the values of the unimportant inputs, and we let $\mathbf{1}$ denote the k_{U}-dimensional vector with all elements equal to 1. Consequently, the second-order polynomial metamodel in Eq. (4.10) gives the following results for combinations (a) and (b), respectively:

$$E(y \mid \mathbf{x}_{\mathrm{U}} = -\mathbf{1}) = \beta_0 - \sum_{j=1}^{k_{\mathrm{U}}} \beta_j + \sum_{j=1}^{k_{\mathrm{U}}} \sum_{j'=j}^{k_{\mathrm{U}}} \beta_{j;j'}$$

and

$$E(y \mid \mathbf{x}_{\mathrm{U}} = \mathbf{1}) = \beta_0 + \sum_{j=1}^{k_{\mathrm{U}}} \beta_j + \sum_{j=1}^{k_{\mathrm{U}}} \sum_{j'=j}^{k_{\mathrm{U}}} \beta_{j;j'}.$$

These two equations together give

$$E(y \mid \mathbf{x}_{\mathrm{U}} = \mathbf{1}) - E(y|\mathbf{x}_{\mathrm{U}} = -\mathbf{1}) = 2\sum_{j=1}^{k_{\mathrm{U}}} \beta_j \qquad (4.28)$$

where Eq. (4.7) implies $\sum_{j=1}^{k_{\mathrm{U}}} \beta_j = \beta_{1-k_{\mathrm{U}}}$.

We assume that the number of replications for these two combinations is m_{val}. To select a value for m_{val}, we may examine the final number of replications that the SPRT needed to test the significance of individual inputs; see m in the boxes at the bottom of Fig. 4.6. We use CRN, to reduce the noise in our estimator of the difference

$$\delta = E(w|\mathbf{x}_{\mathrm{U}} = \mathbf{1}) - E(w|\mathbf{x}_{\mathrm{U}} = -\mathbf{1})$$

where w denotes the output of the simulation model. This enables us to compute the m_{val} differences between the simulation outputs of the combinations (a) and (b):

$$d_r = w_r(\mathbf{x}_{\mathrm{U}} = \mathbf{1}) - w_r(\mathbf{x}_{\mathrm{U}} = -\mathbf{1}) \ (r = 1, \ldots, m_{\mathrm{val}}).$$

These m_{val} differences give the *t-statistic for paired differences*:

$$t_{m_{\mathrm{val}}-1} = \frac{\overline{d} - E(d)}{s(d)/\sqrt{m_{\mathrm{val}}}} \tag{4.29}$$

where $\overline{d}$ and $s(d)$ are the classic symbols for the sample average and standard deviation of d. This t-statistic gives a CI for the mean difference δ. Finally, we may use this CI to test the following null-hypothesis:

$$H_0 : E(d) \leq \Delta \text{ versus } H_1 : E(d) > \Delta \tag{4.30}$$

where $\leq$ implies that we use a one-sided hypothesis, because Assumption 2 (known signs) implies that the first-order effects are not negative. We reject this H_0 only if the observed value of the t-statistic defined in Eq. (4.29) with $E(d)$ replaced by Δ is "too high"; namely, higher than $t_{m_{\mathrm{val}}-1;1-\alpha}$ where $t_{m_{\mathrm{val}}-1;1-\alpha}$ denotes the $1-\alpha$ quantile (or upper α point) of $t_{m_{\mathrm{val}}-1}$. We propose to select $\Delta = 2k_{\mathrm{U}}\Delta_0$ where Δ_0 was used to define unimportant inputs. So we expect that an individual input is declared unimportant if its effect is Δ_0; together, the k_{U} unimportant inputs might have a total effect of $2k_{\mathrm{U}}\Delta_0$; see the factor 2 in Eq. (4.28). Altogether, we accept bigger differences between the outputs for the extreme input combinations, as the number of unimportant inputs increases; see again Eq. (4.28).

Finally, we test whether the *heredity* assumption indeed holds. This assumption implies that the k_{U} unimportant inputs have no second-order effects $\beta_{j;j'}$ $(j, j' = 1, \ldots, k_{\mathrm{U}})$. Our test of the two extreme combinations (a) and (b), is completely insensitive to these $\beta_{j;j'}$; i.e., even if $\beta_{j;j'} \neq 0$, then these $\beta_{j;j'}$ do not affect this test. Therefore we now consider the *center combination* $\mathbf{x}_0 = \mathbf{0}$ where $\mathbf{0}$ denotes the k_{U}-dimensional vector with all elements equal to zero. Obviously, if the heredity assumption does not apply, then $E(y \mid \mathbf{x}_{\mathrm{U}} = \mathbf{0}) \neq E(y \mid \mathbf{x}_{\mathrm{U}} = -\mathbf{1}) = E(y \mid \mathbf{x}_{\mathrm{U}} = \mathbf{1})$. To test this assumption we assume that the number of replications for the central combination equals m_{val}; we used the same m_{val} for the two extreme combinations above. We again use the CRN that are also used for these two extreme combinations. This gives the following difference:

$$\delta_0 = E(w \mid \mathbf{x}_{\mathrm{U}} = \mathbf{0}) - [\frac{E(w \mid \mathbf{x}_{\mathrm{U}} = \mathbf{1}) + E(w \mid \mathbf{x}_{\mathrm{U}} = -\mathbf{1})}{2}]$$

We observe that—whatever the magnitudes and signs of the first-order effects are—if the second-order polynomial for the k_{U} unimportant inputs holds, then $\delta_0 = -\sum_{j=1}^{k_{\mathrm{U}}}\sum_{j'=j}^{k_{\mathrm{U}}}\beta_{j;j'}$ where some of the $k_{\mathrm{U}}(k_{\mathrm{U}}-1)/2 + k_{\mathrm{U}}$ second-order effects $\beta_{j;j'}$ may be negative and some may be positive so we do not make any assumptions about the magnitude of this sum. To estimate δ_0, we compute the m_{val} differences

$$d_{0;r} = w_r(\mathbf{x}_{\mathrm{U}} = \mathbf{0}) - [\frac{w_r(\mathbf{x}_{\mathrm{U}} = -\mathbf{1}) + w_r(\mathbf{x}_{\mathrm{U}} = \mathbf{1})}{2}] \; (r = 1, \ldots, m_{\mathrm{val}}).$$

These differences give the analogue of Eq. (4.29):

$$t_{0;m_{\text{val}}-1} = \frac{\overline{d}_0 - E(d_0)}{s(d_0)/\sqrt{m_{\text{val}}}}. \tag{4.31}$$

We use this t-statistic to test

$$H_0 : E(d_0) = 0 \text{ versus } H_1 : E(d) \neq 0 \tag{4.32}$$

where we now use a two-sided hypothesis, because the second-order effects may be negative or positive. We reject this H_0 if H_0 gives $|t_{0;m_{\text{val}}-1}| > t_{m_{\text{val}}-1;1-\alpha/2}$. If we wish to preserve the experimentwise type-I error rate, then we apply Bonferroni's inequality and replace α by $\alpha/2$ because we test two null-hypotheses; see Eqs. (4.30) and (4.32).

Now we explain our method for MSB in the case of $n = 2$ output types ($n = 2$ holds in the Chinese case-study). If there were a single input group ($q = 1$), then our method would be the same as the method for the SB explained in the preceding paragraph. If we suppose that there are $q = 2$ input groups, then we simulate the two extreme combinations for each of the $n = 2$ output types; i.e., we simulate the following four combinations:

(a) All k_{U} unimportant inputs are at their *low* levels for output type $l = 1$, while we fix the k_{I} important inputs (e.g., at their base levels).

(b) All k_{U} unimportant inputs are at their *high* levels for output type $l = 1$, while we still fix the k_{I} important inputs at the same values as in combination (a).

(c) All k_{U} unimportant inputs are at their *low* levels for output type $l = 2$, while we fix the k_{I} important inputs at the same values as in combination (a).

(d) All k_{U} unimportant inputs are at their *high* levels for output type $l = 2$, while we fix the k_{I} important inputs at the same values as in combination (a).

We replace H_0 defined in (4.30) for SB by

$$H_0 : \beta^{(l)}_{1-k_{\text{U}}} \leq \Delta^{(l)} \text{ versus } H_1 : \beta^{(l)}_{1-k_{\text{U}}} > \Delta^{(l)} \ (l{=}1,\ldots,\ n). \tag{4.33}$$

Using Bonferroni's inequality, we reject this H_0 if

$$\max_{l} \frac{\overline{d}^{(l)} - \Delta^{(l)}}{s(d^{(l)})/\sqrt{m_{\text{val}}}} > t_{m_{\text{val}}-1;1-\alpha/n} \ (l = 1, \ldots, n); \tag{4.34}$$

we propose to select $\Delta^{(l)} = 2k_{\text{U}}\Delta^{(l)}_0$. In general, our method requires only $2n$ (extreme) combinations; e.g., if $n = 2$, then the method requires the $2n = 4$ combinations labeled (a) through (d) above.

Shi and Kleijnen (2015) also presents a method that takes advantage of the existence of *input groups*. These input groups enable us to estimate the effects of an input group for all n output types *simultaneously*, so we save on simulation effort. The total number of input combinations needed to estimate $\beta^{(l)}_{1-k_{\mathrm{U}}}$ can be proven to be $2q$ with $q \leq n$ so this method may be more efficient than the method we detailed above. For example, the Chinese case-study has $q = n = 2$ so both methods have the same efficiency, but if this case study would involve $n = 3$ output types and still have $q = 2$ input groups, then the method that we detailed is less efficient. Shi and Kleijnen (2015) also presents numerical results for a Monte Carlo experiment and the Chinese case study, investigating various methods for validating the SB and MSB assumptions.

If a validation method suggests that the SB or MSB assumptions do not hold, then we need to look for a different screening method; see again Sect. 4.1.

4.8 Conclusions

In this chapter we started with an overview of different screening designs, including R-III, supersaturated, and group-screening designs. Then we focused on SB. This screening method may be applied to deterministic and random simulation models. We detailed the various assumptions of SB; namely, a first-order or a second-order polynomial metamodel with know signs of the first-order effects, and heredity for the second-order polynomial metamodel. We considered SB for random simulation with either a fixed number of replications or a number selected through a SPRT applied in each stage such that the type-I and the type-II error rates are controlled. We extended SB to MSB for multiple output types, selecting input groups such there is no cancellation of first-order effects for any output type. We concluded with a method for validating the assumptions of SB and MSB.

Monte Carlo experiments—ensuring that all assumptions of SB or MSB are satisfied—suggest that MSB is more efficient (requires fewer observations) than SB, and more effective (gives better control of the two error rates). Various case studies suggest that in practice the SB or MSB assumptions are not too restrictive.

Future research may consider SPRT variants, MSB for more than two output types, and additional case studies. Moreover, future research may compare SB and MSB with the other screening methods mentioned in the introduction of this chapter.

Solutions of Exercises

1. The mirror combination of the extreme combination with all k inputs at their low levels, is the other extreme combination with all inputs at their high levels. So the very first stage of SB uses only two different observations.

2. $\overline{\widehat{\beta}}_{1-92} = (15{,}126{,}388.0 + 14{,}949{,}669.0)/2 = 15{,}038{,}000$ and $s(\overline{\widehat{\beta}}_{1-92}) = 694{,}610/\sqrt{2} = 491{,}160$ so $t_1 = 15{,}038{,}000/491{,}160 = 30.62$.

3. The group effect of the two extreme combinations for the latest variant of the supply chain is smaller than for the old variant.

References

Ankenman BE, Cheng RCH, Lewis SM (2006) A polytope method for estimating factor main effects efficiently. In: Perrone LF, Wieland FP, Liu J, Lawson BG, Nicol DM, Fujimoto RM (eds) Proceedings of the 2006 Winter Simulation Conference, Monterey, pp 369–375

Ankenman BE, Cheng RCH, Lewis SM (2015) Screening for dispersion effects by sequential bifurcation. ACM Transactions on Modeling and Computer Simulation, 25(1):2:1–2:27

Antoniadis A, Bigot J, Gijbels I (2007) Penalized wavelet monotone regression. Stat Probab Lett 77(16):1608–1621

Archer GEB, Saltelli A, Sobol IM (1997) Sensitivity measures, ANOVA-like techniques and the use of bootstrap. J Stat Comput Simul 58:99–120

Borgonovo E, Plischke E (2015) Sensitivity analysis: a review of recent advances. Eur J Oper Res (in press)

Bettonvil B (1990) Detection of important factors by sequential bifurcation. Ph.D. dissertation, Tilburg University Press, Tilburg

Bettonvil B, Kleijnen JPC (1997) Searching for important inputs in simulation models with many inputs: sequential bifurcation. Eur J Oper Res 96:180–194

Boukouvalas A, Gosling JP, Maruri-Aguilar H (2014) An efficient screening method for computer experiments. Technometrics 56(4):422–431

Cachon GP, Terwiesch C (2006) Matching supply with demand. McGraw-Hill, New York

Campolongo F, Cariboni J, Saltelli A (2007) An effective screening design for sensitivity analysis of large models. Environ Model Softw 22(10):1509–1518

Campolongo F, Kleijnen JPC, Andres T (2000) Screening methods. In: Saltelli A, Chan K, Scott EM (eds) Sensitivity analysis. Wiley, Chichester, pp 65–89

Claeys-Bruno M, Dobrijevic M, Cela R, Phan-Tan-Luu R, Sergent M (2011) Supersaturated designs for computer experiments: comparison of construction methods and new methods of treatment adapted to the high dimensional problem. Chemom Intell Lab Syst 105(2):137–146

De SK, Baron M (2015) Sequential tests controlling generalized familywise error rates. Stat Methodol 23:88–102

De Vos C, Saatkamp HW, Nielen M, Huirne RBM (2006) Sensitivity analysis to evaluate the impact of uncertain factors in a scenario tree model for classical swine fever introduction. Risk Anal 26(5):1311–1322

Döhler S (2014) A sufficient criterion for control of some generalized error rates in multiple testing. Stat Probab Lett 92:114–120

Dorfman R (1943) The detection of defective members of large populations. Ann Math Stat 14(4):436–440

Draguljić D, Woods DC, Dean AM, Lewis SM, Vine AE (2014) Screening strategies in the presence of interactions, including comments and rejoinder. Technometrics 56(1):1–28

Edwards DJ, Mee RW (2011) Supersaturated designs: are our results significant? Comput Stat Data Anal 55(9):2652–2664

Fédou J-M, Rendas M-J (2015) Extending Morris method: identification of the interaction graph using cycle-equitable designs. J Stat Comput Simul 85(7):1398–1419

Frazier PI, Jedynak B, Chen L (2012) Sequential screening: a Bayesian dynamic programming analysis. In: Laroque C, Himmelspach J, Pasupathy R, Rose O, Uhrmacher AM (eds) Proceedings of the 2012 Winter Simulation Conference, Berlin, pp 555–566

Holcomb DC, Montgomery DC, Carlyle WM (2007) The use of supersaturated experiments in turbine engine development. Qual Eng 19:17–27

Huang J, Xu J, Xia Z, Liu L, Zhang Y, Li J, Lan G, Qi Y, Kamon M, Sun X, Li Y (2015) Identification of influential parameters through sensitivity analysis of the TOUGH + Hydrate model using LH-OAT sampling. Mar Pet Geol 65:141–156

Jacoby JE, Harrison S (1962) Multi-variable experimentation and simulation models. Nav Res Logist Q 9(2):121–136

Jones B, Nachtsheim CJ (2015) Blocking schemes for definitive screening designs. Technometrics (in press)

Khare YP, Munoz-Carpena R, Rooney RW, Martinez CJ (2015) A multi-criteria trajectory-based parameter sampling strategy for the screening method of elementary effects. Environmental Modelling and Software, 64:230–239

Kleijnen JPC (1975) Screening designs for poly-factor experimentation. Technometrics 17(4):487–493

Kleijnen JPC (1987) Statistical tools for simulation practitioners. Marcel Dekker, New York

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Kleijnen JPC (2009) Factor screening in simulation experiments: review of sequential bifurcation. In: Alexopoulos C, Goldsman D, Wilson JR (eds) Advancing the frontiers of simulation: a festschrift in honor of George S. Fishman. Springer, New York, pp 169–173

Kleijnen JPC, Bettonvil B, Persson F (2006) Screening for the important inputs in large discrete-event simulation: sequential bifurcation and its applications. In: Dean A, Lewis S (eds) Screening: methods for experimentation in industry, drug discovery, and genetics. Springer, New York, pp 287–307

Koukouvinos C, Massou E, Mylona K, Parpoula C (2011) Analyzing supersaturated designs with entropic measures. J Stat Plan Inference 141:1307–1312

Lim E, Glynn PW (2006) Simulation-based response surface computation in the presence of monotonicity. In: Perrone LF, Wieland FP, Liu J, Lawson BG, Nicol DM, Fujimoto RM (eds) Proceedings of the 2006 Winter Simulation Conference, Monterey, pp 264–271

Martin R, Lazakis I, Barbouchi S, Johanning L (2016) Sensitivity analysis of offshore wind farm operation and maintenance cost and availability. Renewable Energy (in press)

Miller GA (1956) The magical number seven plus or minus two: some limits on our capacity for processing information. Psychol Rev 63:81–97

Moon H, Santner T, Dean A (2010) Two-stage sensitivity-based group screening in computer experiments. Working paper, Department of Statistics, The Ohio State University, Columbus

Morris MD (2006) An overview of group factor screening. In: Dean A, Lewis S (eds) Screening methods for experimentation in industry, drug discovery, and genetics. Springer, New York, pp 191–206

Murray K, Müller S, Turlach BA (2013) Revisiting fitting monotone polynomials to data. Comput Stat 28(5):1989–2005

Oh RPT, Sanchez SM, Lucas TW, Wan H, Nissen ME (2009) Efficient experimental design tools for exploring large simulation models. Comput Math Organ Theory 15(3):237–257

Persson JF, Olhager J (2002) Performance simulation of supply chain designs. Int J Prod Econ 77:231–245

Phoa FKH, Chen R-B, Wang W, Wong WK (2015) Optimizing two-level supersaturated designs using swarm intelligence techniques. Technometrics (in press)

Pujol G (2009) Simplex-based screening designs for estimating metamodels. Reliab Eng Syst Saf 94(7):1156–1160

Rosen SL, Guharay SK (2013) A case study examining the impact of factor screening for neural network metamodels. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 Winter Simulation Conference, Washington, DC, pp 486–496

Sanchez SM, Moeeni F, Sanchez PJ (2006) So many factors, so little time…simulation experiments in the frequency domain. Int J Prod Econ 103(1):149–165

Sanchez SM, Wan H, Lucas TW (2009) A two-phase screening procedure for simulation experiments. ACM Trans Model Comput Simul 19(2):7:1–7:24

Sarkar A, Lin DKJ, Chatterjee K (2009) Probability of correct model identification in supersaturated design. Stat Probab Lett 79:1224–1230

Schonlau M, Welch WJ (2006) Screening the input factors to a computer code via analysis of variance and visualization. In: Dean A, Lewis S (eds) Screening: methods for experimentation in industry, drug discovery, and genetics. Springer, New York, pp 308–327

Schruben LW, Cogliano VJ (1987) An experimental procedure for simulation response surface model identification. Commun ACM 30(8):716–730

Siem AYD, de Klerk E, den Hertog D (2008) Discrete least-norm approximation by nonnegative (trigonometric) polynomials and rational functions. Struct Multidiscip Optim 35(4):327–339

Shen H, Wan H (2009) Controlled sequential factorial design for simulation factor screening. Eur J Oper Res 198:511–519

Shen H, Wan H, Sanchez SM (2010) A hybrid method for simulation factor screening. Nav Res Logist 57(1):45–57

Shi W, Kleijnen JPC (2015) Validating the assumptions in multiresponse sequential bifurcation. Center discussion paper (in preparation)

Shi W, Kleijnen JPC, Liu Z (2014a) Factor screening for simulation with multiple responses: sequential bifurcation. Eur J Oper Res 237(1):136–147

Shi W, Shang J, Liu Z, Zuo X (2014b) Optimal design of the auto parts supply chain for JIT operations: sequential bifurcation factor screening and multi-response surface methodology. Eur J Oper Res 236(2):664–676

Shih DT, Kim SB, Chen VCP, Rosenberger JM, Pilla VL (2014) Efficient computer experiment-based optimization through variable selection. Ann Oper Res 216(1):287–305

Singh P, Ferranti F, Deschrijver D, Couckuyt I, Dhaene T (2014) Classification aided domain reduction for high dimensional optimization. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 Winter Simulation Conference, Savannah, pp 3928–3939

Sohi Z, Noorossanaa R, Niakib STA (2012) New statistic to increase correctness in simulation factor screening using frequency domain method. Commun Stat Theory Methods 41(12):2242–2255

Tan MHY (2015) Monotonic quantile regression with Bernstein polynomials for stochastic simulation. Technometrics, (in press)

Tenne Y, Goh C-K (eds) (2010) Computational intelligence in expensive optimization problems. Springer, Berlin

Tsai P-W, Gilmour SG, Mead R (2007) Three-level main-effects designs exploiting prior information about model uncertainty. J Stat Plan Inference 137(2):619–627

Van der Sluijs JP, Craye M, Funtowicz S, Kloprogge P, Ravetz J, Risbey J (2005) Combining quantitative and qualitative measures of uncertainty in model based environmental assessment: the NUSAP system. Risk Anal 25(2):481–492

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690

Wald A (1945) Sequential tests of statistical hypotheses. Ann Math Stat 16(2):117–220

Wan H, Ankenman BE, Nelson BL (2006) Controlled sequential bifurcation: a new factor-screening method for discrete-event simulation. Oper Res 54(4):743–755

Wan H, Ankenman BE, Nelson BL (2010) Improving the efficiency and efficacy of controlled sequential bifurcation for simulation factor screening. INFORMS J Comput 22(3):482–492

Wang W, Wan H (2014) Sequential procedures for multiple responses factor screening. In: Tolk A, Diallo SD, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 Winter Simulation Conference, Savannah, pp 745–756

Wang L, Xue L (2015) Constrained polynomial spline estimation of monotone additive models. J Stat Plan Inference (in press)

Watson CS (1961) A study of the group screening method. Technometrics 3:371–388

Wu CFJ, Hamada M (2009) Experiments; planning, analysis, and parameter design optimization, 2nd edn. Wiley, New York

Wu J, Meyer MC, Opsomer JD (2015) Penalized isotonic regression. J Stat Plan Inference, 161:12–24

Xing D, Wan H, Zhu Y, Sanchez SM, Kaymal T (2013) Simulation screening experiments using LASSO-optimal supersaturated design and analysis: a maritime operations application. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 Winter Simulation Conference, Washington, DC, pp 497–508

Xiong W, Ding J (2015) Robust procedures for experimental design in group testing considering misclassification. Stat Probab Lett 100:35–41

5

Kriging Metamodels and Their Designs

This chapter is organized as follows. Section 5.1 introduces Kriging, which is also called Gaussian process (GP) or spatial correlation modeling. Section 5.2 details so-called *ordinary Kriging* (OK), including the basic Kriging assumptions and formulas assuming deterministic simulation. Section 5.3 discusses parametric bootstrapping and conditional simulation for estimating the variance of the OK predictor. Section 5.4 discusses *universal Kriging* (UK) in deterministic simulation. Section 5.5 surveys designs for selecting the input combinations that gives input/output data to which Kriging metamodels can be fitted; this section focuses on *Latin hypercube sampling* (LHS) and customized sequential designs. Section 5.6 presents *stochastic Kriging* (SK) for random simulations. Section 5.7 discusses bootstrapping with acceptance/rejection for obtaining Kriging predictors that are monotonic functions of their inputs. Section 5.8 discusses sensitivity analysis of Kriging models through *functional analysis of variance* (FANOVA) using Sobol's indexes. Section 5.9 discusses *risk analysis* (RA) or *uncertainty analysis* (UA). Section 5.10 discusses several remaining issues. Section 5.11 summarizes the major conclusions of this chapter, and suggests topics for future research. The chapter ends with Solutions of exercises, and a long list of references.

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_5

5.1 Introduction

In the preceding three chapters we focussed on linear regression metamodels (surrogates, emulators); namely, low-order polynomials. We fitted those models to the input/output (I/O) data of the—either local or global—experiment with the underlying simulation model; this simulation model may be either deterministic or random. We used these metamodels for the explanation of the simulation model's behavior, and for the prediction of the simulation output for input combinations that were not yet simulated.

In the present chapter, we focus on *Kriging* metamodels. The name Kriging refers to Danie Krige (1919–2013), who was a South African mining engineer. In the 1960s Krige's empirical work in *geostatistics*—see Krige (1951)—was formalized by the French mathematician George Matheron (1930–2000), using GPs—see Matheron (1963).

Note: A standard textbook on Kriging in geostatistics involving "spatial datan" is Cressie (1993); more recent books are Chilès and Delfiner (2012) and Stein (1999).

Kriging was introduced as a metamodel for *deterministic simulation models* or "computer models" in Sacks et al. (1989). Simulation models have k-dimensional input combinations where k is a given positive integer, whereas geostatistics considers only two or three dimensions.

Note: Popular textbooks on Kriging in computer models are Forrester et al. (2008) and Santner et al. (2003). A popular survey article is Simpson et al. (2001).

Kriging for *stochastic (random) simulation models* was briefly discussed in Mitchell and Morris (1992). Next, Van Beers and Kleijnen (2003) details Kriging in such simulation models, simply replacing the deterministic simulation output by the average computed from the replications that are usual in stochastic simulation. Although Kriging has not yet been frequently applied in stochastic simulation, we believe that the track record Kriging achieved in deterministic simulation holds promise for Kriging in stochastic simulation; also see Kleijnen (2014).

Note: Kleijnen (1990) introduced Kriging into the discrete-event simulation community. A popular review article is Kleijnen (2009). The classic discussion of Kriging in stochastic simulation is Ankenman et al. (2010). More references will follow in the next sections of this chapter.

Kriging is also studied in *machine learning*. A popular textbook is Rasmussen and Williams (2006). Web sites on GPs in machine learning are

http://www.gaussianprocess.org/
http://ml.dcs.shef.ac.uk/gpss/
http://www.mlss.cc/.

Besides the Anglo-Saxon literature, there is a vast *French* literature on Kriging, inspired by Matheron's work; see

http://www.gdr-mascotnum.fr/documents.html.

Typically, Kriging models are fitted to data that are obtained for larger experimental areas than the areas used in low-order polynomial regression metamodels; i.e., Kriging models are *global* instead of local. Kriging models are used for prediction. The final goals are sensitivity analysis and risk analysis—as we shall see in this chapter—and optimization—as we shall see in the next chapter; these goals were also discussed in Sect. 1.2.

5.2 Ordinary Kriging (OK) in Deterministic Simulation

In this section we focus on OK, which is the simplest form of *universal Kriging* (UK), as we shall see in Sect. 5.4. OK is popular and successful in practical deterministic simulation, as many publications report.

Note: These publications include Chen et al. (2006), Martin and Simpson (2005), and Sacks et al. (1989). Recently, Mehdad and Kleijnen (2015a) also reports that in practice OK is likely to give better predictors than UK.

In Sect. 5.2.1 we present the basics of OK; in Sect. 5.2.2 we discuss the problems caused by the estimation of the (hyper)parameters of OK.

5.2.1 OK Basics

OK assumes the following metamodel:

$$y(\mathbf{x}) = \mu + M(\mathbf{x}) \quad \text{with } \mathbf{x} \in \mathbb{R}^k \tag{5.1}$$

where μ is the constant mean $E[y(\mathbf{x})]$ in the given k-dimensional experimental area, and $M(\mathbf{x})$ is the additive noise that forms a Gaussian (multivariate normal) stationary process with zero mean. By definition, a *stationary process* has a constant mean, a constant variance, and covariances that depend only on the distance between the input combinations (or "points" in $\mathbb{R}^k$) $\mathbf{x}$ and $\mathbf{x}'$ (stationary processes were also defined in Definition 3.2).

Because different Kriging publications use different symbols for the same variable, we now discuss our symbols. We use $\mathbf{x}$—instead of $\mathbf{d}$—because the Kriging literature uses $\mathbf{x}$ for the combination of inputs—even though the design of experiments (DOE) literature and the preceding chapters use $\mathbf{d}$ for the combination of design variables (or factors); $\mathbf{d}$ determines products such as $x_j x_{j'}$ with $j, j' = 1, \ldots, k$. The constant mean μ in Eq. (5.1) is also denoted by β_0; also see the section on UK (Sect. 5.4). Ankenman et al. (2010) calls $M(\mathbf{x})$ the *extrinsic noise* to distinguish it from the *intrinsic noise* in stochastic simulation. OK assumes that the simulation output is deterministic (say) w. We distinguish between y (metamodel output) and w (simulation model output), whereas most Kriging publications do not distinguish between y and w (we also distinguished between y and w in the preceding chapters on linear regression; an example of our use of y and w

is the predictor formula in Eq. (5.2) below). We try to stick to the symbols used in the preceding chapters; e.g., to denote the number of dimensions we use k (not d, which is used in some Kriging publications), $\mathbf{\Sigma}$ (not $\mathbf{\Gamma}$) to denote a covariance matrix, and σ (not γ or $\mathbf{\Sigma}(\mathbf{x}_0, .)$) to denote a vector with covariances.

OK with its constant mean μ does not imply a *flat* response surface. Actually, OK assumes that $M(\mathbf{x})$ has positive covariances so $\text{cov}[\text{y}(\mathbf{x}), \text{y}(\mathbf{x}')] > 0$. Consequently, if it happens that $y(\mathbf{x}) > \mu$, then $E[y(\mathbf{x}')] > \mu$ is "very likely" (i.e., the probability is greater than 0.50)—especially if $\mathbf{x}$ and $\mathbf{x}'$ lie close in $\mathbb{R}^k$. However, a linear regression metamodel with white noise implies $\text{cov}[y(\mathbf{x}), y(\mathbf{x}')] = 0$; see the definition of white noise that we gave in Definition 2.3.

OK uses a *linear* predictor. So let $\mathbf{w} = (w(\mathbf{x}_1), \ldots, w(\mathbf{x}_n))'$ denote the n observed values of the simulation model at the n so-called *old* points (in machine learning these old points are called the "training set"). OK computes the predictor $\widehat{y}(\mathbf{x}_0)$ for a *new* point $\mathbf{x}_0$ as a linear function of the n observed outputs at the old points:

$$\widehat{y}(\mathbf{x}_0) = \textstyle\sum_{i=1}^{n} \lambda_i w_i = \boldsymbol{\lambda}'\mathbf{w} \tag{5.2}$$

where $w_i = f_{\text{sim}}(\mathbf{x}_i)$ and f_{sim} denotes the mathematical function that is defined by the simulation model itself (also see Eq. (2.6); the weight λ_i decreases with the *distance* between the new input combination $\mathbf{x}_0$ and the old combination $\mathbf{x}_i$, as we shall see in Eq. (5.6); i.e., the weights $\boldsymbol{\lambda}' = (\lambda_1, \ldots, \lambda_n)$ are not constants (whereas $\boldsymbol{\beta}$ in linear regression remains constant). Notice that $\mathbf{x}_i = (x_{i;j})$ $(i = 1, \ldots, n;\ j = 1, \ldots, k)$ so $\mathbf{X}' = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is a $k \times n$ matrix.

To determine the *optimal* values for the weights $\boldsymbol{\lambda}$ in Eq. (5.2), we need to specify a criterion for OK. In fact, OK (like other types of Kriging) uses the *best linear unbiased predictor (BLUP)*, which (by definition) minimizes the mean squared error (MSE) of the predictor:

$$\min\ \text{MSE}[\widehat{y}(\mathbf{x}_0)] = \min\ \{E[\widehat{y}(\mathbf{x}_0) - y(\mathbf{x}_0)\}^2]; \tag{5.3}$$

moreover, the predictor must be *unbiased* so

$$E[\widehat{y}(\mathbf{x}_0)] = E[y(\mathbf{x}_0)]. \tag{5.4}$$

This bias constraint implies that if the new point coincides with one of the old points, then the predictor must be an *exact interpolator*; i.e., $\widehat{y}(\mathbf{x}_i) = w(\mathbf{x}_i)$ with $i = 1, \ldots, n$ (also see Exercise 5.2 below).

Note: Linear regression uses as criterion the sum of squared residuals (SSR), which gives the least squares (LS) estimator. This estimator is not an exact interpolator, unless $n = q$ where q denotes the number of regression parameters; see Sect. 2.2.1.

It can be proven that the solution of the constrained minimization problem defined by Eqs. (5.3) and (5.4) implies that $\boldsymbol{\lambda}$ must satisfy the following condition where $\mathbf{1} = (1, \ldots, 1)'$ is an n-dimensional vector with all elements equal to 1 (a more explicit notation would be $\mathbf{1}_n$):

$$\sum_{i=1}^{n} \lambda_i = \mathbf{1}'\boldsymbol{\lambda} = 1. \tag{5.5}$$

Furthermore, it can be proven that the *optimal* weights are

$$\boldsymbol{\lambda}_o' = \left[\boldsymbol{\sigma}(x_0) + \mathbf{1}\frac{1 - \mathbf{1}'\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}(x_0)}{\mathbf{1}'\boldsymbol{\Sigma}^{-1}\mathbf{1}}\right]' \boldsymbol{\Sigma}^{-1} \tag{5.6}$$

where $\boldsymbol{\Sigma} = (\text{cov}(y_i, y_{i'}))$—with $i, i' = 1, \ldots, n$—denotes the $n \times n$ symmetric and positive definite matrix with the covariances between the metamodel's "old" outputs (i.e., outputs of input combinations that have already been simulated), and $\boldsymbol{\sigma}(x_0) = (\text{cov}(y_i, y_0))$ denotes the n-dimensional vector with the covariances between the metamodel's n "old" outputs y_i and y_0, where y_0 denotes the metamodel's new output. Equation (5.1) implies $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_M$, but we suppress the subscript M until we really need it; see the section on stochastic simulation (Sect. 5.6). Throughout this book, we use Greek letters to denote unknown parameters (such as covariances), and bold upper case letters for matrixes and bold lower case letters for vectors.

Finally, it can be proven (see, e.g., Lin et al. 2004) that Eqs. (5.1), (5.2), and (5.6) together imply

$$\widehat{y}(\mathbf{x}_0) = \mu + \boldsymbol{\sigma}(x_0)'\boldsymbol{\Sigma}^{-1}(\mathbf{w} - \mu\mathbf{1}). \tag{5.7}$$

We point out that this predictor varies with $\boldsymbol{\sigma}(x_0)$; given are the Kriging parameters μ and $\boldsymbol{\Sigma}$—where $\boldsymbol{\Sigma}$ depends on the given old input data $\mathbf{X}$—and the old simulation output $\mathbf{w}(\mathbf{X})$. So we might replace $\widehat{y}(\mathbf{x}_0)$ by $\widehat{y}(\mathbf{x}_0|\mu, \boldsymbol{\Sigma}, \mathbf{X}, \mathbf{w})$ or $\widehat{y}(\mathbf{x}_0|\mu, \boldsymbol{\Sigma}, \mathbf{X})$—because the output $\mathbf{w}$ of a deterministic simulation model is completely determined by $\mathbf{X}$—but we do not use this unwieldy notation.

Exercise 5.1 *Is the conditional expected value of the predictor in Eq. (5.7) smaller, equal, or larger than the unconditional mean μ if that condition is as follows: $w_1 > \mu$, $w_2 = \mu$, $\ldots$, $w_n = \mu$?*

Exercise 5.2 *Use Eq. (5.7) to derive the predictor if the new point is an old point, so $\mathbf{x}_0 = \mathbf{x}_i$.*

The Kriging predictor's *gradient* $\nabla(\widehat{y}) = (\partial\widehat{y}/\partial x_1, \ldots, \partial\widehat{y}/\partial x_k)$ results from Eq. (5.7); details are given in Lophaven et al. (2002, Eq. 2.18). Gradients will be used in Sect. 5.7 and in the next chapter (on simulation optimization). We should not confuse $\nabla(\widehat{y})$ (the gradient of the Kriging metamodel) and $\nabla(w)$, the gradient of the underlying simulation model. Sometimes we can indeed compute $\nabla(w)$ in deterministic simulation (or estimate $\nabla(w)$ in stochastic simulation); we may then use $\nabla(w)$ (or $\widehat{\nabla}(w)$)

to estimate better Kriging metamodels; see Qu and Fu (2014), Razavi et al. (2012), Ulaganathan et al. (2014), and Viana et al. (2014)'s references numbered 52, 53, and 54 (among the 221 references in that article).

If we let τ^2 denote the variance of y—where y was defined in Eq. (5.1)—then the MSE of the optimal predictor $\widehat{y}(\mathbf{x}_0)$—where $\widehat{y}(\mathbf{x}_0)$ was defined in Eq. (5.7)—can be proven to be

$$\begin{aligned} \text{MSE}\,[\widehat{y}(\mathbf{x}_0)] = \tau^2 - \boldsymbol{\sigma}(\mathbf{x}_0)'\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}(\mathbf{x}_0) \\ + \frac{[1-\mathbf{1}'\boldsymbol{\Sigma}^{-1}\boldsymbol{\sigma}(\mathbf{x}_0)]^2}{\mathbf{1}'\boldsymbol{\Sigma}^{-1}\mathbf{1}}. \end{aligned} \tag{5.8}$$

Because the predictor $\widehat{y}(\mathbf{x}_0)$ is unbiased, this MSE equals the predictor variance—which is often called the *Kriging variance*. We denote this variance by σ^2_{OK}, the variance of the OK predictor. Analogously to the comment we made on Eq. (5.7), we now point out that this MSE depends on $\boldsymbol{\sigma}(x_0)$ only because the other factors in Eq. (5.8) are fixed by the old I/O data (we shall use this property when selecting a new point in sequential designs; see Sect. 5.5.2).

Exercise 5.3 *Use Eq. (5.8) to derive that* $\sigma^2_{OK} = 0$ *if* $\mathbf{x}_0$ *equals one of the points already simulated; e.g.,* $\mathbf{x}_0 = \mathbf{x}_1$.

Because σ^2_{OK} is zero if $\mathbf{x}_0$ is an old point, the function $\sigma^2_{OK}(\mathbf{x}_0)$ has many local minima if $n > 1$—and has many local maxima too; i.e., $\sigma^2_{OK}(\mathbf{x}_0)$ is nonconcave. Experimental results of many experiments suggest that $\sigma^2_{OK}(\mathbf{x}_0)$ has local maxima at $\mathbf{x}_0$ approximately halfway between old input combinations $\mathbf{x}_i$; see part c of Fig. 5.2 below. We shall return to this characteristic in Sect. 6.3.1 on "efficient global optimization" (EGO).

Obviously, the optimal weight vector $\boldsymbol{\lambda}_o$ in Eq. (5.6) depends on the covariances—or equivalently the correlations—between the outputs of the Kriging metamodel in Eq. (5.1). Kriging assumes that these correlations are determined by the "distance" between the input combinations. In geostatistics, Kriging often uses the *Euclidean distance* (say) h between the inputs $\mathbf{x}_g$ and $\mathbf{x}_{g'}$ with $g, g' = 0, 1, \ldots, n$ (so g and g' range between 0 and n and consequently $\mathbf{x}_g$ and $\mathbf{x}_{g'}$ cover both the new point and the n old points):

$$h_{g;g'} = \|\mathbf{x}_g - \mathbf{x}_{g'}\|_2 = \sqrt{\textstyle\sum_{j=1}^k (x_{g;j} - x_{g';j})^2} \tag{5.9}$$

where $\|\bullet\|_2$ denotes the L_2 norm. This assumption means that

$$\rho[y(\mathbf{x}_g), y(\mathbf{x}_{g'})] = \frac{\sigma(h_{g;g'})}{\tau^2}, \tag{5.10}$$

which is called an *isotropic* correlation function; see Cressie (1993, pp. 61–62).

In simulation, however, we often assume that the Kriging metamodel has a correlation function—which implies a covariance function—that is

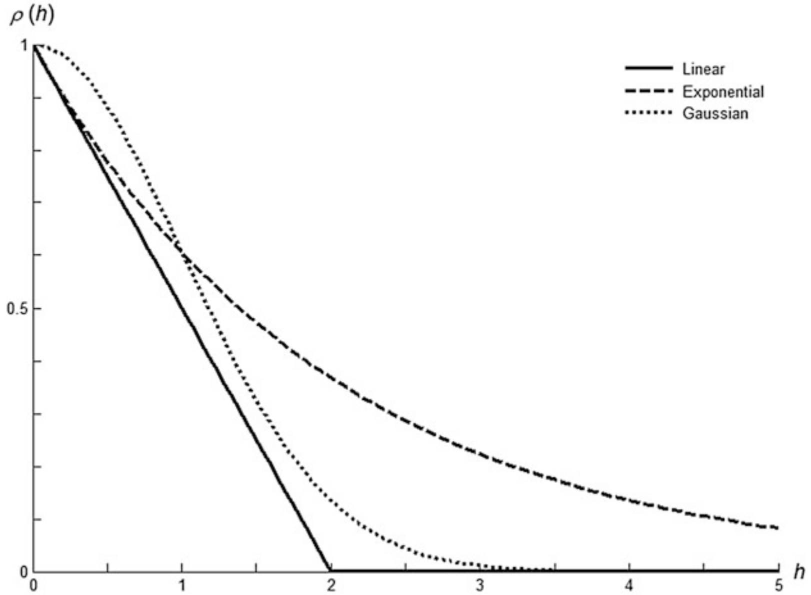


FIGURE 5.1. Three types of correlation functions $\rho(h)$ with distance h and parameter $\theta = 0.5$

not isotropic, but is *anisotropic*; e.g., in a *separable* anisotropic correlation function we replace Eq. (5.10) by the *product* of k one-dimensional correlation functions:

$$\rho[y(\mathbf{x}_g), y(\mathbf{x}_{g'})] = \prod_{j=1}^{k} \rho(x_{g;j}, x_{g';j}) \ (g, g' = 0, 1, \ldots, n). \tag{5.11}$$

Because Kriging assumes a stationary process, the correlations in Eq. (5.11) depend only on the distances in the k dimensions:

$$h_{g;g';j} = |x_{g;j} - x_{g';j}| \quad (j = 1, \ldots, k); \tag{5.12}$$

also see Eq. (5.9). So, $\rho(x_{g;j}, x_{g';j})$ in Eq. (5.11) reduces to $\rho(h_{g;g';j})$. Obviously, if the simulation model has a single input so $k = 1$, then these isotropic and the anisotropic correlation functions are identical. Furthermore, Kriging software standardizes (scales, codes, normalizes) the original simulation inputs and outputs, which affects the distances h; also see Kleijnen and Mehdad (2013).

Note: Instead of correlation functions, geostatisticians use variograms, covariograms, and correlograms; see the literature on Kriging in geostatistcs in Sect. 5.1.

There are several types of correlation functions that give valid (positive definite) covariance matrices for stationary processes; see the general literature on GPs in Sect. 5.1, especially Rasmussen and Williams (2006, pp. 80–104). Geostatisticians often use so-called Matérn correlation functions, which are more complicated than the following three popular functions—displayed in Fig. 5.1 for a single input with parameter $\theta = 0.5$:

- Linear: $\rho(h) = \max(1 - \theta h, 0)$
- Exponential: $\rho(h) = \exp(-\theta h)$
- Gaussian: $\rho(h) = \exp(-\theta h^2)$

Note: It is straightforward to prove that the Gaussian correlation function has its point of inflection at $h = 1/\sqrt{2\theta}$, so in Fig. 5.1 this point lies at $h = 1$. Furthermore, the linear correlation function gives correlations $\rho(h)$ that are smaller than the exponential function gives, for $\theta > 0$ and $h > 0$; Fig. 5.1 demonstrates this behavior for $\theta = 0.5$. Finally, the linear correlation function gives $\rho(h)$ smaller than the Gaussian function does, for (roughly) $\theta > 0.45$ and $h > 0$. There are also correlation functions $\rho(h)$ that do not monotonically decrease as the lag h increases; this is called a "hole effect" (see http://www.statistik.tuwien.ac.at/ public/ dutt/ vorles/ geost_03/node80.html).

In simulation, a popular correlation function is

$$\rho(\mathbf{h}) = \prod_{j=1}^{k} \exp\left(-\theta_j h_j^{p_j}\right) = \exp\left(-\sum_{j=1}^{k} \theta_j h_j^{p_j}\right) \tag{5.13}$$

where θ_j quantifies the importance of input j—the higher θ_j is, the less effect input j has— and p_j quantifies the smoothness of the correlation function—e.g., $p_j = 2$ implies an infinitely differentiable function. Figure 5.1 has already illustrated an exponential function and a Gaussian function, which correspond with $p = 1$ and $p = 2$ in Eq. (5.13). (We shall discuss better measures of importance than θ_j, in Sect. 5.8.)

Exercise 5.4 *What is the value of $\rho(h)$ in Eq. (5.13) with $p > 0$ when $h = 0$ and $h = \infty$, respectively?*

Exercise 5.5 *What is the value of θ_j in Eq. (5.13) with $p_j > 0$ when input j has no effect on the output?*

Note: The choice of a specific type of correlation function may also affect the numerical properties of the Kriging model; see Harari and Steinberg (2014b).

Because $\rho(\mathbf{h})$ in Eq. (5.13) decreases as the distance $\mathbf{h}$ increases, the optimal weights $\boldsymbol{\lambda}_o$ in Eq. (5.6) are relatively high for old inputs close to the new input to be predicted.

Note: Some of the weights may be *negative*; see Wackernagel (2003, pp. 94–95). If negative weights give negative predictions and all the observed outputs w_i are nonnegative, then Deutsch (1996) sets negative weights and small positive weights to zero while restandardizing the sum of the remaining positive weights to one to make the predictor unbiased.

It is well known that Kriging results in bad extrapolation compared with interpolation; see Antognini and Zagoraiou (2010). Our intuitive explanation is that in *interpolation* the new point is surrounded by relatively many old points that are close to the new point; let us call them "close neighbors". Consequently, the predictor combines many old outputs that are strongly positively correlated with the new output. In *extrapolation*, however, there are fewer close neighbors. Note that linear regression also gives minimal predictor variance at the center of the experimental area; see Eq. (6.7).

5.2.2 Estimating the OK Parameters

A major problem in OK is that the optimal Kriging weights λ_i $(i = 1, \ldots, n)$ depend on the correlation function of the assumed metamodel—*but it is unknown which correlation function gives a valid metamodel.* In Kriging we usually select either an isotropic or an anisotropic type of correlation function and a specific type of decay such as linear, exponential, or Gaussian; see Fig. 5.1. Next we must estimate the parameter values; e.g. θ_j $(j = 1, \ldots, k)$ in Eq. (5.13). For this estimation we usually select the *maximum likelihood* (ML) criterion, which gives the ML estimators (MLEs) $\widehat{\theta}_j$. ML requires the selection of a distribution for the metamodel output $y(\mathbf{x})$ in Eq. (5.1). The standard distribution in Kriging is a multivariate normal, which explains the term GP. This gives the *log-likelihood function*

$$\begin{aligned} l(\mu, \tau^2, \boldsymbol{\theta}) = & -\ln[(2\pi)^{n/2}] \\ & - \frac{1}{2}\ln[\left|\tau^2\mathbf{R}(\boldsymbol{\theta})\right|] - \frac{1}{2}(\mathbf{w}-\mu\mathbf{1})'[\tau^2\mathbf{R}(\boldsymbol{\theta})]^{-1}(\mathbf{w}-\mu\mathbf{1}) \\ \text{with } \boldsymbol{\theta} \geq \mathbf{0} & \end{aligned} \tag{5.14}$$

where $|\cdot|$ denotes the determinant and $\mathbf{R}(\boldsymbol{\theta})$ denotes the correlation matrix of $\mathbf{y}$. Obviously, MLE requires that we minimize

$$\ln[\left|\tau^2\mathbf{R}(\boldsymbol{\theta})\right|] + (\mathbf{w}-\mu\mathbf{1})'[\tau^2\mathbf{R}(\boldsymbol{\theta})]^{-1}(\mathbf{w}-\mu\mathbf{1}). \tag{5.15}$$

We denote the resulting MLEs by a "hat", so the MLEs are $\widehat{\mu}$, $\widehat{\tau}^2$, and $\widehat{\boldsymbol{\theta}}$. This minimization is a difficult mathematical problem. The classic solution in Kriging is to "divide and conquer"—called the "profile likelihood" or the "concentrated likelihood" in mathematical statistics—as we summarize in the following algorithm (in practice we use standard Kriging software that we shall list near the end of this section).

Algorithm 5.1

1. Initialize $\widehat{\boldsymbol{\theta}}$, which defines $\widehat{\mathbf{R}}$.

2. Compute the generalized least squares (GLS) estimator of the mean:
$$\widehat{\mu} = (\mathbf{1}^T\widehat{\mathbf{R}}^{-1}\mathbf{1})^{-1}\mathbf{1}'\widehat{\mathbf{R}}^{-1}\mathbf{y}. \tag{5.16}$$

3. Substitute $\widehat{\mu}$ resulting from Step 2 and $\widehat{\mathbf{R}}$ resulting from Step 1 into the MLE variance estimator
$$\widehat{\tau}^2 = \frac{(\mathbf{w}-\widehat{\mu}\mathbf{1})'\widehat{\mathbf{R}}^{-1}(\mathbf{w}-\widehat{\mu}\mathbf{1})}{n}. \tag{5.17}$$

   Comment: $\widehat{\tau}^2$ has the denominator n, whereas the denominator $n-1$ is used by the classic unbiased estimator assuming $\mathbf{R} = \mathbf{I}$.

4. Solve the remaining problem in Eq. (5.15):
$$\text{Min } \widehat{\tau}^2|\widehat{\mathbf{R}}|^{-n}. \tag{5.18}$$

   Comment: This equation can be found in Lophaven et al. (2002, equation 2.25). To solve this nonlinear minimization problem, Lophaven et al. (2002) applies the classic Hooke-Jeeves heuristic. Gano et al. (2006) points out that this minimization problem is difficult because of "the multimodal and long near-optimal ridge properties of the likelihood function"; i.e., this problem is not convex.

5. Use the $\widehat{\boldsymbol{\theta}}$ that solves Eq. (5.18) in Step 4 to update $\widehat{\mathbf{R}}$, and substitute this updated $\widehat{\mathbf{R}}$ into Eqs. (5.16) and (5.17).

6. If the MLEs have not yet converged, then return to Step 2; else stop.

Note: Computational aspects are further discussed in Bachoc (2013), Butler et al. (2014), Gano et al. (2006), Jones et al. (1998), Li and Sudjianto (2005), Lophaven et al. (2002), Marrel et al. (2008), and Martin and Simpson (2005).

This difficult optimization problem implies that different MLEs may result from different software packages or from initializing the same package with different starting values; the software may even break down. The DACE software uses lower and upper limits for θ_j, which are usually hard to specify. Different limits may give completely different $\widehat{\theta}_j$, as the examples in Lin et al. (2004) demonstrate.

Note: Besides MLEs there are other estimators of $\boldsymbol{\theta}$; e.g., *restricted MLEs* (RMLEs) and cross-validation estimators; see Bachoc (2013), Rasmussen and Williams (2006, pp. 116–124), Roustant et al. (2012), Santner et al.

(2003, pp. 66–68), and Sundararajan and Keerthi (2001). Furthermore, we may use the LS criterion. We have already shown estimators for covariances in Eq. (3.31), but in Kriging the number of observations for a covariance of a given distance h decreases as that distance increases. Given these estimates for various values of h, Kleijnen and Van Beers (2004) and Van Beers and Kleijnen (2003) use the LS criterion to fit a linear correlation function.

Let us denote the MLEs of the OK parameters by $\widehat{\boldsymbol{\psi}} = (\widehat{\mu}, \hat{\tau}^2, \widehat{\boldsymbol{\theta}}')'$ with $\widehat{\boldsymbol{\theta}}' = (\widehat{\theta}_1, \ldots, \widehat{\theta}_k)$ in case of an anisotropic correlation function such as Eq. (5.13); obviously, $\widehat{\boldsymbol{\Sigma}} = \hat{\tau}^2 \widehat{\mathbf{R}}(\widehat{\boldsymbol{\theta}})$. *Plugging* these MLEs into Eq. (5.7), we obtain the predictor

$$\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) = \widehat{\mu} + \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)' \widehat{\boldsymbol{\Sigma}}^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}). \tag{5.19}$$

This predictor depends on the new point $\mathbf{x}_0$ only through $\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)$, because $\widehat{\mu}$ and $\widehat{\boldsymbol{\Sigma}}^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1})$ depend on the old I/O. The second term in this equation shows that this predictor is *nonlinear* (likewise, weighted least squares with estimated weights gives a nonlinear estimator in linear regression metamodels; see Sect. 3.4.4). However, most publications on Kriging compute the MSE of this predictor by simply plugging the MLEs of the Kriging parameters τ^2, $\sigma(\mathbf{x}_0)$, and $\boldsymbol{\Sigma}$ into Eq. (5.8):

$$\begin{aligned}\text{MSE}[\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})] &= \widehat{\tau}^2 - \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)' \widehat{\boldsymbol{\Sigma}}^{-1} \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0) \\ &\quad + \frac{(1 - \mathbf{1}' \widehat{\boldsymbol{\Sigma}}^{-1} \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0))2}{\mathbf{1}' \widehat{\boldsymbol{\Sigma}}^{-1} \mathbf{1}}\end{aligned} \tag{5.20}$$

We shall discuss a bootstrapped estimator of the true MSE of this nonlinear predictor, in the next section (Sect. 5.3).

Note: Martin and Simpson (2005) discusses alternative approaches—namely, validation and Akaike's information criterion (AIC)—and finds that ignoring the randomness of the estimated Kriging parameters underestimates the true variance of the Kriging predictor. Validation for estimating the variance of the Kriging predictor is also discussed in Goel et al. (2006) and Viana and Haftka (2009). Furthermore, Thiart et al. (2014) confirms that the plug-in MSE defined in Eq. (5.20) underestimates the true MSE, and discusses alternative estimators of the true MSE. Jones et al. (1998) and Spöck and Pilz (2015) also imply that the plug-in estimator underestimates the true variance. Stein (1999) gives asymptotic results for Kriging with $\widehat{\psi}$.

We point out that Kriging gives a predictor plus a measure for the accuracy of this predictor; see Eq. (5.20). Some other metamodels—e.g., splines—do not quantify the accuracy of their predictor; see Cressie (1993, p. 182). Like Kriging, linear regression metamodels do quantify the accuracy; see Eq. (3.41).

The MSE in Eq. (5.20) is also used to compute a two-sided symmetric $(1 - \alpha)$ *confidence interval* (CI) for the OK predictor at $\mathbf{x}_0$, where

$\widehat{\sigma}^2_{\text{OK}}\{\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})\}$ equals MSE$[\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})]$ and (say) $a \pm b$ denotes the interval $[a-b, a+b]$:

$$P[w(\mathbf{x}_0) \in [\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) \pm z_{\alpha/2}\sqrt{\widehat{\sigma}^2_{\text{OK}}\{\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})\}}] = 1 - \alpha. \tag{5.21}$$

There is much *software* for Kriging. In our own experiments we have used DACE, which is a free-of-charge MATLAB toolbox well documented in Lophaven et al. (2002). Alternative free software is the R package DiceKriging—which is well documented in Roustant et al. (2012)—and the object-oriented software called the "ooDACE toolbox"—documented in Couckuyt et al. (2014). PeRK programmed in C is documented in Santner et al. (2003, pp. 215–249). More free software is mentioned in Frazier (2011) and in the textbooks and websites mentioned in Sect. 5.1; also see the Gaussian processes for machine learning (GPML) toolbox, detailed in Rasmussen and Nickisch (2010). We also refer to the following four toolboxes (in alphabetical order):
MPERK on
http://www.stat.osu.edu/~comp_exp/jour.club/MperkManual.pdf
STK on
http://sourceforge.net/projects/kriging/
http://octave.sourceforge.net/stk/,
SUMO on
http://www.sumo.intec.ugent.be/,
and Surrogates on
https://sites.google.com/site/felipeacviana/surroga
testoolbox.

Finally, we refer to the commercial JMP/SAS site:
https://www.jmp.com/en_us/software/feature-index.html#K.

Note: For large data sets, the Kriging computations may become problematic; solutions are discussed in Gramacy and Haaland (2015) and Meng and Ng (2015).

As we have already stated in Sect. 1.2, we adhere to a frequentist view in this book. Nevertheless, we mention that there are many publications that interpret Kriging models in a *Bayesian* way. A recent article is Yuan and Ng (2015); older publications are referenced in Kleijnen (2008). Our major problem with the Bayesian approach to Kriging is that we find it hard to come up with prior distributions for the Kriging parameters $\boldsymbol{\psi}$, because we have little intuition about the correlation parameters $\boldsymbol{\theta}$; e.g., what is the prior distribution of $\boldsymbol{\theta}$, in the Kriging metamodel of the $M/M/1$ simulation model?

Note: Kriging seems related to so-called *moving least squares* (MLS), which originated in curve and surface fitting and fits a continuous function using a weighted least squares (WLS) criterion that gives more weight to old points close to the new point; see Lancaster and Salkauskas (1986) and also Forrester and Keane (2009) and Toropov et al. (2005).

The Kriging metamodel may also include *qualitative* inputs besides quantitative inputs. The challenge is to specify a valid covariance matrix; see Zhou et al. (2011).

5.3 Bootstrapping and Conditional Simulation for OK in Deterministic Simulation

In the preceding section we mentioned that in the present section we discuss a bootstrap approach to estimating the MSE of the nonlinear predictor with plugged-in estimated Kriging parameters $\widehat{\boldsymbol{\psi}}$ in Eq. (5.19). We have already discussed the general principles of bootstrapping in Sect. 3.3.5. Now we discuss parametric bootstrapping of the GP assumed in OK that was specified in Eq. (5.1). We also discuss a bootstrap variant called "conditional simulation". Hasty readers may skip this section, because parametric bootstrapping and its variant are rather complicated and turn out to give CIs with coverages and lengths that are not superior compared with the CI specified in Eq. (5.21).

5.3.1 Bootstrapped OK (BOK)

For bootstrapping we use the notation that we introduced in Sect. 3.3.5. So we denote bootstrapped data by the superscript $*$; e.g., $(\mathbf{X}, \mathbf{w}^*)$ denotes the original input and the bootstrapped output of the simulation model. We define bootstrapped estimators analogously to the original estimators, but we compute the bootstrapped estimators from the bootstrapped data instead of the original data; e.g., we compute $\widehat{\boldsymbol{\psi}}$ from $(\mathbf{X}, \mathbf{w})$, but $\widehat{\boldsymbol{\psi}}^*$ from $(\mathbf{X}, \mathbf{w}^*)$. We denote the bootstrap sample size by B and the bth bootstrap observation in this sample by the subscript b with $b = 1, \ldots, B$.

Following Kleijnen and Mehdad (2013), we define the following $(1+n)$-dimensional Gaussian or "normal" (N_{1+n}) distribution:

$$\begin{pmatrix} y(\mathbf{x}_0) \\ y(\mathbf{x}) \end{pmatrix} \sim \mathrm{N}_{1+n}\left[\mu \mathbf{1}_{1+n}, \begin{pmatrix} \tau^2 & \boldsymbol{\sigma}(\mathbf{x}_0)' \\ \boldsymbol{\sigma}(\mathbf{x}_0) & \boldsymbol{\Sigma} \end{pmatrix}\right], \tag{5.22}$$

where all symbols were defined in the preceding section. Obviously, Eq. (5.22) implies $y(\mathbf{x}) \sim \mathrm{N}_n(\mu \mathbf{1}_n, \boldsymbol{\Sigma})$.

Li and Zhou (2015) extends Den Hertog et al. (2006)'s bootstrap method for estimating the variance from univariate GP models to so-called "pairwise meta-modeling" of multivariate GP models assuming nonseparable covariance functions. We saw that if $\mathbf{x}_0$ gets closer to an old point $\mathbf{x}$, then the predictor variance decreases and—because OK is an exact interpolator in deterministic simulation—this variance becomes exactly zero when $\mathbf{x}_0 = \mathbf{x}$. Furthermore, N_{1+n} in Eq. (5.22) implies that the distribution of the new output—given the n old outputs—is the *conditional* normal distribution

$$\mathrm{N}\left[\widehat{\mu}+\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'\widehat{\boldsymbol{\Sigma}}^{-1}[\mathbf{y}(\mathbf{x})-\widehat{\mu}\mathbf{1}_n],\widehat{\tau}^2-\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'\widehat{\boldsymbol{\Sigma}}^{-1}\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)\right]. \tag{5.23}$$

We propose the following BOK pseudo-algorithm.

Algorithm 5.2

1. Use $\mathrm{N}_k\left(\widehat{\mu}\mathbf{1}_k,\widehat{\boldsymbol{\Sigma}}\right)$ B times to sample the n old outputs $\mathbf{y}_b^*(\mathbf{X},\widehat{\boldsymbol{\psi}})=(y_{1;b}^*(\mathbf{X},\widehat{\boldsymbol{\psi}}),\ldots,y_{k;b}^*(\mathbf{X},\widehat{\boldsymbol{\psi}}))'$ where $\widehat{\boldsymbol{\psi}}$ is estimated from the old simulation I/O data $(\mathbf{X},\mathbf{w})$. For each new point $\mathbf{x}_0$ repeat steps 2 through 4 B times.

2. Given the n old bootstrapped outputs $\mathbf{y}_b^*(\mathbf{X},\widehat{\boldsymbol{\psi}})$ of step 1, sample the new output $y_b^*(\mathbf{x}_0,\widehat{\boldsymbol{\psi}})$ from the conditional normal distribution defined in Eq. (5.23).

3. Using the n old bootstrapped outputs $\mathbf{y}_b^*(\mathbf{X},\widehat{\boldsymbol{\psi}})$ of step 1, compute the bootstrapped MLE $\widehat{\boldsymbol{\psi}}_b^*$. Next calculate the bootstrapped predictor

$$\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}_b^*)=\widehat{\mu}_b^*+\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0,\widehat{\theta}_b^*)'\widehat{\boldsymbol{\Sigma}}^{-1}(\widehat{\theta}_b^*)[\mathbf{y}_b^*(\mathbf{X},\widehat{\boldsymbol{\psi}})-\widehat{\mu}_b^*\mathbf{1}_n]. \tag{5.24}$$

4. Given $\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}_b^*)$ of step 3 and $y_b^*(\mathbf{x}_0,\widehat{\boldsymbol{\psi}})$ of step 2, compute the bootstrap estimator of the squared prediction error (SPE):

$$\mathrm{SPE}_b^*=\mathrm{SPE}[\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}_b^*)]=[\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}_b^*)-y_b^*(\mathbf{x}_0,\widehat{\boldsymbol{\psi}})]^2.$$

5. Given the B bootstrap samples SPE_b^* $(b=1,\ldots,B)$ resulting from steps 1 through 4, compute the bootstrap estimator of $\mathrm{MSPE}[\widehat{y}(\mathbf{x}_0)]$ (this MSPE was defined in Eq. (5.8):

$$\mathrm{MSPE}^*=\frac{\sum_{b=1}^{B}\mathrm{SPE}_b^*}{B}. \tag{5.25}$$

If we ignore the bias of the BOK predictor $\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}^*)$, then Eq. (5.25) gives $\widehat{\sigma}^2[\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}^*)]$ which is the bootstrap estimator of $\sigma^2[\widehat{y}(\mathbf{x}_0|\widehat{\boldsymbol{\psi}})]$. We abbreviate $\widehat{\sigma}^2[\widehat{y}(\mathbf{x}_0,\widehat{\boldsymbol{\psi}}^*)]$ to $\widehat{\sigma}^2_{\mathrm{BOK}}$. The standard error (SE) of $\widehat{\sigma}^2_{\mathrm{BOK}}$ follows from Eq. (5.25):

$$\mathrm{SE}(\widehat{\sigma}^2_{\mathrm{BOK}})=\sqrt{\frac{\sum_{b=1}^{B}(\mathrm{SPE}_b^*-\mathrm{MSPE}^*)^2}{(B-1)B}}.$$

We apply t_{B-1} (t-statistic with $B-1$ degrees of freedom) to obtain a two-sided symmetric $(1-\alpha)$ CI for σ^2_{BOK}:

$$P[\sigma^2_{\mathrm{OK}}\in\widehat{\sigma}^2_{\mathrm{BOK}}\pm t_{B-1;\alpha/2}\mathrm{SE}(\widehat{\sigma}^2_{\mathrm{BOK}})]=1-\alpha. \tag{5.26}$$

Obviously, if $B \uparrow \infty$, then $t_{B-1;\alpha/2} \downarrow z_{\alpha/2}$ where $z_{\alpha/2}$ denotes the $\alpha/2$ quantile of the standard normal variable $z \sim \mathrm{N}(0,1)$; typically B is so high (e.g., 100) that we can indeed replace $t_{B-1;\alpha/2}$ by $z_{\alpha/2}$.

Figure 5.2 illustrates BOK for the following test function, taken from Forrester et al. (2008, p. 83):

$$w(x) = (6x-2)^2 \sin(12x-4) \text{ with } 0 \leq x \leq 1. \tag{5.27}$$

This function has one local minimum at $x = 0.01$, and one global minimum at $x = 0.7572$ with output $w = -6.02074$; we shall return to this function in the next chapter, in which we discuss simulation optimization. The plot shows that each of the B bootstrap samples has its own old output values $\mathbf{y}_b^*$. Part (a) displays only $B = 5$ samples to avoid cluttering-up the plot. Part (b) shows less "wiggling" than part (a); $\widehat{y}(\mathbf{x}, \widehat{\boldsymbol{\psi}}_b^*)$, which are the predictions at old points, coincide with $\mathbf{y}_b^*(\mathbf{X}, \widehat{\boldsymbol{\psi}})$, which are the values sampled in part (a). Part (c) uses $B = 100$.

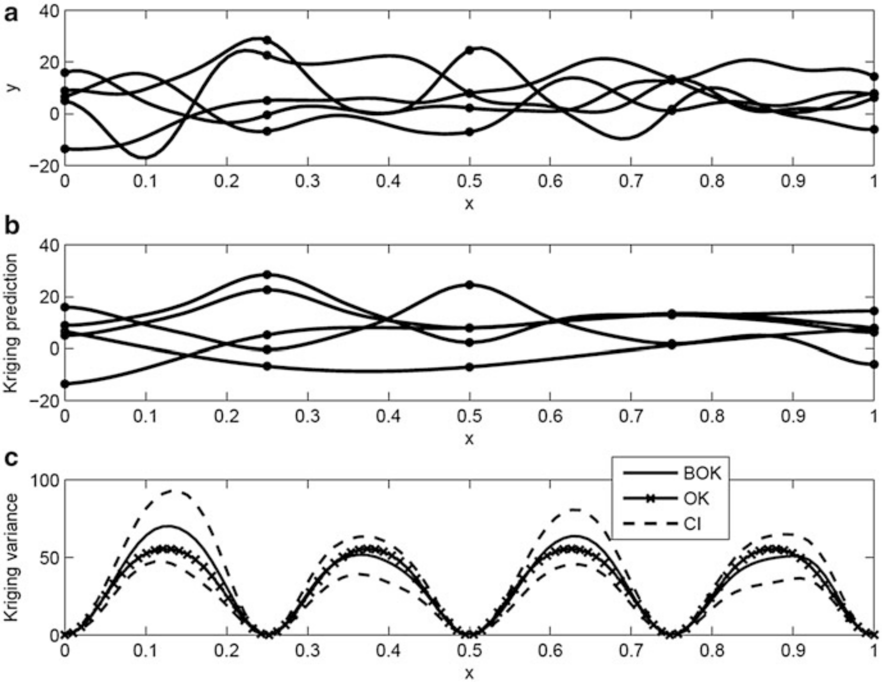


FIGURE 5.2. BOK for the test function in Forrester et al. (2008): (**a**) jointly sampled outputs at 5 equi-spaced old and 98 equi-spaced new points, for $B = 5$; (**b**) Kriging predictions for 98 new points based on 5 old points sampled in (**a**); (**c**) estimated predictor variances and their 95 % CIs for $B = 100$

To compute a two-sided symmetric $(1-\alpha)$ CI for the predictor at $\mathbf{x}_0$, we may use the OK point predictor $\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})$ and $\widehat{\sigma}^2_{\text{BOK}}$(equal to the MSE in Eq. (5.25)):

$$P\{w(\mathbf{x}_0) \in \widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) \pm z_{\alpha/2}\sqrt{\widehat{\sigma}^2_{\text{BOK}}}\} = 1-\alpha. \tag{5.28}$$

If $\widehat{\sigma}^2_{\text{OK}} < \widehat{\sigma}^2_{\text{BOK}}$, then this CI is longer and gives a higher coverage than the CI in Eq. (5.21). Furthermore, we point out that Yin et al. (2010) also finds empirically that a Bayesian approach accounting for the randomness of the estimated Kriging parameters gives a wider CI—and hence higher coverage—than an approach that ignores this estimation.

5.3.2 Conditional Simulation of OK (CSOK)

We denote *conditional simulation* (CS) of OK by CSOK. This method ensures $\hat{y}(\mathbf{x}, \hat{\boldsymbol{\psi}}^*_b) = w(\mathbf{x})$; i.e., in all the bootstrap samples the prediction at an old point equals the observed value. Part (a) of Fig. 5.3 may help understand Algorithm 5.3 for CSOK, which copies steps 1 through 3 of Algorithm 5.2 for BOK in the preceding subsection.

Note: Algorithm 5.3 is based on Kleijnen and Mehdad (2013), which follows Chilès and Delfiner (2012, pp. 478–650). CS may also be implemented through the R software package called "DiceKriging"; see Roustant et al. (2012).

Algorithm 5.3

1. Use $\text{N}_n(\widehat{\mu}\mathbf{1}_n, \widehat{\mathbf{\Sigma}})$ B times to sample the n old outputs $\mathbf{y}^*_b(\mathbf{X}, \widehat{\boldsymbol{\psi}}) = (y^*_{1;b}(\mathbf{X}, \widehat{\boldsymbol{\psi}}), \ldots, y^*_{k;b}(\mathbf{X}, \widehat{\boldsymbol{\psi}}))'$ where $\widehat{\boldsymbol{\psi}}$ is estimated from the old simulation I/O data $(\mathbf{X}, \mathbf{w})$. For each new point $\mathbf{x}_0$, repeat steps 2 through 4 B times.

2. Given the n old bootstrapped outputs $\mathbf{y}^*_b(\mathbf{X}, \widehat{\boldsymbol{\psi}})$ of step 1, sample the new output $y^*_b(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})$ from the conditional normal distribution in Eq. (5.23).

3. Using the k old bootstrapped outputs $\mathbf{y}^*_b(\mathbf{X}, \widehat{\boldsymbol{\psi}})$ of step 1, compute the bootstrapped MLE $\widehat{\boldsymbol{\psi}}^*_b$. Next calculate the bootstrapped predictor

$$\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}^*_b) = \widehat{\mu}^*_b + \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'\widehat{\mathbf{\Sigma}}^{-1}(\widehat{\boldsymbol{\theta}}^*_b)[\mathbf{y}^*_b(\mathbf{X}, \widehat{\boldsymbol{\psi}}) - \widehat{\mu}^*_b\mathbf{1}_n]. \tag{5.29}$$

4. Combining the OK estimator defined in Eq. (5.19) and the BOK estimator defined in Eq. (5.29), compute the CSOK predictor

$$\widehat{y}_{\text{CSOK}}(\mathbf{x}_0, b) = \widehat{\mu} + \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'\widehat{\mathbf{\Sigma}}^{-1}(\mathbf{w} - \widehat{\mu}\mathbf{1}_n) + [y^*_b(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) - \widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}^*_b)]. \tag{5.30}$$

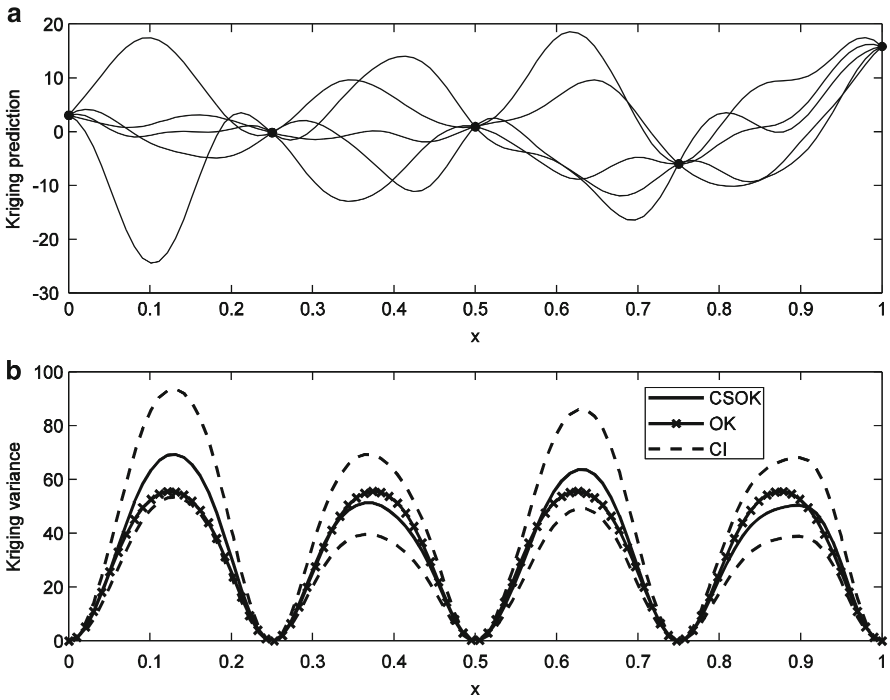


FIGURE 5.3. CSOK for the test function in Forrester et al. (2008): (**a**) predictions at 98 new points, for $B = 5$; (**b**) estimated predictor variances and their 95 % CIs for $B = 100$, and OK's predictor variances

Given these B estimators $\widehat{y}_{\mathrm{CSOK}}(\mathbf{x}_0, b)$ $(b = 1, \ldots, B)$, compute the CSOK estimator of MSPE$[\widehat{y}(\mathbf{x}_0)]$:

$$\widehat{\sigma}^2[\widehat{y}_{\mathrm{CSOK}}(\mathbf{x}_0)] = \frac{\sum_{b=1}^{B}[\widehat{y}_{\mathrm{CSOK}}(\mathbf{x}_0, b) - \overline{\widehat{y}}_{\mathrm{CSOK}}(\mathbf{x}_0)]^2}{B-1} \text{ with}$$
$$\overline{\widehat{y}}_{\mathrm{CSOK}}(\mathbf{x}_0) = \frac{\sum_{b=1}^{B}\widehat{y}_{\mathrm{CSOK}}(\mathbf{x}_0, b)}{B}. \tag{5.31}$$

We abbreviate $\widehat{\sigma}^2[\widehat{Y}_{\mathrm{CSOK}}(\mathbf{x}_0)]$ to $\widehat{\sigma}^2_{\mathrm{CSOK}}$. Mehdad and Kleijnen (2014) proves that $\widehat{\sigma}^2_{\mathrm{CSOK}} \leq \widehat{\sigma}^2_{\mathrm{BOK}}$; in practice, it is not known how much smaller $\widehat{\sigma}^2_{\mathrm{CSOK}}$ is than $\widehat{\sigma}^2_{\mathrm{BOK}}$. We therefore apply a two-sided asymmetric $(1 - \alpha)$ CI for σ^2_{OK} using $\widehat{\sigma}^2_{\mathrm{CSOK}}$ and the chi-square statistic χ^2_{B-1} (this CI replaces the CI for BOK in Eq. (5.28), which assumes B IID variables):

$$P\left(\frac{(B-1)\widehat{\sigma}^2_{\mathrm{CSOK}}}{\chi^2_{B-1;1-\alpha/2}} \leq \sigma^2_{\mathrm{OK}} \leq \frac{(B-1)\widehat{\sigma}^2_{\mathrm{CSOK}}}{\chi^2_{B-1;\alpha/2}}\right) = 1 - \alpha. \tag{5.32}$$

Part (b) of Fig. 5.3 displays $\widehat{\sigma}^2_{\text{CSOK}}$ defined in Eq. (5.31) and its 95 % CIs defined in Eq. (5.32) based on $B = 100$ bootstrap samples; it also displays $\widehat{\sigma}^2_{\text{OK}}$ following from Eq. (5.20). Visual examination of this part suggests that $\widehat{\sigma}^2_{\text{CSOK}}$ tends to exceed $\widehat{\sigma}^2_{\text{OK}}$.

Next, we display both $\widehat{\sigma}^2_{CSOK}$ and $\widehat{\sigma}^2_{BOK}$ and their CIs, for various values of B, in Fig. 5.4. This plot suggests that $\widehat{\sigma}^2_{\text{CSOK}}$ is not significantly smaller than $\widehat{\sigma}^2_{BOK}$. These results seem reasonable, because both CSOK and BOK use $\widehat{\boldsymbol{\psi}}$, which is the *sufficient statistic* of the GP computed from the same $(\mathbf{X}, \mathbf{w})$. CSOK seems simpler than BOK, both computationally and conceptually. CSOK gives better predictions for new points close to old points; but then again, BOK is meant to improve the predictor variance—not the predictor itself.

We may use $\widehat{\sigma}^2_{\text{CSOK}}$ to compute a CI for the OK predictor, using the analogue of Eq. (5.28):

$$P\left\{w(\mathbf{x}_0) \in \widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) \pm z_{\alpha/2}\sqrt{\widehat{\sigma}^2_{\text{CSOK}}}\right\} = 1 - \alpha. \tag{5.33}$$

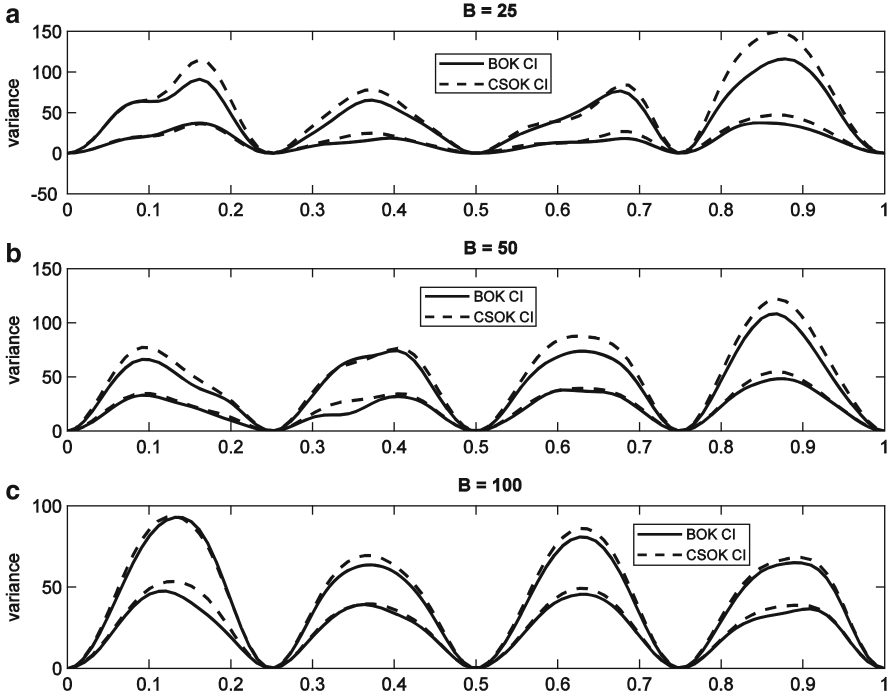


FIGURE 5.4. CIs for BOK versus CSOK for various B values, using the test function in Forrester et al. (2008)

Moreover, we can derive an alternative CI; namely, a *distribution-free* two-sided asymmetric CI based on the so-called *percentile method* (which we defined in Eq. (3.14)). We apply this method to $\widehat{y}_{\text{CSOK}}(\mathbf{x}_0, b)$ ($b = 1, \ldots, B$), which are the B CSOK predictors defined in Eq. (5.30). Because the percentile method uses order statistics, we now denote $\widehat{y}_{\text{CSOK}}(\mathbf{x}_0, b)$ by $\widehat{y}_{\text{CSOK};\ b}\ (\mathbf{x}_0)$, apply the usual subscript (.) (e.g., $(B\alpha/2)$) to denote order statistics (resulting from sorting the B values from low to high), and select B such that $B\alpha/2$ and $B(1-\alpha/2)$ are integers:

$$P[\widehat{y}_{\text{CSOK};\ (B\alpha/2)}(\mathbf{x}_0) \leq w(\mathbf{x}_0) \leq \widehat{y}_{\text{CSOK};\ (B(1-\alpha/2))}(\mathbf{x}_0)] = 1 - \alpha. \quad (5.34)$$

An advantage of the percentile method is that this CI does not include negative values if the simulation output is not negative; also see Sect. 5.7 on bootstrapping OK to preserve known characteristics of the I/O functions (nonnegative outputs, monotonic I/O functions, etc.). We do not apply the percentile method to BOK, because BOK gives predictions at the n old points that do not equal the observed old simulation outputs w_i.

For OK, BOK, and CSOK Mehdad and Kleijnen (2015a) studies CIs with a nominal coverage of $1-\alpha$ and reports the estimated expected coverage $1 - E(\widehat{\alpha})$ and the estimated expected length $E(l)$ of the CIs, for a GP with two inputs so $k = 2$ and an anisotropic Gaussian correlation function such as Eq. (5.13) with $p = 2$. In general, we prefer the CI with the shortest length, unless this CI gives too low coverage. The reported results show that OK with $\widehat{\sigma}_{\text{OK}}$ gives shorter lengths than CSOK with $\widehat{\sigma}_{\text{CSOK}}$, and yet OK gives estimated coverages that are not significantly lower. The percentile method for CSOK gives longer lengths than OK, but its coverage is not significantly better than OK's coverage. Altogether the results do not suggest that BOK or CSOK is superior, so we recommend OK when predicting a new output; i.e., OK seems a robust method.

Exercise 5.6 *Consider the three alternative CIs that use OK, BOK, and CSOK, respectively. Do you think that the length of such a CI for a new point tends to decrease or increase as n (number of old points) increases?*

5.4 Universal Kriging (UK) in Deterministic Simulation

UK replaces the constant μ in Eq. (5.1) for OK by $\mathbf{f}(\mathbf{x})^{'}\boldsymbol{\beta}$ where $\mathbf{f}(\mathbf{x})$ is a $q \times 1$ vector of known functions of $\mathbf{x}$ and $\boldsymbol{\beta}$ is a $q \times 1$ vector of unknown parameters (e.g., if $k = 1$, then UK may replace μ by $\beta_0 + \beta_1 x$, which is called a "linear trend"):

$$y(\mathbf{x}) = \mathbf{f}(\mathbf{x})^{'}\boldsymbol{\beta} + M(\mathbf{x}) \quad \text{with } \mathbf{x} \in \mathbb{R}^k. \quad (5.35)$$

The disadvantage of UK compared with OK is that UK requires the estimation of additional parameters. More precisely, besides β_0 UK involves

$q-1$ parameters, whereas OK involves only $\beta_0 = \mu$. We conjecture that the estimation of the extra $q-1$ parameters explains why UK has a higher MSE. In practice, most Kriging models do not use UK but OK

Note: This higher MSE for UK is also discussed in Ginsbourger et al. (2009) and Tajbakhsh et al. (2014). However, Chen et al. (2012) finds that UK in stochastic simulation with CRN may give better estimates of the gradient; also see Sect. 5.6. Furthermore, to eliminate the effects of estimating $\boldsymbol{\beta}$ in UK, Mehdad and Kleijnen (2015b) applies *intrinsic random functions* (IRFs) and derives the corresponding *intrinsic Kriging* (IK) and *stochastic intrinsic Kriging* (SIK). An IRF applies a linear transformation such that $\mathbf{f}(\mathbf{x})'\boldsymbol{\beta}$ in Eq. (5.35) vanishes. Of course, this transformation also changes the covariance matrix $\boldsymbol{\Sigma}_M$, so the challenge becomes to determine a covariance matrix of IK that is valid (symmetric and "conditionally" positive definite). Experiments suggest that IK outperforms UK, and SIK outperforms SK. Furthermore, a refinement of UK is so-called *blind Kriging*, which does not assume that the functions $\mathbf{f}(\mathbf{x})$ are known. Instead, blind Kriging chooses these functions from a set of candidate functions, assuming heredity (which we discussed below Eq. (4.11)) and using Bayesian techniques (which we avoid in this book; see Sect. 5.2). Blind Kriging is detailed in Joseph et al. (2008) and also in Couckuyt et al. (2012). Finally, Deng et al. (2012) compares UK with a new Bayesian method that also tries to eliminate unimportant inputs in the Kriging metamodel; the elimination of unimportant inputs we discussed in Chap. 4 on screening.

5.5 Designs for Deterministic Simulation

An $n \times k$ design matrix $\mathbf{X}$ specifies the n combinations of the k simulation inputs. The literature on designs for Kriging in deterministic simulation abounds, and proposes various design types. Most popular are Latin hypercube designs (LHDs). Alternative types are orthogonal array, uniform, maximum entropy, minimax, maximin, integrated mean squared prediction error (IMSPE), and "optimal" designs.

Note: Many references are given in Chen and Zhou (2014), Damblin et al. (2013), Janssen (2013), and Wang et al. (2014). Space-filling designs that account for statistical dependencies among the k inputs—which may be quantitative or qualitative—are given in Bowman and Woods (2013). A textbook is Lemieux (2009). More references are given in Harari and Steinberg (2014a), and Kleijnen (2008, p. 130). Relevant websites are

http://lib.stat.cmu.edu

and

http://www.spacefillingdesigns.nl/.

LHDs are specified through *Latin hypercube sampling* (LHS). Historically speaking, McKay et al. (1979) invented LHS not for Kriging but for risk

	Input 3's level				
Input 2's level	1	2	3	4	5
1	1	4	2	5	3
2	4	1	3	2	5
3	3	2	5	4	1
4	2	5	1	3	4
5	5	3	4	1	2

TABLE 5.1. A Latin square with three inputs, each with five levels

analysis using deterministic simulation models ("computer codes"); LHS was proposed as an alternative for crude Monte Carlo sampling (for Monte Carlo methods we refer to Chap. 1). LHS assumes that an adequate metamodel is more complicated than a low-order polynomial (these polynomial metamodels and their designs were discussed in the preceding three chapters). LHS does not assume a specific metamodel that approximates the I/O function defined by the underlying simulation model; actually, LHS focuses on the input space formed by the k–dimensional unit cube defined by the standardized simulation inputs. LHDs are one of the space-filling types of design (LHDs will be detailed in the next subsection, Sect. 5.5.1).

Note: It may be advantageous to use space-filling designs that allow sequential addition of points; examples of such designs are the *Sobol sequences* detailed on

`http://en.wikipedia.org/wiki/Sobol_sequence#References`.

We also refer to the *nested LHDs* in Qian et al. (2014) and the "sliced" LHDs in Ba et al. (2014), Li et al. (2015), and Yang et al. (2014); these sliced designs are useful for experiments with both qualitative and quantitative inputs. Furthermore, taking a subsample of a LHD—as we do in validation—destroys the LHD properties. Obviously, the most flexible method allowing addition and elimination of points is a simple random sample of n points in the k-dimensional input space.

In Sect. 5.5.1 we discuss LHS for designs with a given number of input combinations, n; in Sect. 5.5.2 we discuss designs that determine n sequentially and are customized.

5.5.1 Latin Hypercube Sampling (LHS)

Technically, LHS is a type of stratified sampling based on the classic *Latin square* designs, which are square matrixes filled with different symbols such that each symbol occurs exactly once in each row and exactly once in each column. Table 5.1 is an example with $k = 3$ inputs and five levels per input; input 1 is the input of real interest, whereas inputs 2 and 3 are nuisance inputs or block factors (also see our discussion on blocking in Sect. 2.10). This example requires only $n = 5 \times 5 = 25$ combinations instead of $5^3 = 125$ combinations. For further discussion of Latin (and Graeco-Latin) squares we refer to Chen et al. (2006).

	Input 3's level				
Input 2's level	1	2	3	4	5
1	1	2	3	4	5
2	5	1	2	3	4
3	4	5	1	2	3
4	3	4	5	1	2
5	2	3	4	5	1

TABLE 5.2. A systematic Latin square with three inputs, each with five levels

Note: Another Latin square—this time, constructed in a *systematic* way—is shown in Table 5.2. This design, however, may give a biased estimator of the effect of interest. For example, suppose that the input of interest (input 1) is wheat, and wheat comes in five varieties. Suppose further that this table determines the way wheat is planted on a piece of land; input 2 is the type of harvesting machine, and input 3 is the type of fertilizer. If the land shows a very fertile strip that runs from north-west to south-east (see the main diagonal of the matrix in this table), then the effect of wheat type 1 is overestimated. Therefore *randomization* should be applied to protect against unexpected effects. Randomization makes such bias unlikely—but not impossible. Therefore random selection may be corrected if its realization happens to be too systematic. For example, a LHD may be corrected to give a "nearly" orthogonal design; see Hernandez et al. (2012), Jeon et al. (2015), and Vieira et al. (2011).

The following algorithm details LHS for an experiment with n combinations of k inputs (also see Helton et al. (2006b).

Algorithm 5.4

1. Divide the range of each input into $n > 1$ mutually exclusive and exhaustive intervals of equal probability.
   Comment: If the distribution of input values is uniform on $[a, b]$, then each interval has length $(b - a)/n$. If the distribution is Gaussian, then intervals near the mode are shorter than in the tails.

2. Randomly select one value for x_1 from each interval, without replacement, which gives n values $x_{1;1}$ through $x_{1;n}$.

3. Pair these n values with the n values of x_2, randomly without replacement.

4. Combine these n pairs with the n values of x_3, randomly without replacement to form n triplets.

5. And so on, until a set of n n-tupples is formed.

Table 5.3 and Fig. 5.5 give a LHD example with $n = 5$ combinations of the two inputs x_1 and x_2; these combinations are denoted by as in fig. 5.5. The table shows that each input has five discrete levels, which are labelled 1 through 5. If the inputs are continuous, then the label (say) 1 may denote a value within interval 1; see Fig. 5.5.

LHS does not imply a strict mathematical relationship between n (number of combinations actually simulated) and k (number of simulation inputs), whereas DOE uses (for example) $n = 2^k$ so n drastically increases with k. Nevertheless, if LHS keeps n "small" and k is "large", then the resulting LHD covers the experimental domain $\mathbb{R}^k$ so sparsely that the fitted Kriging model may be an inadequate metamodel of the underlying simulation model. Therefore a well-known rule-of-thumb for LHS in Kriging is $n = 10k$; see Loeppky et al. (2009).

	x_1				
x_2	1	2	3	4	5
1			●		
2		●			
3				●	
4					●
5	●				

TABLE 5.3. A LHS example with $n = 5$ combinations of two inputs x_1 and x_2

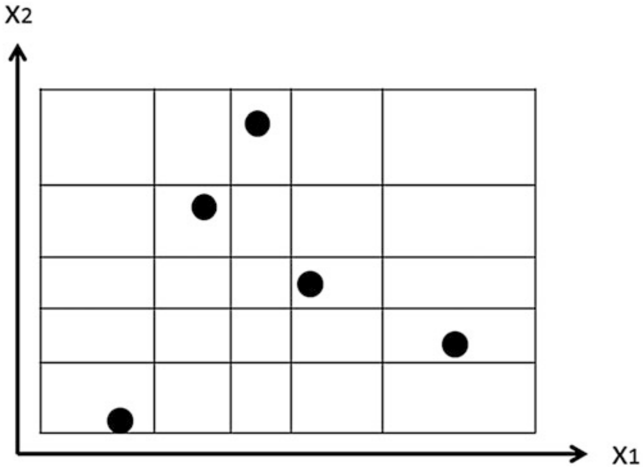


FIGURE 5.5. LHS example with $n = 5$ combinations of two inputs x_1 and x_2

Note: Wang et al. (2014) recommends $n = 20k$. Furthermore, Hernandez et al. (2012) provides a table for LHDs with acceptable nonorthogonality for various $(n,\, k)$ combinations with $n \leq 1{,}025$ and $k \leq 172$.

Usually, LHS assumes that the k inputs are independently distributed—so their joint distribution becomes the product of their k individual marginal distributions—and the marginal distributions are uniform (symbol U) in the interval $(0, 1)$ so $x_j \sim \mathrm{U}(0, 1)$. An alternative assumption is a multivariate Gaussian distribution, which is completely characterized by its covariances and means. For nonnormal joint distributions, LHS may use Spearman's correlation coefficient (discussed in Sect. 3.6.1); see Helton et al. (2006b). If LHS assumes a nonuniform marginal distribution for x_j (as we may assume in risk analysis, discussed in Sect. 5.9), then LHS defines n—mutually exclusive and exhaustive—subintervals $[l_{j;g},\, u_{j'g}]$ $(g = 1, \ldots, n)$ for the standardized x_j such that each subinterval has the same probability; i.e., $P(l_{j;g} \leq x_j \leq u_{j;g}) = 1/n$. This implies that near the mode of the x_j distribution, the subintervals are relatively short, compared with the subintervals in the tails of this distribution.

In LHS we may either *fix* the value of x_j to the middle of the subinterval g so $x_j = (l_{j;g} + u_{j;g})/2$ or we may *sample* the value of x_j within that subinterval accounting for the distribution of its values. Fixing x_j is attractive when we wish to estimate the sensitivity of the output to the inputs (see Sect. 5.8, in which we shall discuss global sensitivity analysis through Sobol's indexes). A random x_j is attractive when we wish to estimate the probability of the output exceeding a given threshold as a function of an uncertain input x_j, as we do in risk analysis (see Sect. 5.9).

LHDs are *noncollapsing*; i.e., if an input turns out to be unimportant, then each remaining individual input is still sampled with one observation per subinterval. DOE, however, then gives multiple observations for the same value of a remaining input—which is a waste in deterministic simulation (in stochastic simulation it improves the accuracy of the estimated intrinsic noise). Kriging with an anisotropic correlation function may benefit from the noncollapsing property of LHS, when estimating the correlation parameters θ_j. Unfortunately, projections of a LHD point in n dimensions onto more than one dimension may give "bad" designs. Therefore standard LHS is further refined, leading to so-called maximin LHDs and nearly-orthogonal LHDs.

Note: For these LHDs we refer to Damblin et al. (2013), Dette and Pepelyshev (2010), Deutsch and Deutsch (2012), Georgiou and Stylianou (2011), Grosso et al. (2009), Janssen (2013), Jourdan and Franco (2010), Jones et al. (2015), Ranjan and Spencer (2014) and the older references in Kleijnen (2008, p. 130).

In a case study, Helton et al. (2005) finds that crude Monte Carlo and LHS give similar results if these two methods use the same "big" sample size. In general, however, LHS is meant to improve results in simulation applications; see Janssen (2013).

There is much software for LHS. For example, Crystal Ball, @Risk, and Risk Solver provide LHS, and are add-ins to Microsoft's Excel spreadsheet software. LHS is also available in the MATLAB Statistics toolbox subroutine lhs and in the R package DiceDesign. We also mention Sandia's DAKOTA software:
http://dakota.sandia.gov/.

5.5.2 Sequential Customized Designs

The preceding designs for Kriging have a given number of input combinations n and consider only the input domain $\mathbf{x} \in \mathbb{R}^k$; i.e., these designs do not consider the output. Now we present designs that select n input combinations *sequentially* and consider the specific I/O function f_{sim} of the underlying simulation model so these designs are application-driven or *customized*. We notice that the importance of sequential sampling is also emphasized in Simpson et al. (2004), reporting on a panel discussion.

Note: Sequential designs for Kriging metamodels of deterministic simulation models are also studied in Busby et al. (2007), Crombecq et al. (2011), Koch et al. (2015), and Jin et al. (2002). Sequential LHDs ignoring the output (e.g., so-called "replicated LHDs") are discussed in Janssen (2013). Our sequential customized designs are no longer LHDs (even though the first stage may be a LHD), as we shall see next.

The designs discussed so far in this section, are *fixed sample* or *one shot* designs. Such designs suit the needs of experiments with real systems; e.g., agricultural experiments may have to be finished within a single growing season. Simulation experiments, however, proceed sequentially—unless parallel computers are used, and even then not the whole experiment is finished in one shot. In general, sequential statistical procedures are known to be more "efficient" in the sense that they require fewer observations than fixed-sample procedures; see, e.g., Ghosh and Sen (1991). In sequential designs we learn about the behavior of the underlying system as we experiment with this system and collect data. (The preceding chapter on screening also showed that sequential designs may be attractive in simulation.) Unfortunately, extra computer time is needed in sequential designs for Kriging if we re-estimate the Kriging parameters when new I/O data become available. Fortunately, computations may not start from scratch; e.g., we may initialize the search for the MLEs in the sequentially augmented design from the MLEs in the preceding stage.

Note: Gano et al. (2006) updates the Kriging parameters only when the parameter estimates produce a poor prediction. Toal et al. (2008) examines five update strategies, and concludes that it is bad not to update the estimates after the initial design. Chevalier and Ginsbourger (2012) presents formulas for updating the Kriging parameters and predictors for designs that add I/O data either purely sequential (a single new point with its output) or batch-sequential (batches of new points with their outputs). We shall also discuss this issue in Sect. 5.6 on SK.

Kleijnen and Van Beers (2004) proposes the following algorithm for specifying a customized sequential design for Kriging in deterministic simulation.

Algorithm 5.5

1. Start with a pilot experiment using some space-filling design (e.g., a LHD) with only a few input combinations; use these combinations as the input for the simulation model, and obtain the corresponding simulation outputs.

2. Fit a Kriging model to the I/O simulation data resulting from Step 1.

3. Consider (but do not yet simulate) a set of candidate combinations that have not yet been simulated and that are selected through some space-filling design; find the "winner", which is the candidate combination with the highest predictor variance.

4. Use the winner found in Step 3 as the input to the simulation model that is actually run, which gives the corresponding simulation output.

5. Re-fit (update) the Kriging model to the I/O data that is augmented with the I/O data resulting from Step 4.
   Comment: Step 5 refits the Kriging model, re-estimating the Kriging parameters $\boldsymbol{\psi}$; to save computer time, this step might not re-estimate $\boldsymbol{\psi}$.

6. Return to Step 3 until either the Kriging metamodel satisfies a given goal or the computer budget is exhausted.

Furthermore, Kleijnen and Van Beers (2004) compares this sequential design with a sequential design that uses the predictor variance with plugged -in parameters specified in Eq. (5.20). The latter design selects as the next point the input combination that maximizes this variance. It turns out that the latter design selects as the next point the input farthest away from the old input combinations, so the final design spreads all its points (approximately) evenly across the experimental area—like space-filling designs do. However, the predictor variance may also be estimated through *cross-validation* (we have already discussed cross-validation of Kriging models below Eq. (5.20)); see Fig. 5.6, which we discuss next.

Figure 5.6 displays an example with a fourth-order polynomial I/O function f_{sim} with two local maxima and three local minima; two minima occur at the border of the experimental area. Leave-one-out cross-validation means successive deletion of one of the n old I/O observations (which are already simulated), which gives the data set $(\mathbf{X}_{-i}, \mathbf{w}_{-i})$. $(i = 1, \ldots, n)$.

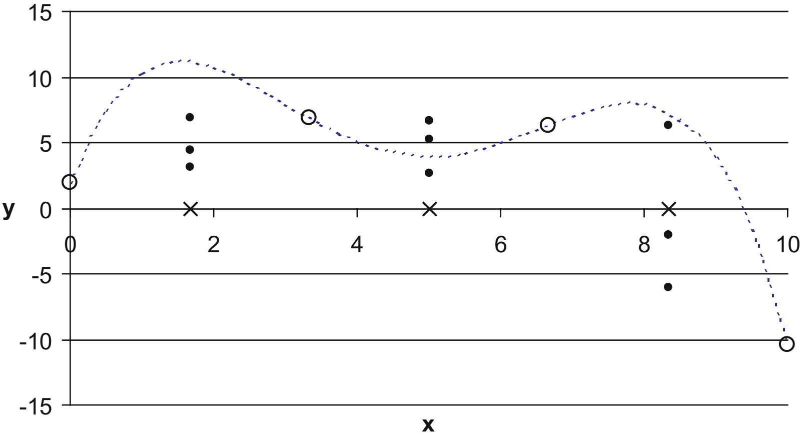


FIGURE 5.6. Cross-validation in fourth-order polynomial example with four pilot observations (see circles) and three candidate input values (see solid dots)

Next, we compute the Kriging predictor, after re-estimating the Kriging parameters. For each of three candidate points, the plot shows the three Kriging predictions computed from the original data set (no data deleted), and computed after deleting observation 2 and observation 3, respectively; the two extreme inputs ($x = 0$ and $x = 10$) are not deleted because Kriging does not extrapolate well. The point that is most difficult to predict turns out to be the candidate point $x = 8.33$ (the highest candidate point in the plot). To quantify this prediction uncertainty, we may jackknife the predictor variances, as follows.

In Sect. 3.3.3, we have already discussed jackknifing in general (jackknifing is also applied to stochastic Kriging, in Chen and Kim (2013)). Now, we calculate the jackknife's pseudovalue J for candidate point j as the weighted average of the original and the cross-validation predictors, letting c denote the number of candidate points and n the number of points already simulated and being deleted successively:

$$J_{j;i} = n\widehat{y}_j - (n-1)\widehat{y}_{j;-i} \quad \text{with} \quad j = 1, \ldots, c \text{ and } i = 1, \ldots, n.$$

From these pseudovalues we compute the classic variance estimator (also see Eq. (3.12)):

$$s^2(J_j) = \frac{\sum_{i=1}^{n}(J_{j;i} - \overline{J}_j)^2}{n(n-1)} \text{ with } \overline{J}_j = \frac{\sum_{i=1}^{n} J_{j;i}}{n}.$$

Figure 5.7 shows the candidate points that are selected for actual simulation. The pilot sample consists of four equally spaced points; also see

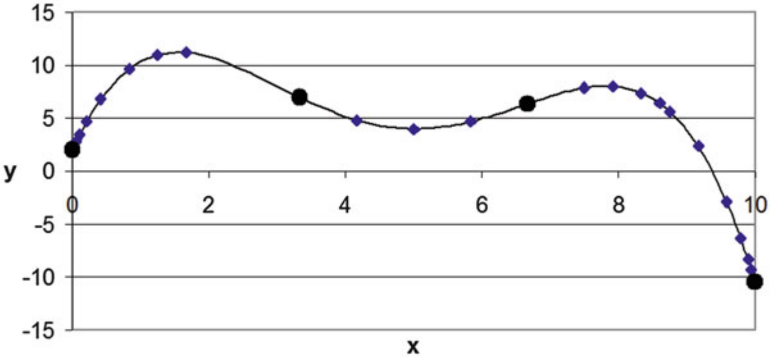


FIGURE 5.7. A fourth-degree polynomial example (see curve) of a sequential and customized design (see diamonds) and four initial points (see solid dots)

Fig. 5.6. The sequential design selects relative few points in subareas that generate an approximately linear I/O function; the design selects many points near the edges, where the function changes much. So the design favors points in subareas that have "more interesting" I/O behavior.

Note: Lin et al. (2002) criticizes cross-validation for the validation of Kriging metamodels, but in this section we apply cross-validation for the estimation of the prediction error when selecting the next design point in a customized design. Kleijnen and Van Beers (2004)'s method is also applied by Golzari et al. (2015).

5.6 Stochastic Kriging (SK) in Random Simulation

The interpolation property of Kriging is attractive in deterministic simulation, because the observed simulation output is unambiguous. In *random* simulation, however, the observed output is only one of the many possible values. Van Beers and Kleijnen (2003) replaces w_i (the simulation output at point i with $i = 1, \ldots, n$) by $\overline{w}_i = \sum_{r=1}^{m_i} w_{i;r}/m_i$ (the average simulated output computed from m_i replications). These averages, however, are still random, so the interpolation property loses its intuitive appeal. Nevertheless, Kriging may be attractive in random simulation because Kriging may decrease the predictor MSE at input combinations close together.

Note: Geostatisticians often use a model for (random) measurement errors that assumes a so-called *nugget effect* which is white noise; see Cressie (1993, pp. 59, 113, 128) and also Clark (2010). The Kriging predictor is then no longer an exact interpolator. Geostatisticians also study noise with heterogeneous variances; see Opsomer et al. (1999). In machine learning this problem is studied under the name *heteroscedastic GP regression*; see Kleijnen (1983) and our references in Sect. 5.1. Roustant et al. (2012) distinguishes between the nugget effect and homogeneous noise, such that the former gives a Kriging metamodel that remains an exact interpolator, whereas the latter does not. Historically speaking, Danie Krige worked in mining engineering and was confronted with the "nugget effect"; i.e., gold diggers may either miss the gold nugget "by a hair" or hit it "right on the head". Measurement error is a fundamentally different issue; i.e., when we measure (e.g.) the temperature on a fixed location, then we always get different values when we repeat the measurement at points of time "only microseconds apart", the "same" locations separated by nanomillimeters only, using different measurement tools or different people, etc.

In deterministic simulation, we may study *numerical* problems arising in Kriging. To solve such numerical noise, Lophaven et al. (2002, Eq. 3.16) and Toal et al. (2008) add a term to the covariance matrix Σ_M (also see Eq. (5.36) below); this term resembles the nugget effect, but with a "variance" that depends on the computer's accuracy.

Note: Gramacy and Lee (2012) also discusses the use of the nugget effect to solve numerical problems, but emphasizes that the nugget effect may also give better statistical performance such as better CIs. Numerical problems are also discussed in Goldberg et al. (1998), Harari and Steinberg (2014b), and Sun et al. (2014).

In Sect. 5.6.1 we discuss a metamodel for stochastic Kriging (SK) and its analysis; in Sect. 5.6.2 we discuss designs for SK.

5.6.1 A Metamodel for SK

In the analysis of random (stochastic) simulation models—which use pseudorandom numbers (PRNs)—we may apply SK, adding the *intrinsic noise* term $\varepsilon_r(\mathbf{x})$ for replication r at input combination $\mathbf{x}$ to the GP metamodel in Eq.(5.1) for OK with the *extrinsic noise*$M(\mathbf{x})$:

$$y_r(\mathbf{x}) = \mu + M(\mathbf{x}) + \varepsilon_r(\mathbf{x}) \quad \text{with} \quad \mathbf{x} \in \mathbb{R}^k \text{ and } r = 1, \ldots, m_i \tag{5.36}$$

where $\varepsilon_r(\mathbf{x})$ has a Gaussian distribution with zero mean and variance $\text{Var}[\varepsilon_r(\mathbf{x})]$ and is independent of the extrinsic noise $M(\mathbf{x})$. If the simulation does not use CRN, then $\mathbf{\Sigma}_\varepsilon$—the covariance matrix for the intrinsic noise—is diagonal with the elements $\text{Var}[\varepsilon(\mathbf{x})]$ on the main diagonal. If the simulation does use CRN, then $\mathbf{\Sigma}_\varepsilon$ is not diagonal; obviously, $\mathbf{\Sigma}_\varepsilon$ should still be symmetric and positive definite. (Some authors—e.g. Challenor (2013)—use the term "aleatory" noise for the intrinsic noise, and the term "epistemic noise" for the extrinsic noise in Kriging; we use these alternative terms in Chaps. 1 and 6.)

Averaging the m_i replications gives the average metamodel output $\overline{y}(\mathbf{x}_i)$ and average intrinsic noise $\overline{\varepsilon}(\mathbf{x}_i)$, so Eq. (5.36) is replaced by

$$\overline{y}(\mathbf{x}_i) = \mu + M(\mathbf{x}_i) + \overline{\varepsilon}(\mathbf{x}_i) \text{ with } \mathbf{x} \in \mathbb{R}^k \quad \text{and} \quad i = 1, \ldots, n. \tag{5.37}$$

Obviously, if we obtain m_i replicated simulation outputs for input combination i and we do not use CRN, then $\boldsymbol{\Sigma}_{\overline{\varepsilon}}$ is a diagonal matrix with main-diagonal elements $\mathrm{Var}[\varepsilon(\mathbf{x}_i)]/m_i$. If we do use CRN and m_i is a constant m, then $\boldsymbol{\Sigma}_{\overline{\varepsilon}} = \boldsymbol{\Sigma}_{\varepsilon}/m$ where $\boldsymbol{\Sigma}_{\varepsilon}$ is a symmetric positive definite matrix.

SK may use the classic estimators of $\mathrm{Var}[\varepsilon(\mathbf{x}_i)]$ using $m_i > 1$ replications, which we have already discussed in Eq. (2.27):

$$s^2(w_i) = \frac{\sum_{r=1}^{m_i}(w_{i;r} - \overline{w}_i)^2}{m_i - 1} \; (i = 1, \ldots n)$$

Instead of these point estimates of the intrinsic variances, SK may use another Kriging metamodel for the variances $\mathrm{Var}[\varepsilon(\mathbf{x}_i)]$—besides the Kriging metamodel for the mean $E[y_r(\mathbf{x}_i)]$— to predict the intrinsic variances. We expect this alternative to be less volatile than $s^2(w_i)$; after all, $s^2(w_i)$ is a *chi-square* variable (with $m_i - 1$ degrees of freedom) and has a large variance. Consequently, $s^2(w_i)$ is not normally distributed so the GP assumed for $s^2(w_i)$ is only a rough approximation. Because $s^2(w_i) \geq 0$, Goldberg et al. (1998) uses $\log[s^2(w_i)]$ in the Kriging metamodel. Moreover, we saw in Sect. 3.3.3 that a logarithmic transformation may make the variable normally distributed. We also refer to Kamiński (2015) and Ng and Yin (2012).

Note: Goldberg et al. (1998) assumes a known mean $E[y(\mathbf{x})]$, and a Bayesian approach using *Markov chain Monte Carlo* (MCMC) methods. Kleijnen (1983) also uses a Bayesian approach but no MCMC. Both Goldberg et al. (1998) and Kleijnen (1983) do not consider replications. Replications are standard in stochastic simulation; nevertheless, stochastic simulation without replication is studied in (Marrel et al. 2012). Risk and Ludkovski (2015) applies SK with estimated constant mean $\widehat{\mu}$ (like OK does) and mean function $f(\mathbf{x}; \widehat{\beta})$ (like UK does), and reports several case studies that give smaller MSEs for $f(\mathbf{x}; \widehat{\beta})$ than for $\widehat{\mu}$.

SK uses the OK predictor and its MSE replacing $\boldsymbol{\Sigma}_M$ by $\boldsymbol{\Sigma}_M + \boldsymbol{\Sigma}_{\overline{\varepsilon}}$ and $\mathbf{w}$ by $\overline{\mathbf{w}}$, so the SK predictor is

$$\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}}) = \widehat{\mu} + \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'(\widehat{\boldsymbol{\Sigma}}_M + \widehat{\boldsymbol{\Sigma}}_{\overline{\varepsilon}})^{-1}(\overline{\mathbf{w}} - \widehat{\mu}\mathbf{1}) \tag{5.38}$$

and its MSE is

$$\begin{aligned} \mathrm{MSE}[\widehat{y}(\mathbf{x}_0, \widehat{\boldsymbol{\psi}})] &= \widehat{\tau}^2 - \widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)'(\widehat{\boldsymbol{\Sigma}}_M + \widehat{\boldsymbol{\Sigma}}_{\overline{\varepsilon}})^{-1}\widehat{\sigma}(\mathbf{x}_0) \\ &\quad + \frac{[1 - \mathbf{1}'(\widehat{\boldsymbol{\Sigma}}_M + \widehat{\boldsymbol{\Sigma}}_{\overline{\varepsilon}})^{-1}\widehat{\boldsymbol{\sigma}}(\mathbf{x}_0)]^2}{\mathbf{1}'(\widehat{\boldsymbol{\Sigma}}_M + \widehat{\boldsymbol{\Sigma}}_{\overline{\varepsilon}})^{-1}\mathbf{1}}; \end{aligned} \tag{5.39}$$

also see Ankenman et al. (2010, Eq. 25).

The output of a stochastic simulation may be a *quantile* instead of an average (Eq. (5.37) does use averages). For example, a quantile may be relevant in chance-constrained optimization; also see Eq. (6.35) and Sect. 6.4 on robust optimization. Chen and Kim (2013) adapts SK for the latter type of simulation output; also see Bekki et al. (2014), Quadrianto et al. (2009), and Tan (2015).

Note: Salemi et al. (2014) assumes that the simulation inputs are integer variables, and uses a Gaussian Markov random field. Chen et al. (2013) allows some inputs to be qualitative, extending the approach for deterministic simulation in Zhou et al. (2011). Estimation of the whole density function of the output is discussed in Moutoussamy et al. (2014).

There is not much *software* for SK. The Matlab software available on the following web site is distributed "without warranties of any kind":

`http://www.stochastickriging.net/`.

The R package "DiceKriging" accounts for heterogeneous intrinsic noise; see Roustant et al. (2012). The R package "mlegp" is available on

`http://cran.r-project.org/web/packages/mlegp/mlegp.pdf`.

Software in C called PErK may also account for a nugget effect; see Santner et al. (2003, pp. 215–249).

In Sect. 5.3 we have already seen that ignoring the randomness of the estimated Kriging parameters $\widehat{\boldsymbol{\psi}}$ tends to underestimate the true variance of the Kriging predictor. To solve this problem in case of deterministic simulation, we may use *parametric bootstrapping* or its refinement called *conditional simulation*. (Moreover, the three variants—plugging-in $\widehat{\boldsymbol{\psi}}$, bootstrapping, or conditional simulation—may give predictor variances that reach their maxima for different new input combinations; these maxima are crucial in simulation optimization through "efficient global optimization", as we shall see in Sect. 6.3.1). In stochastic simulation, we obtain several replications for each old input combination—see Eq. (5.37)—so a simple method for estimating the true predictor variance uses *distribution-free bootstrapping*. We have already discussed the general principles of bootstrapping in Sect. 3.3.5. Van Beers and Kleijnen (2008) applies distribution-free bootstrapping assuming no CRN, as we shall see in the next subsection (Sect. 5.6.2). Furthermore. Yin et al. (2009) also studies the effects that the estimation of the Kriging parameters has on the predictor variance.

Note: Mehdad and Kleijnen (2015b) applies *stochastic intrinsic Kriging* (SIK), which is more complicated than SK. Experiments with stochastic simulations suggest that SIK outperforms SK.

To estimate the true variance of the SK predictor, Kleijnen and Mehdad (2015a) applies the Monte Carlo method, distribution-free bootstrapping, and parametric bootstrapping, respectively—using an M/M/1 simulation model for illustration.

5.6.2 Designs for SK

Usually SK employs the same designs as OK and UK do for deterministic simulation. So, SK often uses a one-shot design such as a *LHD*; also see Jones et al. (2015) and MacCalman et al. (2013).

However, besides the $n \times k$ matrix with the n design points $\mathbf{x}_i \in \mathbb{R}^k$ $(i = 1, \ldots, n)$ we need to select the *number of replications* m_i. In Sect. 3.4.5 we have already discussed the analogous problem for linear regression meta-

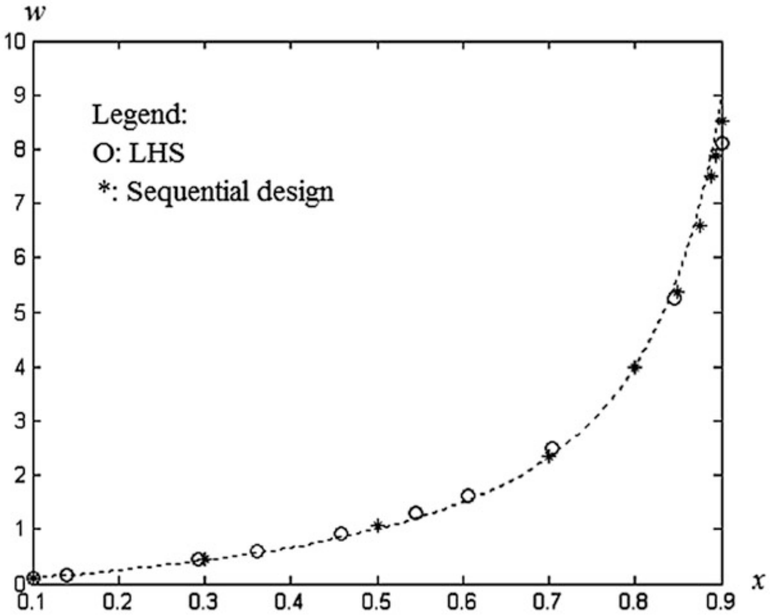


FIGURE 5.8. A LHD and a sequentialized design for the M/M/1 simulation with traffic rate $0 < x < 1$ and $n = 10$ points

models; a simple rule-of-thumb is to select m_i such that with $1 - \alpha$ probability the average output is within γ% of the true mean; see Eq. (3.30).

Note: For SK with heterogeneous intrinsic variances but without CRN (so Σ_ε is diagonal), Boukouvalas et al. (2014) examines *optimal designs* (which we also discussed for linear regression metamodels in Sect. 2.10.1). That article shows that designs that optimize the determinant of the so-called *Fisher information matrix* (FIM) outperform space-filling designs (such as LHDs), with or without replications. This FIM criterion minimizes the estimation errors of the GP covariance parameters (not the parameters $\boldsymbol{\beta}$ of the regression function $\mathbf{f}(\mathbf{x})'\beta$). That article recommends designs with at least two replications at each point; the optimal number of replications is determined through an optimization search algorithm. Furthermore, that article proposes the logarithmic transformation of the intrinsic variance when estimating a metamodel for this variance (we also discussed such a transformation in Sect. 3.4.3). Optimal designs for SK with homogeneous intrinsic variances (or a nugget effect) are also examined in Harari and Steinberg (2014a), and Spöck and Pilz (2015).

There are more complicated approaches. In sequential designs, we may use Algorithm 5.5 for deterministic simulation, but we change Step 3—which finds the candidate point with the highest predictor variance—such that we find this point through distribution-free bootstrapping based on replication, as we shall explain below. Figure 5.8 is reproduced from Van Beers and Kleijnen (2008); it displays a fixed LHS design with $n = 10$ values for the traffic rate x in an M/M/1 simulation with experimental area $0.1 \leq x \leq 0.9$, and a sequentialized design that is stopped after simulating the same number of observations (namely, $n = 10$). The plot shows that the sequentialized design selects more input values in the part of the input range that gives a drastically increasing (highly nonlinear) I/O function; namely $0.8 < x \leq 0.9$. It turns out that this design gives better Kriging predictions than the fixed LHS design does—especially for small designs, which are used in expensive simulations.

The M/M/1 simulation in Fig. 5.8 selects a run-length that gives a 95 % CI for the mean simulation output with a relative error of no more than 15 %. The sample size for the distribution-free bootstrap method is selected to be $B = 50$.

To estimate the predictor variance, Van Beers and Kleijnen (2008) uses *distribution-free bootstrapping* and treats the observed average bootstrapped outputs $\overline{w}_i^*$ $(i = 1, \ldots, n)$ as if they were the true mean outputs; i.e., the Kriging metamodel is an exact interpolator of $\overline{w}_i^*$ (obviously, this approach ignores the split into intrinsic and extrinsic noise that SK assumes).

Note: Besides the M/M/1 simulation, Van Beers and Kleijnen (2008) also investigates an (s, S) inventory simulation. Again, the sequentialized design for this (s, S) inventory simulation gives better predictions than a fixed-size (one-shot) LHS design; the sequentialized design concentrates its points in the steeper part of the response surface. Chen and Li (2014) also determines the number of replications through a relative precision requirement, but assumes linear interpolation instead of Kriging; that article also provides a comparison with the approach in Van Beers and Kleijnen (2008).

Note: Ankenman et al. (2010) does use the SK model in Eq. (5.36), and tries to find the design that allocates a fixed computer budget such that "new points" (input combinations not yet simulated) may be selected or additional replications for old points may be obtained. Chen and Zhou (2014) uses this approach, applying a variety of design criteria based on the MSE. Plumlee and Tuo (2014) also examines the number of replications in SK. Hernandez and Grover (2010) discusses sequential designs for Kriging metamodels of random simulation models; namely, models of so-called nanoparticles. Furthermore, Forrester (2013) recommends re-estimation of the Kriging hyperparameters $\boldsymbol{\psi}$, as the sequential design provides new I/O data. Kamiński (2015) gives various methods that avoid re-estimation of $\boldsymbol{\psi}$ in case of SK and sequential designs. Mehdad and Kleijnen (2015b) discusses sequential designs for *stochastic intrinsic Kriging* (SIK). More research on this issue is needed.

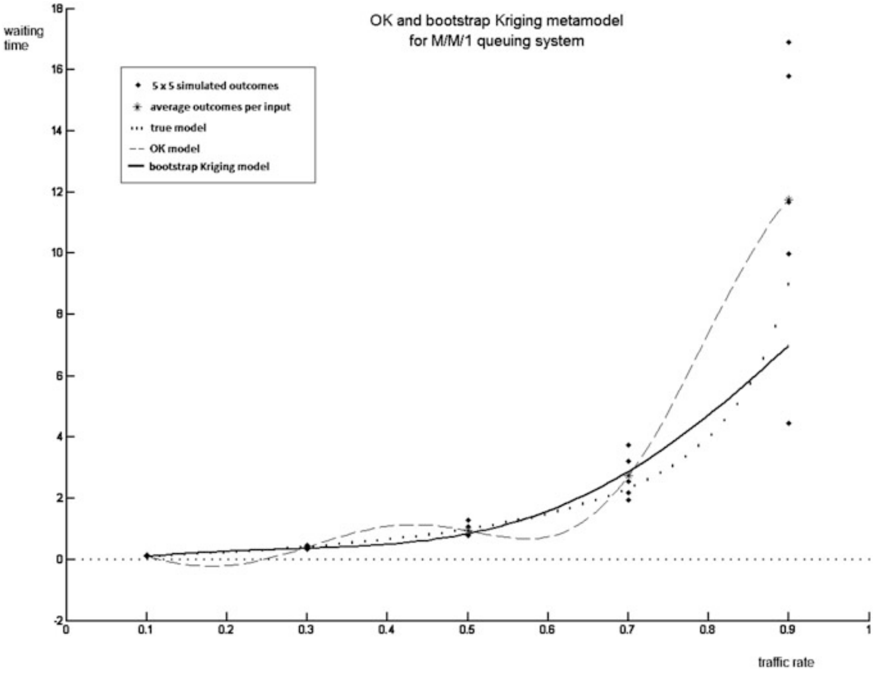


FIGURE 5.9. OK versus monotonic bootstrapped Kriging with acceptance/rejection, and the true I/O function for M/M/1 with $n = 5$ old input values and $m = 5$ replications

5.7 Monotonic Kriging: Bootstrapping and Acceptance/Rejection

In practice we sometimes know (or assume we know) that the I/O function implicitly specified by the simulation model is *monotonic*; e.g., if the traffic rate increases, then the mean waiting time increases. More examples are given in our chapter on screening (Chap. 4). We define a monotonic function as follows (as we also did in Definition 4.1):

Definition 5.1 *The function $w = f(x)$ is called monotonically increasing if $w(x = x_1) \leq w(x = x_2)$ if $x_1 \leq x_2$.*

The Kriging metamodel, however, may show a "wiggling" (erratic) I/O function, if the sample size is small; see the wiggly curve in Fig. 5.9. To make the Kriging predictor $\widehat{y}(x_j)$ $(j = 1, \ldots, k)$ a monotonic function of the input x_j, we propose *bootstrapping* with *acceptance/rejection*; i.e., we reject the Kriging metamodel fitted in bootstrap sample b—with $b = 1, \ldots, B$ and bootstrap sample size B—if this metamodel is not monotonic. In this section we summarize how Kleijnen and Van Beers (2013) uses distribution-free bootstrapping assuming stochastic simulation with replications for each input combination; at the end of this section, we shall briefly discuss parametric bootstrapping for deterministic simulation. (The general principles of distribution-free bootstrapping and parametric bootstrapping were discussed in Sect. 3.3.5.)

Note: Instead of bootstrapping, Da Veiga and Marrel (2012) solves the monotonicity problem and related problems analytically. However, their solution suffers from the curse of dimensionality; i.e., its *scalability* is questionable.

Kleijnen and Van Beers (2013) uses the popular DACE Matlab Kriging software, which is meant for deterministic simulation so it gives an exact interpolator. Bootstrapped Kriging, however, is not an exact interpolator for the original observations; i.e., its predictor $\widehat{y}^*(\mathbf{x}_i)$ for the n old input combinations $\mathbf{x}_i$ $(i = 1, \ldots, n)$ does not necessarily equal the n corresponding original average simulated outputs $\overline{w}_i = \sum_{r=1}^{m_i} w_{i;r}/m_i$ where m_i $(\gg 2)$ denotes the number of replications for input combination i. Actually, bootstrapped Kriging using DACE is an exact interpolator of the bootstrapped averages $\overline{w}_i^* = \sum_{r=1}^{m_i} w_{i;r}^*/m_i$, but not of $\overline{w}_i$. A CI is given by the well-known percentile method, now applied to the (say) B_a $(\leq B)$ accepted bootstrapped Kriging predictors $\widehat{y}_{b_a}^*(\mathbf{x})$ $(b_a = 1, \ldots, B_a)$.

More precisely, a monotonic predictor implies that the estimated *gradients* of the predictor remains positive as the inputs increase; we focus on monotonically increasing functions, because monotonically decreasing functions are a strictly analogous problem. An advantage of monotonic metamodeling is that the resulting sensitivity analysis is understood and accepted by the clients of the simulation analysts so these clients have more confidence in the simulation as a decision support tool. Furthermore, we shall see that monotonic Kriging gives smaller MSE and a CI with higher coverage and acceptable length. Finally, we conjecture that estimated gradients with correct signs will improve simulation optimization, discussed in the next chapter.

Technically speaking, we assume that no CRN are used so the number of replications may vary with the input combination $(m_i \neq m)$. Furthermore, we assume a Gaussian correlation function. We let $\mathbf{x}_i < \mathbf{x}_{i'}$ $(i, i' = 1, \ldots, n;$ $i \neq i')$ mean that at least one component of $\mathbf{x}_i$ is smaller than the corresponding component of $\mathbf{x}_{i'}$ and none of the remaining components is bigger. For example, the M/M/1 queueing simulation with the traffic rate x as the single input (so $k = 1$) implies that $\mathbf{x}_i < \mathbf{x}_{i'}$ becomes $x_i < x_{i'}$, whereas the (s, S) inventory simulation with the $k = 2$ inputs s and S implies that $\mathbf{x}_i < \mathbf{x}_{i'}$ may mean $s_i < s_{i'}$ and $S_i \leq S_{i'}$. The DACE software gives the estimated gradients $\nabla\widehat{y}(\mathbf{x})$, besides the prediction $\widehat{y}(\mathbf{x})$. We use a *test set* with v "new" points (in the preceding sections we denoted a single new point by

$\mathbf{x}_0$). We let $\lceil x \rceil$ denote the integer resulting from rounding x upwards, $\lfloor x \rfloor$ the integer resulting from rounding x downwards; the subscript () denotes the order statistics.

We propose the following algorithm (which adapts step 1 of Algorithm 5.2, and deviates only in its details but not in its overall goal from the algorithm in Kleijnen and Van Beers 2013); we assume that a 90 % CI is desired.

Algorithm 5.6

1. Resample the m_i original outputs $w_{i;r}$ $(i = 1, \ldots, n;\ r = 1, \ldots, m_i)$ with replacement, to obtain the bootstrapped output vectors $\mathbf{w}^*_{i;b} = (w^*_{i;r;b}, \ \ldots, \ w^*_{i;r;b})'$ $(b = 1, \ldots, B)$, which give $(\mathbf{X}, \overline{\mathbf{w}}^*_b)$ where $\mathbf{X}$ denotes the $n \times k$ matrix with the original n old combinations of the k simulation inputs and $\overline{\mathbf{w}}^*_b$ denotes the n-dimensional vector with the bootstrap averages $\overline{w}^*_{i;b} = \sum_{r=1}^{m_i} w^*_{i;r;b}/m_i$.

2. Use DACE to compute $\widehat{\boldsymbol{\psi}}^*_b$, the MLEs of the Kriging parameters $\boldsymbol{\psi}$ computed from the bootstrapped I/O data $(\mathbf{X}, \overline{\mathbf{w}}^*_b)$ of step 1.

3. Apply DACE using $(\mathbf{X}, \overline{\mathbf{w}}^*_b)$ of step 1 and $\widehat{\boldsymbol{\psi}}^*_b$ of step 2 to compute the Kriging predictor $\widehat{y}^*_b$ that interpolates so $\widehat{y}^*_b(\mathbf{x}_i) = \overline{w}^*_{i;b}$.

4. Accept the Kriging predictor $\widehat{y}^*_b$ of step 3 only if $\widehat{y}^*_b$ is monotonically increasing; i.e., all k components of the n gradients are positive:

$$\nabla \widehat{y}^*_{i;b'} > \mathbf{0} \qquad (i = 1, \ldots, n) \tag{5.40}$$

   where $\mathbf{0}$ denotes an n-dimensional vector with all elements equal to zero.

5. Use the B_a accepted bootstrapped Kriging metamodels resulting from step 4 to compute B_a predictions for v new points $\mathbf{x}_u$ $(u = 1, \ldots, v)$ with the point estimate equal to the sample median $\widehat{y}^*_{u;(\lceil 0.50 B_a \rceil)}$ and the two-sided 90 % CI equal to $[\widehat{y}^*_{u;(\lfloor 0.05 B_a \rfloor)}, \widehat{y}^*_{u;(\lceil 0.95 B_a \rceil)}]$.

If we find that step 5 gives a CI interval that is too wide, then we add more bootstrap samples so B increases and B_a probably increases too. For example, the M/M/1 simulation starts with $B = 100$ and augments B with 100 until either $B_a \geq 100$ or—to avoid excessive computational time—$B = 1{,}000$. This M/M/1 example has two performance measures; namely, the mean and the 90 % quantile of the steady-state waiting time distribution. Furthermore, the example illustrates both "short" and "long" simulation runs. Finally, $n = 5$ and $m_i = 5$ with $0.1 \leq x \leq 0.9$ and $v = 25$ new points; also see Fig. 5.9. This plot shows wiggling OK (so $d\widehat{y}/dx$ is negative for at least one x-value in the area of interest), whereas the bootstrap with acceptance/rejection gives monotonic predictions. This plot also shows—for each of the $n = 5$ input values—the $m = 5$ replicated simulation outputs (see dots) and their averages (see stars). Furthermore, the plot shows the analytical (dotted) I/O curve. Low traffic rates give such small variability of the individual simulation outputs that this variability is hardly visible; nevertheless, the bootstrap finds a monotonic Kriging model.

To quantify the performance of the preceding algorithm, we may use the *integrated mean squared error* (IMSE) defined in Sect. 2.10.1. To estimate the IMSE, we select v test points. If we let ζ_u $(u = 1, \ldots, v)$ denote the true output at test point u, then the *estimated integrated mean squared error* (EIMSE) MSE averaged over these v test points is the *estimated integrated MSE* (EIMSE) is

$$\text{EIMSE} = \frac{\sum_{u=1}^{v} (\widehat{y}^{*}_{u;(\lceil 0.50B' \rceil)} - \zeta_u)^2}{v}.$$

Note: We point out that a disadvantage of the IMSE criterion is that a high MSE at some point $\mathbf{x}_u$ can be "camouflaged" by a low MSE at some other point $\mathbf{x}_{u'}$ $(u \neq u')$.

Furthermore, OK uses the CI defined in Eq. (5.21). This CI is symmetric around its point estimate $\widehat{y}$ and may include negative values—even if negative values are impossible, as is the case for waiting times—whether it be the mean or the 90 % quantile.

A number of macroreplications (namely, 100) enable the estimation of the variance of the EIMSE estimate and the CI's coverage and width. These macroreplications show that this algorithm gives a smaller EIMSE than OK does, but this EIMSE is not significantly smaller. Of course, the EIMSE for the 90 % quantile is higher than the EIMSE for the mean. This algorithm also gives significantly higher estimated coverages, without widening the CI. Increasing n (number of old points) from 5 to 10 gives coverages close to the nominal 90 %—without significantly longer CIs—whereas OK still gives coverages far below the desired nominal value.

Besides using bootstrapped Kriging with acceptance/rejection to preserve monotonicity, we may also preserve other characteristics of the simulation I/O function; e.g., the Kriging predictions should not be *negative* for waiting times, variances, and thickness. Deutsch (1996) also investigates negative predictions in OK arising when some weights λ_i are negative (see again Sect. 5.2); also see

http://www.gslib.com/.

Furthermore, we may apply bootstrapping with acceptance/rejection to other metamodeling methods besides Kriging; e.g., *linear regression* (which we detailed in Chaps. 2 and 3).

If the simulation model is *deterministic*, then there are no replications so we may replace distribution-free bootstrapping by parametric bootstrapping assuming a multivariate Gaussian distribution as implied by a GP; also see Sect. 5.3.

Kleijnen et al. (2012) applies distribution-free bootstrapping with acceptance/rejection to find Kriging metamodels that preserve the assumed *convexity* of the simulation I/O function. Checking this convexity requires extending the DACE software to compute *Hessians*. Unfortunately, it turns out that this method does not give truly convex Kriging prediction functions. On hindsight, we may argue that in practice we do not really know whether the I/O function of the simulation model is convex; e.g., is the cost function of a realistic inventory-simulation model convex? We might assume that the simulation model has a unique optimal solution; convexity implies that the global and the local optima coincide. Da Veiga and Marrel (2012, p. 5) states: "Sometimes, the practitioner further knows that f (the I/O function) is convex at some locations, due to physical insight". Jian et al. (2014) develops a Bayesian approach for estimating whether a noisy function is convex.

5.8 Global Sensitivity Analysis: Sobol's FANOVA

So far we focused on the *predictor* $\widehat{y}(\mathbf{x})$, but now we discuss *sensitivity analysis* (SA) measuring how sensitive the simulation output w is to the individual inputs x_1 through x_k and their interactions. Such an analysis may help us to understand the underlying simulation model; i.e., SA may help us to find the important simulation inputs. In the three previous chapters we used polynomials of first order or second order to approximate the simulation I/O function $w = f_{\text{sim}}(\mathbf{x})$, so the regression parameters $\boldsymbol{\beta}$ quantify the first-order and second-order effects of the inputs. OK gives a more complicated approximation; namely, Eq. (5.1) including the extrinsic noise term $M(\mathbf{x})$ which makes y a *nonlinear* function of $\mathbf{x}$. To quantify the importance of the inputs of the simulation model—possibly approximated through a metamodel—we now apply so-called *functional analysis of variance* (FANOVA). This analysis uses variance-based indexes that were originally proposed by the Russian mathematician Sobol; see Sobol (1990) and the references in Archer et al. (1997).

FANOVA decomposes the variance of the simulation output w into fractions that refer to the individual inputs or to sets of inputs; e.g., FANOVA may show that 70 % of the output variance is caused by the variance in x_1, 20 % by the variance in x_2, and 10 % by the interaction between x_1 and x_2. As we have already seen in Sect. 5.5.1, we assume that the input $\mathbf{x}$ has a prespecified (joint) distribution (which may the product of k marginal distributions). Below Eq. (5.13) we stated that θ_j denotes the importance of x_j. However, the importance of x_j is much better quantified through FANOVA, which also measures interactions—as we shall see in this section.

It can be proven that the following variance decomposition—into a sum of 2^{k-1} components—holds:

$$\sigma_w^2 = \sum_{j=1}^{k} \sigma_j^2 + \sum_{j<j'}^{k} \sigma_{j;j'}^2 + \ldots + \sigma_{1;\ldots;k}^2 \tag{5.41}$$

with the main-effect (first order) variance

$$\sigma_j^2 = \mathrm{Var}[E(w|x_j)] \tag{5.42}$$

and the two-factor interaction variance

$$\sigma_{j;j'}^2 = \mathrm{Var}[E(w|x_j, x_{j'})]$$

and so on, ending with the k-factor interaction variance

$$\sigma_{1;\ldots;k}^2 = \mathrm{Var}[E(w|x_1, \ldots, x_k)]. \tag{5.43}$$

In Eq. (5.42) $E(w|x_j)$ denotes the mean of w if x_j is kept fixed while all $k-1$ remaining inputs $\mathbf{x}_{-j} = (\ldots, x_{j-1}, x_{j+1}, \ldots)'$ do vary. If x_j has a "large" main effect, then $E(w|x_j)$ changes much as x_j changes. Furthermore, Eq. (5.42) shows $\mathrm{Var}[E(w|x_j)$, which is the variance of $E(w|x_j)$ if x_j varies; so if x_j has a large main effect, then $\mathrm{Var}[E(w|x_j)$ is high if x_j varies. We point out that in Eq. (5.43) $\mathrm{Var}[E(w|x_1, \ldots, x_k)]$ denotes the variance of the mean of w if all k inputs are fixed; consequently, this variance is zero in deterministic simulation, and equals the intrinsic noise in stochastic simulation (the intrinsic noise in stochastic simulation may vary with $\mathbf{x}$, as we saw in Sect. 5.6).

The measure σ_j^2 defined in Eq. (5.42) leads to the following variance-based measure of importance, which the FANOVA literature calls the *first-order sensitivity index* or the *main effect index* and which we denote by γ (we use Greek letters for parameters, throughout this book):

$$\gamma_j = \frac{\sigma_j^2}{\sigma_w^2}.$$

So, γ_j quantifies the effect of varying x_j alone—averaged over the variations in all the other $k-1$ inputs; σ_w^2 in the denominator standardizes γ_j to provide a fractional contribution (in linear regression analysis we standardize the inputs x_j so that β_j measures the relative main effect; see Sect. 2.3.1). The interaction indices $\sigma_{j;j'}^2$ through $\sigma_{1;\ldots;k}^2$ are also divided by σ_w^2. The result of this standardization is the following equation:

$$\sum_{j=1}^{k} \gamma_j + \sum_{j=1}^{k-1} \sum_{j'=j+1}^{k} \gamma_{j;j'} + \ldots + \gamma_{1;\ldots;k} = 1. \tag{5.44}$$

As k increases, the number of measures in Eqs. (5.41) or (5.44) increases dramatically; actually, this number is $2^k - 1$ (as we know from classic ANOVA). The estimation of all these measures may require too much computer time, as we shall see below. Moreover, such a large number of measures may be hard to interpret; also see Miller (1956). So—as we did in the immediately preceding three chapters—we might assume that only the first-order measures γ_j—and possibly the second-order measures $\gamma_{j;j'}$—are important, and verify whether they sum up to a fraction "close enough" to 1 in Eq. (5.44); i.e., do they contribute the major part of the total variance σ_w^2?

Alternatively, we might compute the *total-effect index* or *total-order index* (say) $\gamma_{j;-j}$, which measures the contribution to σ_w^2 due to x_j including all variance caused by all the interactions between x_j and any other input variables $\mathbf{x}_{-j}$:

$$\gamma_{j;-j} = \frac{E[\mathrm{Var}(w|\mathbf{x}_{-j})]}{\sigma_w^2} = 1 - \frac{\mathrm{Var}[E(w|\mathbf{x}_{-j})]}{\sigma_w^2}.$$

It can be proven that $\sum_{j=1}^{k} \gamma_{j;-j} \geq 1$—unless there are only first-order effects—because the interaction effect between (say) x_j and $x_{j'}$ is counted in both $\gamma_{j;-j}$ and $\gamma_{j';-j'}$.

The *estimation* of the various sensitivity measures uses Monte Carlo methods. We may improve the accuracy of the estimators, replacing the "crude" Monte Carlo method by *quasi-Monte Carlo* methods, such as LHS and Sobol sequences (which we discussed in Sect. 5.5). To save computer time, we may replace the simulation model by a metamodel such as an OK model (with a specific correlation function; e.g., the Gaussian function).

Note: Details are given in Saltelli et al. (2008, pp. 164–67); also see Fang et al. (2006, pp. 31–33, 193–202), Helton et al. (2006b), Le Gratiet and Cannamela (2015), and Saltelli et al. (2010). The method in Le Gratiet and Cannamela (2015) is available in the package "sensitivity" (linked to the R package DiceKriging).

Note: FANOVA is the topic of much current research; see Anderson et al. (2014), Borgonovo and Plischke (2015), Farah and Kottas (2014), Ginsbourger et al. (2015), Henkel et al. (2012), Jeon et al. (2015), Lamboni et al. (2013), Marrel et al. (2012), Muehlenstaedt et al. (2012), Owen et al. (2013), Quaglietta (2013), Razavi and Gupta (2015), Shahraki and Noorossana (2014), Storlie et al. (2009), Tan (2014a), Tan (2014b), Tan (2015), Wei et al. (2015), and Zuniga et al. (2013).

5.9 Risk Analysis

In the preceding section on global sensitivity analysis through FANOVA we assumed that the input $\mathbf{x} \in \mathbb{R}^k$ has a given (joint) distribution. This assumption implies that even a *deterministic* simulation model gives a random output w; by definition, a stochastic simulation model always gives a random output. In *risk analysis* (RA) or *uncertainty analysis* (UA) we may wish to estimate $P(w > c)$, which denotes the probability of the output w exceeding a given threshold value c. RA is applied in nuclear engineering, finance, water management, etc. A probability such as $P(w > c)$ may be very small—so $w > c$ is called a *rare event*—but may have disastrous consequences (we may then apply "importance sampling"; see Kleijnen et al. 2013). In Sect. 1.1 we have already discussed the simple Example 1.1 with the net present value (NPV) as output and the discount factor or the cash flows as uncertain inputs, so the input values are sampled from given distribution functions; spreadsheets are popular software for such NPV computations.

Note: Borgonovo and Plischke (2015) applies FANOVA to inventory management models—such as the economic order quantity (EOQ) model—with uncertain inputs. We also refer to the publications that we gave in Sect. 1.1; namely, Evans and Olson (1998) and Vose (2000). Another type of deterministic simulation is used in project planning through the *critical path method* (CPM) and *program evaluation and review technique* (PERT), which in RA allows for uncertain durations of the project components so these durations are sampled from beta distributions; see Lloyd-Smith et al. (2004). More examples of RA are given in Kleijnen (2008, p. 125); also see Helton et al. (2014).

The uncertainty about the exact values of the input values is called *subjective* or *epistemic*, whereas the "intrinsic" uncertainty in stochastic simulation (see Sect. 5.6) is called *objective* or *aleatory*; see Helton et al. (2006a). There are several methods for obtaining subjective distributions for the input $\mathbf{x}$ based on *expert opinion*.

Note: Epistemic and aleatory uncertainties are also discussed in Barton et al. (2014), Batarseh and Wang (2008), Callahan (1996), De Rocquigny et al. (2008), Helton et al. (2010), Helton and Pilch (2011), and Xie et al. (2014).

We emphasize that the goals of RA and SA do differ. SA tries to answer the question "Which are the most important inputs in the simulation model of a given real system?", whereas RA tries to answer the question "What is the probability of a given (disastrous) event happening?". We have already seen designs for SA that uses low-order polynomials (which are a type of linear regression metamodels) in the immediately preceding three chapters; designs for RA are samples from the given distribution of the input $\mathbf{x}$ through Monte Carlo or quasi-Monte Carlo methods, as we discussed in the preceding section on FANOVA (Sect. 5.8). SA identifies those inputs for which the distribution in RA needs further refinement.

Note: Similarities and dissimilarities between RA and SA are further discussed in Kleijnen (1983, 1994, 1997), Martin and Simpson (2006), Norton (2015), Oakley and O'Hagan (2004), and Song et al. (2014).

We propose the following algorithm for RA with the goal of estimating $P(w > c)$.

Algorithm 5.7

1. Use a Monte Carlo method to sample input combination $\mathbf{x}$ from its given distribution.
   Comment: If the inputs are independent, then this distribution is simply the product of the marginal distributions.

2. Use $\mathbf{x}$ of step 1 as input into the given simulation model.
   Comment: This simulation model may be either deterministic or stochastic.

3. Run the simulation model of step 2 to transform the input $\mathbf{x}$ of step 2 into the output w.
   Comment: This run is called "propagation of uncertainty".

4. Repeat steps 1 through 3 n times to obtain the estimated distribution function (EDF) of the output w.

5. Use the EDF of step 4 to estimate the required probability $P(w > c)$.

Exercise 5.7 *Perform a RA of an M/M/1 simulation, as follows. Suppose that you have available* m *IID observations on the interarrival time, and on the service time, respectively, denoted by* a_i *and* s_i $(i = 1, \ldots, m)$*. Actually, you sample these values from exponential distributions with parameter* $\lambda = \rho$ *and* $\mu = 1$ *where* ρ *is the traffic rate that you select. Resample with replacement (i.e., use distribution-free bootstrapping) to obtain* m *interarrival times and* m *service times, which you use to estimate the arrival and service rates* λ *and* μ*. Use this pair of estimated rates as input to your M/M/1 simulation. In this simulation, you observe the output that you are interested in (e.g., the estimated steady-state mean waiting time). Perform* M *macroreplications, to estimate the aleatory uncertainty. Repeat the bootstrapping, to find different values for the pair of estimated rates; again simulate the M/M/1 system to estimate the epistemic uncertainty. Compare the effects of both types of uncertainty.*

Because (by definition) an *expensive* simulation model requires much computer time per run, we may perform RA as follows: do not run n simulation runs (see steps 3 and 4 in the preceding algorithm), but run its *metamodel* n times. For example, Giunta et al. (2006) uses crude Monte Carlo, LHS, and orthogonal arrays to sample from two types of metamodels—namely, Kriging and multivariate adaptive regression splines (MARS)—and finds that the true mean output can be better estimated through inexpensive sampling of many values from the metamodel, which is estimated from relatively few I/O values obtained from the expensive simulation model (because that publication estimates an expected value, it does not perform a true RA). Another example is Martin and Simpson (2006), using a Kriging metamodel to assess output uncertainty. Furthermore, Barton et al. (2014)

uses bootstrapping and stochastic Kriging (SK) to obtain a CI for the mean output of the real system. Another interesting article on RA is Lemaître et al. (2014). The British research project called *Managing uncertainty in complex models* (MUCM) also studies uncertainty in simulation models, including uncertainty quantification, uncertainty propagation, risk analysis, and sensitivity analysis; see

`http://www.mucm.ac.uk`.

Related to MUCM is the "Society for Industrial and Applied Mathematics (SIAM)"'s "Conference on Uncertainty Quantification (UQ16)", held in cooperation with the "American Statistical Association (ASA)" and the "Gesellschaft für Angewandte Mathematik und Mechanik (GAMM)"'s "Activity Group on Uncertainty Quantification (GAMM AG UQ)", in Lausanne (Switzerland), 5–8 April 2016; see

`http://www.siam.org/meetings/uq16/`.

We shall return to uncertainty in the input $\mathbf{x}$ in the next chapter, in which we discuss robust optimization (which accounts for the uncertainty in some of the inputs); see Sect. 6.4.

Chevalier et al. (2013) and Chevalier et al. (2014) use a Kriging metamodel to estimate the *excursion set* defined as the set of inputs—of a deterministic simulation model—resulting in an output that exceeds a given threshold, and quantifies uncertainties in this estimate; a sequential design may reduce this uncertainty. Obviously, the volume of the excursion set is closely related to the *failure probability* $P(w > c)$ defined in the beginning of this section. Kleijnen et al. (2011) uses a first-order polynomial metamodel (instead of a Kriging metamodel) to estimate which combinations of uncertain inputs form the frontier that separates acceptable and unacceptable outputs; both aleatory uncertainty—characteristic for random simulation—and epistemic uncertainty are included.

Note: Stripling et al. (2011) creates a "manufactured universe" (namely, a nuclear "particle-transport universe") that generates data on which a simulation model may be built; next, this simulation model generates data to which a metamodel is fitted. This metamodel produces predictions, which may be compared to the true values in the manufactured universe. We may compare this approach with the Monte Carlo experiment in Exercise 5.7, in which the manufactured universe is an M/M/1 system and the metamodel is a SK model; actually, we may use an M/G/1 system—where G stands for general service time distribution (e.g., a lognormal distribution)—and the simulator builds an M/M/1 simulation model with exponential arrival and service parameters estimated from the data generated by the M/G/1 system, so model errors are made besides estimation errors.

RA is related to the *Bayesian* approach, as the latter approach also assumes that the parameters of the simulation model are unknown and assumes given "prior" distributions for these parameters. The Bayesian paradigm selects these prior distributions in a more formal way (e.g., it selects so-called conjugate priors), obtains simulation I/O data, and *calibrates* the metamodel's parameters; i.e., it computes the posterior distribution (or likelihood) using the well-known Bayes theorem. *Bayesian model averaging* and *Bayesian melding* formally account—not only for the uncertainty of the input parameters—but also for the uncertainty in the form of the (simulation) model itself. The Bayesian approach is very interesting, especially from an academic point of view; practically speaking, however, classic frequentist RA has been applied many more times. References to the Bayesian approach are given in Kleijnen (2008, p. 126); also see "Bayesian model averaging" in Wit et al. (2012) and the specific Bayesian approach in Xie et al. (2014).

Note: We present a methodology that treats the simulation model as a black box, so this methodology can be applied to any simulation model. A disadvantage, however, is that this methodology cannot make use of knowledge about the specific model under discussion; e.g., Bassamboo et al. (2010) uses knowledge about specific call-center queueing models, when examining epistemic and aleatory uncertainties.

5.10 Miscellaneous Issues in Kriging

Whereas we focussed on Kriging metamodels for the *mean* simulation output in the preceding sections, Plumlee and Tuo (2014) examines Kriging metamodels for a fixed *quantile* (e.g., the 90 % quantile) of the random simulation output. Jala et al. (2014) uses Kriging to estimate a quantile of a deterministic simulation with random input (which results in uncertainty propagation, as we saw in Sect. 5.9). In Sect. 5.6.1 we have already mentioned that Chen and Kim (2013) adapts SK for quantiles, and we have also referred to Bekki et al. (2014), Quadrianto et al. (2009), and Tan (2015).

Another issue is *multivariate Kriging*, which may be applied in *multi-fidelity* metamodeling; i.e., we use several simulation models of the same real system, and each model has its own degree of detail representing the real system. Obviously, the various simulation models give external noises $M(\mathbf{x})$ that are correlated. An example in *finite element modeling* (FEM) is the use of different simulation models with different meshes (grids). However, we are not aware of much multi-fidelity modeling in discrete-event simulation; however, Xu et al. (2015) does discuss multifidelity in such simulation.

Note: Multi-fidelity metamodeling is further discussed in Couckuyt et al. (2014), Koziel et al. (2014), Le Gratiet and Cannamela (2015), Razavi et al. (2012), Tuo et al. (2014), and Viana et al. (2014, Section III).

We may also combine the output of a simulation model with the output of the real system, so-called *field data*. For such problems Goh et al. (2013) uses a Bayesian approach.

In practice, a discrete-event simulation model usually produces *multiple responses*, which have intrinsic noises $\varepsilon(\mathbf{x})$ that are correlated because these outputs are (different) functions of the same PRNs. For such a simulation model we might use a multivariate Kriging metamodel. However, Kleijnen and Mehdad (2014) finds that we might as well apply univariate Kriging to each type of simulation response separately. Notice that FANOVA for multivariate Kriging is examined in Zhang (2007) and Zhang et al. (2007). Li and Zhou (2015) considers multivariate GP metamodels for deterministic simulation models with multiple output types.

We may combine Kriging metamodels, each with a different type of correlation function (e.g., Gaussian and exponential) in an *ensemble*; see Harari and Steinberg (2014b), Viana et al. (2014, Figure 5), and the other references in Sect. 1.2.

We may partition the input domain $\mathbf{x} \in \mathbb{R}^k$ into subdomains, and fit a separate GP model within each subdomain; these subdomains may be determined through *classification and regression trees* (CART); for CART we also refer to Chap. 1. Gramacy and Lee (2008) speak of a *treed Gaussian process*. An R package for treed GPs is available on

`http://users.soe.ucsc.edu/~rbgramacy/tgp.html`.

Another issue in Kriging is the *validation* of Kriging metamodels. In deterministic simulation we may proceed analogously to our validation of linear regression metamodels in deterministic simulation, discussed in Sect. 3.6; i.e., we may compute the coefficients of determination R^2 and R^2_{adj}, and apply cross-validation (as we also did in Fig. 5.6). We also refer to the free R package DiceEval; see

`http://cran.r-project.org/web/packages/DiceEval/index.html`.

Scatterplots with $(w_i, \widehat{y}_i)$—not $(w_i, \widehat{y}_{-i})$ as in cross-validation—are used in many deterministic simulations; an example is the climate simulation in Hankin (2005). The validation of Kriging metamodels is also discussed in Bastos and O'Hagan (2009), following a Bayesian approach. An interesting issue in cross-validation is the fast re-computation of the Kriging model (analogous to the shortcut in Eq. (3.50) for linear regression that uses the hat matrix); also see Hubert and Engelen (2007), discussing fast cross-validation for principle component analysis (PCA).

For deterministic simulations Challenor (2013) and Iooss et al. (2010) examine LHDs with an extra criterion based on the distances between the points in the original and the validation designs (so no cross-validation is applied).

A final issue in Kriging is the variant that Salemi et al. (2013) introduces; namely, *generalized integrated Brownian fields* (GIBFs). Related to these GIBFs are the *intrinsic random functions* that Mehdad and Kleijnen (2015b) introduces into Kriging metamodeling of deterministic and stochastic simulation models, as we have already seen in Sect. 5.4.

5.11 Conclusions

In this chapter we started with an introduction of Kriging and its application in various scientific disciplines. Next we detailed OK for deterministic simulation. For the unbiased estimation of the variance of the OK predictor with estimated Kriging parameters we discussed parametric bootstrapping and conditional simulation. Next we discussed UK for deterministic simulation. Then we surveyed designs for Kriging metamodels, focusing on one-shot standardized LHS and sequentialized, customized designs. We continued with SK for random simulation. To preserve the monotonicity of the I/O function, we proposed bootstrapping with acceptance/rejection. Next we discussed FANOVA using Sobol's sensitivity indexes. Furthermore we discussed RA. Finally, we discussed several remaining issues. Throughout this chapter we also mentioned issues requiring further research.

Solutions of Exercises

Solution 5.1 $E(y|w_1 > \mu, w_2 = \mu, \ldots, w_n = \mu) > \mu$ *because* $\boldsymbol{\sigma}(x_0')\boldsymbol{\Sigma}^{-1} > \mathbf{0}'$.

Solution 5.2 *In general* $\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1} = \mathbf{I}$. *If* $\mathbf{x}_0 = \mathbf{x}_i$, *then* $\boldsymbol{\sigma}(x_0)$ *is a vector of* $\boldsymbol{\Sigma}$. *So* $\boldsymbol{\sigma}(x_0)'\boldsymbol{\Sigma}^{-1}$ *equals a vector with* $n-1$ *zeroes and one element with the value one. So* $\boldsymbol{\sigma}(\mathbf{x}_0)'\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\mu\mathbf{1})$ *reduces to* $w_i - \mu$. *Finally,* $\widehat{y}(\mathbf{x}_0|\mathbf{w})$ *becomes* $\mu + (w_i - \mu) = w_i$.

Solution 5.3 *If* $\mathbf{x}_0 = \mathbf{x}_1$, *then* $\lambda_1 = 1$ *and* $\lambda_2 = \ldots = \lambda_n = 0$ *(because* $\widehat{y}(\mathbf{x}_0)$ *is an exact interpolator), so* $\mathrm{Var}[\widehat{y}(\mathbf{x}_0)] = 2\mathrm{cov}(y_1, y_1) - [\mathrm{cov}(y_1, y_1) + \mathrm{cov}(y_1, y_1)] = 0$.

Solution 5.4 *When* $h = 0$, *then* $\rho = 1/\exp(0) = 1/1 = 1$. *When* $h = \infty$, *then* $\rho = 1/\exp(\infty) = 1/\infty = 0$.

Solution 5.5 *When input* j *has no effect on the output, then* $\theta_j = \infty$ *in Eq. (5.13) so the correlation function drops to zero.*

Solution 5.6 *As* n *(number of old points) increases, the new point has neighbors that are closer and have outputs that are more correlated with the output of the new point. So the length of the CI decreases.*

Solution 5.7 *The results depend on your choice of the parameters of this Monte Carlo experiment; e.g., the parameter* m.

References

Anderson B, Borgonovo E, Galeotti M, Roson R (2014) Uncertainty in climate change modeling: can global sensitivity analysis be of help? Risk Anal 34(2):271–293

Ankenman B, Nelson B, Staum J (2010) Stochastic kriging for simulation metamodeling. Oper Res 58(2):371–382

Antognini B, Zagoraiou M (2010) Exact optimal designs for computer experiments via kriging metamodelling. J Stat Plan Inference 140(9):2607–2617

Archer GEB, Saltelli A, Sobol IM (1997) Sensitivity measures, ANOVA-like techniques and the use of bootstrap. J Stat Comput Simul 58:99–120

Ba S, Brenneman WA, Myers WR (2014) Optimal sliced Latin hypercube designs. Technometrics (in press)

Bachoc F (2013) Cross validation and maximum likelihood estimation of hyper-parameters of Gaussian processes with model misspecification. Comput Stat Data Anal 66:55–69

Barton RR, Nelson BL, Xie W (2014) Quantifying input uncertainty via simulation confidence intervals. INFORMS J Comput 26(1):74–87

Bassamboo A, Randhawa RS, Zeevi A (2010) Capacity sizing under parameter uncertainty: safety staffing principles revisited. Manag Sci 56(10):1668–1686

Bastos LS, O'Hagan A (2009) Diagnostics for Gaussian process emulators. Technometrics 51(4):425–438

Batarseh OG, Wang Y (2008) Reliable simulation with input uncertainties using an interval-based approach. In: Mason SJ, Hill RR, Mönch L, Rose O, Jefferson T, Fowler JW (eds) Proceedings of the 2008 winter simulation conference, Miami, pp 344–352

Bekki J, Chen X, Batur D (2014) Steady-state quantile parameter estimation: an empirical comparison of stochastic kriging and quantile regression. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 Winter Simulation Conference, Savannah, pp 3880–3891

Borgonovo E, Plischke E (2015) Sensitivity analysis: a review of recent advances. Eur J Oper Res (in press)

Borgonovo E, Tarantola S, Plischke E, Morris MD (2014) Transformations and invariance in the sensitivity analysis of computer experiments. J R Stat Soc, Ser B 76:925–947

Boukouvalas A, Cornford D, Stehlík M (2014) Optimal design for correlated processes with input-dependent noise. Comput Stat Data Anal 71:1088–1102

Bowman VE, Woods DC (2013) Weighted space-filling designs. J Simul 7:249–263

Busby D, Farmer CL, Iske A (2007) Hierarchical nonlinear approximation for experimental designs and statistical data fitting. SIAM J Sci Comput 29(1):49–69

Butler A, Haynes RD, Humphriesa TD, Ranjan P (2014) Efficient optimization of the likelihood function in Gaussian process modelling. Comput Stat Data Anal 73:40–52

Callahan BG (ed) (1996) Special issue: commemoration of the 50th anniversary of Monte Carlo. Hum Ecol Risk Assess 2(4):627–1037

Challenor P (2013) Experimental design for the validation of Kriging metamodels in computer experiments. J Simul (7):290–296

Chen EJ, Li M (2014) Design of experiments for interpolation-based metamodels. Simul Model Pract Theory 44:14–25

Chen VCP, Tsui K-L, Barton RR, Meckesheimer M (2006) A review on design, modeling, applications of computer experiments. IIE Trans 38:273–291

Chen X, Ankenman B, Nelson BL (2012) The effects of common random numbers on stochastic Kriging metamodels. ACM Trans Model Comput Simul 22(2):7:1–7:20

Chen X, Kim K-K (2013) Building metamodels for quantile-based measures using sectioning. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 winter simulation conference, Washington, DC, pp 521–532

Chen X, Wang K, Yang F (2013) Stochastic kriging with qualitative factors. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 winter simulation conference, Washington, DC, pp 790–801

Chen X, Zhou Q (2014) Sequential experimental designs for stochastic kriging. In: Tolk A, Diallo SD, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3821–3832

Chevalier C, Ginsbourger D (2012) Corrected Kriging update formulae for batch-sequential data assimilation. arXiv, 1203.6452v1

Chevalier C, Ginsbourger D, Bect J, Molchanov I (2013) Estimating and quantifying uncertainties on level sets using the Vorob'ev expectation and deviation with Gaussian process models. In: Ucinski D, Atkinson AC, Patan M (eds) mODa 10 – advances in model-oriented design and analysis; proceedings of the 10th international workshop in model-oriented design and analysis. Springer, New York, pp 35–43

Chevalier C, Ginsbourger D, Bect J, Vazquez E, Picheny V, Richet Y (2014) Fast parallel Kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. Technometrics 56(4): 455–465

Chilès J-P, Delfiner P (2012) Geostatistics: modeling spatial uncertainty, 2nd edn. Wiley, New York

Clark I (2010) Statistics or geostatistics? Sampling error or nugget effect? J S Afr Inst Min Metall 110:307–312

Couckuyt I, Dhaene T, Demeester P (2014) ooDACE toolbox: a flexible object-oriented Kriging implementation. J Mach Learn Res 15:3183–3186

Couckuyt I, Forrester A, Gorissen D, Dhaene T (2012) Blind kriging; implementation and performance analysis. Adv Eng Softw 49:1–13

Cressie NAC (1993) Statistics for spatial data, rev edn. Wiley, New York

Crombecq K, Laermans E, Dhaene T (2011) Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling. Eur J Oper Res 214:683–696

Damblin G, Couplet M, Iooss B (2013) Numerical studies of space-filling designs: optimization of Latin hypercube samples and subprojection properties.J Simul 7:276–289

Da Veiga S, Marrel A (2012) Gaussian process modeling with inequality constraints. Annales de la faculté des sciences de Toulouse Sér. 6 21(3):529–555

De Rocquigny E, Devictor N, Tarantola S (2008) Uncertainty settings and natures of uncertainty. In: de Rocquigny E, Devictor N, Tarantola S (eds) Uncertainty in industrial practice. Wiley, Chichester

Den Hertog D, Kleijnen JPC, Siem AYD (2006) The correct Kriging variance estimated by bootstrapping. J Oper Res Soc 57(4):400–409

Deng H, Shao W, Ma Y, Wei Z (2012) Bayesian metamodeling for computer experiments using the Gaussian Kriging models. Qual Reliab Eng 28(4):455–466

Dette H, Pepelyshev A (2010) Generalized Latin hypercube design for computer experiments. Technometrics 25:421–429

Deutsch CV (1996) Correcting for negative weights in ordinary Kriging. Comput Geosci 22(7):765–773

Deutsch JL, Deutsch CV (2012) Latin hypercube sampling with multidimensional uniformity. J Stat Plan Inference 142(3):763–772

Evans JR, Olson DL (1998) Introduction to simulation and risk analysis. Prentice-Hall, Upper Saddle River

Fang K-T, Li R, Sudjianto A (2006) Design and modeling for computer experiments. Chapman & Hall/CRC, London

Farah M, Kottas A (2014) Bayesian inference for sensitivity analysis of computer simulators, with an application to radiative transfer models. Technometrics 56(2):159–173

Forrester AIJ (2013) Comment: properties and practicalities of the expected quantile improvement. Technometrics 55(1):13–18

Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1–3):50–79

Forrester A, Sóbester A, Keane A (2008) Engineering design via surrogate modelling: a practical guide. Wiley, Chichester

Frazier PI (2011) Learning with dynamic programming. In: Cochran JJ, Cox LA, Keskinocak P, Kharoufeh JP, Smith JC (eds) Encyclopedia of operations research and management science. Wiley, New York

Gano SE, Renaud JE, Martin JD, Simpson TW (2006) Update strategies for Kriging models for using in variable fidelity optimization. Struct Multidiscip Optim 32(4):287–298

Georgiou SD, Stylianou S (2011) Block-circulant matrices for constructing optimal Latin hypercube designs. J Stat Plan Inference 141:1933–1943

Ghosh BK, Sen PK (eds) (1991) Handbook of sequential analysis. Marcel Dekker, New York

Ginsbourger D, Dupuy D, Badea A, Carraro L, Roustant O (2009) A note on the choice and the estimation of Kriging models for the analysis of deterministic computer experiments. Appl Stoch Models Bus Ind 25: 115–131

Ginsbourger D, Iooss B, Pronzato L (2015) Editorial. J Stat Comput Simul 85(7):1281–1282

Giunta AA, McFarland JM, Swiler LP, Eldred MS (2006) The promise and peril of uncertainty quantification using response surface approximations. Struct Infrastruct Eng 2(3–4):175–189

Goel T, Haftka R, Queipo N, Shyy W (2006) Performance estimate and simultaneous application of multiple surrogates. In: 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, multidisciplinary analysis optimization conferences. American Institute of Aeronautics and Astronautics, Reston, VA 20191–4344, pp 1–26

Goh J, Bingham D, Holloway JP, Grosskopf MJ, Kuranz CC, Rutter E (2013) Prediction and computer model calibration using outputs from multi-fidelity simulators. Technometrics 55(4):501–512

Goldberg PW, Williams CKI, Bishop CM (1998) Regression with input-dependent noise: a Gaussian process treatment. In: Jordan MI, Kearns MJ, Solla SA (eds) Advances in neural information processing systems, vol 10. MIT, Cambridge, pp 493–499

Golzari A, Sefat MH, Jamshidi S (2015) Development of an adaptive surrogate model for production optimization. J Petrol Sci Eng (in press)

Gramacy RB and Haaland B (2015) Speeding up neighborhood search in local Gaussian process prediction. Technometrics (in press)

Gramacy RB, Lee HKH (2008) Bayesian treed Gaussian process models with an application to computer modeling. J Am Stat Assoc 103(483):1119–1130

Gramacy RB, Lee HKH (2012) Cases for the nugget in modeling computer experiments. Stat Comput 22:713–722

Grosso A, Jamali ARMJU, Locatelli M (2009) Finding maximin Latin hypercube designs by iterated local search heuristics. Eur J Oper Res 197(2):541–54

Hankin RKS (2005) Introducing BACCO, an R bundle for Bayesian analysis of computer code output. J Stat Softw 14(16):1–21

Harari O, Steinberg DM (2014a) Optimal designs for Gaussian process models via spectral decomposition. J Stat Plan Inference (in press)

Harari O, Steinberg DM (2014b) Convex combination of Gaussian processes for Bayesian analysis of deterministic computer experiments. Technometrics 56(4):443–454

Helton JC, Davis FJ, Johnson JD (2005) A comparison of uncertainty and sensitivity results obtained with random and Latin hypercube sampling. Reliab Eng Syst Saf 89:305–330

Helton JC, Johnson JD, Oberkampf WD, Sallaberry CJ (2006a) Sensitivity analysis in conjunction with evidence theory representations of epistemic uncertainty. Reliab Eng Syst Saf 91:1414–1434

Helton JC, Johnson JD, Oberkampf WD, Sallaberry CJ (2010) Representation of analysis results involving aleatory and epistemic uncertainty. Int J Gen Syst 39(6):605–646

Helton JC, Johnson JD, Sallaberry CJ, Storlie CB (2006b) Survey of sampling-based methods for uncertainty and sensitivity analysis. Reliab Eng Syst Saf 91:1175–1209

Helton JC, Pilch M (2011) Guest editorial: quantification of margins and uncertainty. Reliab Eng Syst Saf 96:959–964

Helton JC, Hansen CW, Sallaberry CJ (2014) Conceptual structure and computational organization of the 2008 performance assessment for the proposed high-level radioactive waste repository at Yucca Mountain, Nevada. Reliab Eng Syst Saf 122:223–248

Henkel T, Wilson H, Krug W (2012) Global sensitivity analysis of nonlinear mathematical models – an implementation of two complementing variance-based algorithms. In: Laroque C, Himmelspach J, Pasupathy R, Rose O, Uhrmacher AM (eds) Proceedings of the 2012 winter simulation conference, Washington, DC, pp 1737–1748

Hernandez AF, Grover MA (2010) Stochastic dynamic predictions using Gaussian process models for nanoparticle synthesis. Comput Chem Eng 34(12):1953–1961

Hernandez AS, Lucas TW, Sanchez PJ (2012) Selecting random Latin hypercube dimensions and designs through estimation of maximum absolute pairwise correlation. In: Laroque C, Himmelspach J, Pasupathy R, Rose O, Uhrmacher AM (eds) Proceedings of the 2012 winter simulation conference, Berlin, pp 280–291

Hubert M, Engelen S (2007) Fast cross-validation of high-breakdown resampling methods for PCA. Comput Stat Data Anal 51(10):5013–5024

Iooss B, Boussouf L, Feuillard V, Marrel A (2010) Numerical studies of the metamodel fitting and validation processes. Int J Adv Syst Meas 3:11–21

Jala M, Lévy-Leduc C, Moulines É, Conil E, Wiart J (2014) Sequential design of computer experiments for the assessment of fetus exposure to electromagnetic fields. Technometrics (in press)

Janssen H (2013) Monte-Carlo based uncertainty analysis: sampling efficiency and sampling convergence. Reliab Eng Syst Saf 109:123–132

Jeon JS, Lee SR, Pasquinelli L, Fabricius IL (2015) Sensitivity analysis of recovery efficiency in high-temperature aquifer thermal energy storage with single well. Energy (in press)

Jian N, Henderson S, Hunter SR (2014) Sequential detection of convexity from noisy function evaluations. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3892–3903

Jin, R, Chen W, Sudjianto A (2002) On sequential sampling for global metamodeling in engineering design. In: Proceedings of DET'02, ASME 2002 design engineering technical conferences and computers and information in engineering conference, DETC2002/DAC-34092, Montreal, 29 Sept–2 Oct 2002

Jones B, Silvestrini RT, Montgomery DC, Steinberg DM (2015) Bridge designs for modeling systems with low noise. Technometrics 57(2): 155–163

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13:455–492

Joseph VR, Hung Y, Sudjianto A (2008) Blind Kriging : a new method for developing metamodels. J Mech Des 130(3):31–102

Jourdan A, Franco J (2010) Optimal Latin hypercube designs for the Kullback-Leibler criterion. AStA Adv Stat Anal 94:341–351

Kamiński B (2015) A method for updating of stochastic Kriging metamodels. Eur J Oper Res (accepted)

Kersting K, Plagemann C, Pfaff P, Burgard W (2007) Most-likely heteroscedastic Gaussian process regression. In: Ghahramani Z (ed) Proceedings of the 24th annual international conference on machine learning (ICML-07), Corvalis, pp 393–400

Kleijnen JPC (1983). Risk analysis and sensitivity analysis: antithesis or synthesis?. Simuletter, **14**(1–4):64–72

Kleijnen JPC (1990) Statistics and deterministic simulation models: why not? In: Balci O, Sadowski RP, Nance RE (eds) Proceedings of the 1990 winter simulation conference, Washington, DC, pp 344–346

Kleijnen JPC (1994) Sensitivity analysis versus uncertainty analysis: when to use what? In: Grasman J, van Straten G (eds) Predictability and nonlinear modelling in natural sciences and economics. Kluwer, Dordrecht, pp 322–333

Kleijnen JPC (1997) Sensitivity analysis and related analyses: a review of some statistical techniques. J Stat Comput Simul 57(1–4):111–142

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Kleijnen JPC (2009) Kriging metamodeling in simulation: a review. Eur J Oper Res 192(3):707–716

Kleijnen JPC (2014) Simulation-optimization via Kriging and bootstrapping: a survey. J Simul 8(4):241–250

Kleijnen JPC, Mehdad E (2013) Conditional simulation for efficient global optimization. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 winter simulation conference, Washington, DC, pp 969–979

Kleijnen JPC, Mehdad E (2014) Multivariate versus univariate Kriging metamodels for multi-response simulation models. Eur J Oper Res 236:573–582

Kleijnen JPC, Mehdad E (2015) Estimating the correct predictor variance in stochastic Kriging. CentER Discussion Paper, 2015, Tilburg

Kleijnen JPC, Mehdad E, Van Beers WCM (2012) Convex and monotonic bootstrapped Kriging. In: Laroque C, Himmelspach J, Pasupathy R, Rose O, Uhrmacher AM (eds) Proceedings of the 2012 winter simulation conference, Washington, DC, pp 543–554

Kleijnen JPC, Pierreval H, Zhang J (2011) Methodology for determining the acceptability of system designs in uncertain environments. Eur J Oper Res 209:176–183

Kleijnen JPC, Ridder AAN, Rubinstein RY (2013) Variance reduction techniques in Monte Carlo methods. In: Gass SI, Fu MC (eds) Encyclopedia of operations research and management science, 3rd edn. Springer, New York, pp 1598–1610

Kleijnen JPC, Van Beers WCM (2004) Application-driven sequential designs for simulation experiments: Kriging metamodeling. J Oper Res Soc 55(9):876–883

Kleijnen JPC, Van Beers WCM (2013) Monotonicity-preserving bootstrapped Kriging metamodels for expensive simulations. J Oper Res Soc 64:708–717

Koch P, Wagner T, Emmerich MTM, Bäck T, Konen W (2015) Efficient multi-criteria optimization on noisy machine learning problems. Appl Soft Comput (in press)

Koziel S, Bekasiewicz A, Couckuyt I, Dhaene T (2014) Efficient multi-objective simulation-driven antenna design using co-Kriging. IEEE Trans Antennas Propag 62(11):5901–5915

Krige DG (1951) A statistical approach to some basic mine valuation problems on the Witwatersrand. J Chem, Metall Min Soc S Afr 52(6):119–139

Lamboni M, Iooss B, Popelin A-L, Gamboa F (2013) Derivative-based global sensitivity measures: general links with Sobol indices and numerical tests. Math Comput Simul 87:45–54

Lancaster P, Salkauskas K (1986) Curve and surface fitting: an introduction. Academic, London

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Le Gratiet L, Cannamela C (2015) Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes. Technometrics 57(3):418–427

Lemaître P, Sergienko E, Arnaud A, Bousquet N, Gamboa F, Iooss B (2014) Density modification based reliability sensitivity analysis. J Stat Comput Simul (in press)

Lemieux C (2009) Monte Carlo and quasi-Monte Carlo sampling. Springer, New York

Li K, Jiang B, Ai M (2015) Sliced space-filling designs with different levels of two-dimensional uniformity. J Stat Plan Inference 157–158:90–99

Li R, Sudjianto A (2005) Analysis of computer experiments using penalized likelihood in Gaussian Kriging models. Technometrics 47(2):111–120

Li Y, Zhou Q (2015) Pairwise meta-modeling of multivariate output computer models using nonseparable covariance function. Technometrics (in press)

Lin Y, Mistree F, Allen JK, Tsui K-L, Chen VCP (2004) Sequential metamodeling in engineering design. In: 10th AIAA/ISSMO symposium on multidisciplinary analysis and optimization, Albany, 30 Aug–1 Sept, 2004. Paper number AIAA-2004-4304

Lin Y, Mistree F, Tsui K-L, Allen JK (2002) Metamodel validation with deterministic computer experiments. In: 9th AIAA/ISSMO symposium on multidisciplinary analysis and optimization, Atlanta, 4–6 Sept 2002. Paper number AIAA-2002-5425

Lloyd-Smith B, Kist AA, Harris RJ, Shrestha N (2004) Shortest paths in stochastic networks. In: Proceedings 12th IEEE international conference on networks 2004, Wakefield, MA, vol 2, pp 492–496

Loeppky JL, Sacks J, Welch W (2009) Choosing the sample size of a computer experiment: a practical guide. Technometrics 51(4):366–376

Lophaven SN, Nielsen HB, Sondergaard J (2002) DACE: a Matlab Kriging toolbox, version 2.0. IMM Technical University of Denmark, Kongens Lyngby

MacCalman AD, Vieira H, Lucas T (2013) Second order nearly orthogonal Latin hypercubes for exploring stochastic simulations. Naval Postgraduate School, Monterey

McKay MD, Beckman RJ, Conover WJ (1979) A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21(2):239–245 (reprinted in Technometrics, 42(1,2000):55–61)

Marrel A, Iooss B, Da Veiga S, Ribatet M (2012) Global sensitivity analysis of stochastic computer models with joint metamodels. Stat Comput 22:833–847

Marrel A, Iooss B, Van Dorpe F, Volkova E (2008) An efficient methodology for modeling complex computer codes with Gaussian processes. Comput Stat Data Anal 52:4731–4744

Martin JD, Simpson TW (2005) Use of Kriging models to approximate deterministic computer models. AIAA J 43(4):853–863

Martin JD, Simpson TW (2006) A methodology to manage system-level uncertainty during conceptual design. ASME J Mech Des 128(4): 959–968

Matheron G (1963) Principles of geostatistics. Econ Geol 58(8):1246–1266

Mehdad E, Kleijnen JPC (2015a) Classic Kriging versus Kriging with bootstrapping or conditional simulation: classic Kriging's robust confidence intervals and optimization. J Oper Res Soc (in press)

Mehdad E, Kleijnen JPC (2015b) Stochastic intrinsic Kriging for simulation metamodelling. CentER Discussion Paper, Tilburg

Meng Q, Ng SH (2015, in press) An additive global and local Gaussian process model for large datasets. In: Yilmaz L, Chan WKV, Moon I, Roeder TMK, Macal C, Rossetti MD (eds) Proceedings of the 2015 winter simulation conference. [Will be made available on the WSC website in January 2016, after the conference in Dec. 2015]

Miller GA (1956) The magical number seven plus or minus two: some limits on our capacity for processing information. The Psychol Rev 63:81–97

Mitchell TJ, Morris MD (1992) The spatial correlation function approach to response surface estimation. In: Swain JJ, Goldsman D, Crain RC, Wilson JR (eds) Proceedings of the 1992 winter simulation conference, Arlington

Moutoussamy V, Nanty S, Pauwels B (2014) Emulators for stochastic simulation codes. In: ESAIM: Proceedings, Azores, pp 1–10

Muehlenstaedt T, Roustant O, Carraro L, Kuhnt S (2012) Data-driven Kriging models based on FANOVA-decomposition. Stat Comput 22:723–738

Ng SH, Yin J (2012), Bayesian Kriging analysis and design for stochastic simulations. ACM Trans Model Comput Simul **22**(3):1–26

Norton J (2015) An introduction to sensitivity assessment of simulation models. Environ Model Softw 69:166–174

Oakley J, O'Hagan A (2004) Probabilistic sensitivity analysis of complex models: a Bayesian approach. J R Stat Soc, Ser B, 66(3):751–769

Opsomer JD, Ruppert D, Wand MP, Holst U, Hossjer O (1999) Kriging with nonparametric variance function estimation. Biometrics 55(3): 704–710

Owen AB, Dick J, Chen S (2013) Higher order Sobol' indices. http://arxiv.org/abs/1306.4068

Plumlee M, Tuo R (2014) Building accurate emulators for stochastic simulations via quantile Kriging, Technometrics 56(4):466–473

Qian PZG, Hwang Y, Ai M, Su H (2014) Asymmetric nested lattice samples. Technometrics 56(1):46–54

Qu H, Fu MC (2014) Gradient extrapolated stochastic kriging. ACM Trans Model Comput Simul 24(4):23:1–23:25

Quadrianto N, Kersting K, Reid MD, Caetano TS, Buntine WL (2009) Kernel conditional quantile estimation via reduction revisited. In: IEEE 13th international conference on data mining (ICDM), Miami, pp 938–943

Quaglietta E (2013) Supporting the design of railway systems by means of a Sobol variance-based sensitivity analysis. Transp Res Part C 34:38–54

Ranjan P, Spencer N (2014) Space-filling Latin hypercube designs based on randomization restrictions in factorial experiments. Stat Probab Lett (in press)

Rasmussen CE, Nickisch H (2010) Gaussian processes for machine learning (GPML) toolbox. J Mach Learn Res 11:3011–3015

Rasmussen CE, Williams C (2006) Gaussian processes for machine learning. MIT, Cambridge

Razavi S, Tolson BA, Burn DH (2012) Review of surrogate modeling in water resources. Water Resour Res 48, W07401:1–322

Razavi S, Gupta HV (2015) What do we mean by sensitivity analysis? The need for comprehensive characterization of “global” sensitivity in earth and environmental systems models. Water Resour Res 51 (in press)

Risk J, Ludkovski M (2015) Statistical emulators for pricing and hedging longevity risk products. Preprint arXiv:1508.00310

Roustant O, Ginsbourger D, Deville Y (2012) DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization. J Stat Softw 51(1):1–55

Sacks J, Welch WJ, Mitchell TJ, Wynn HP (1989) Design and analysis of computer experiments (includes comments and rejoinder). Stat Sci 4(4):409–435

Salemi P, Staum J, Nelson BL (2013) Generalized integrated Brownian fields for simulation metamodeling. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) Proceedings of the 2013 winter simulation conference, Washington, DC, pp 543–554

Saltelli A, Annoni P, Azzini I, Campolongo F, Ratto M, Tarantola S (2010) Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index. Comput Phys Commun 181:259–270

Saltelli A, Ratto M, Andres T, Campolongo F, Cariboni J, Gatelli D, Saisana M, Tarantola S (2008) Global sensitivity analysis: the primer. Wiley, Chichester

Santner TJ, Williams BJ, Notz WI (2003) The design and analysis of computer experiments. Springer, New York

Shahraki AF, Noorossana R (2014) Reliability-based robust design optimization: a general methodology using genetic algorithm. Comput Ind Eng 74:199–207

Simpson TW, Booker AJ, Ghosh D, Giunta AA, Koch PN, Yang R-J (2004) Approximation methods in multidisciplinary analysis and optimization: a panel discussion. Struct Multidiscip Optim 27(5):302–313

Simpson TW, Mauery TM, Korte JJ, Mistree F (2001) Kriging metamodels for global approximation in simulation-based multidisciplinary design. AIAA J 39(12):853–863

Sobol IM (1990) Sensitivity estimates for non-linear mathematical models. Matematicheskoe Modelirovanie 2:112–118

Song E, Nelson BL, Pegden D (2014) Advanced tutorial: input uncertainty quantification. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 162–176

Spöck G, Pilz J (2015) Incorporating covariance estimation uncertainty in spatial sampling design for prediction with trans-Gaussian random fields. Front Environ Sci 3(39):1–22

Stein ML (1999) Statistical interpolation of spatial data: some theory for Kriging. Springer, New York

Storlie CB, Swiler LP, Helton JC, Sallaberry CJ (2009) Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models. Reliab Eng Syst Saf 94(11): 1735–1763

Stripling HF, Adams ML, McClarren RG, Mallick BK (2011) The method of manufactured universes for validating uncertainty quantification methods. Reliab Eng Syst Saf 96(9):1242–1256

Sun L, Hong LJ, Hu Z (2014) Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. Oper Res **62**(6):1416–1438

Sundararajan S, Keerthi SS (2001) Predictive approach for choosing hyperparameters in Gaussian processes. Neural Comput 13(5):1103–1118

Tajbakhsh DS, Del Castillo E, Rosenberger JL (2014) A fully Bayesian approach to sequential optimization of computer metamodels for process improvement. Qual Reliab Eng Int **30**(4):449–462

Tan MHY (2014a) Robust parameter design with computer experiments using orthonormal polynomials. Technometrics (in press)

Tan MHY (2014b) Stochastic polynomial interpolation for uncertainty quantification with computer experiments. Technometrics (in press)

Tan MHY (2015) Monotonic quantile regression with Bernstein polynomials for stochastic simulation. Technometrics (in press)

Thiart C, Ngwenya MZ, Haines LM (2014) Investigating 'optimal' kriging variance estimation using an analytic and a bootstrap approach. J S Afr Inst Min Metall 114:613–618

Toal DJJ, Bressloff NW, Keane AJ (2008) Kriging hyperparameter tuning strategies. AIAA J 46(5):1240–1252

Toropov VV, Schramm U, Sahai A, Jones R, Zeguer T (2005) Design optimization and stochastic analysis based on the moving least squares method. In: 6th world congress of structural and multidisciplinary optimization, Rio de Janeiro, paper no. 9412

Tuo RC, Wu FJ, Yuc D (2014) Surrogate modeling of computer experiments with different mesh densities. Technometrics 56(3):372–380

Ulaganathan S, Couckuyt I, Dhaene T, Laermans E (2014) On the use of gradients in Kriging surrogate models. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 2692–2701

Van Beers WCM, Kleijnen JPC (2003) Kriging for interpolation in random simulation. J Oper Res Soc 54:255–262

Van Beers WCM, Kleijnen JPC (2008) Customized sequential designs for random simulation experiments: Kriging metamodeling and bootstrapping. Eur J Oper Res 186(3):1099–1113

Viana FAC, Haftka RT (2009) Cross validation can estimate how well prediction variance correlates with error. AIAA J 47(9):2266–2270

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling in multidisciplinary design optimization: how far have we really come? AIAA J 52(4):670–690

Vieira H, Sanchez S, Kienitz KH, Belderrain MCN (2011) Generating and improving orthogonal designs by using mixed integer programming. Eur J Oper Res 215:629–638

Vose D (2000) Risk analysis; a quantitative guide, 2nd edn. Wiley, Chichester

Wackernagel H (2003) Multivariate geostatistics: an introduction with applications, 3rd edn. Springer, Berlin

Wang C, Duan Q, Gong W, Ye A, Di Z, Miao C (2014) An evaluation of adaptive surrogate modeling based optimization with two benchmark problems. Environ Model Softw 60:167–179

Wei P, Lu Z, Song J (2015) Variable importance analysis: a comprehensive review. Reliab Eng Syst Saf 142:399–432

Wit E, Van den Heuvel E, Romeijn J-W (2012) All models are wrong . . . : an introduction to model uncertainty, Statistica Neerlandica 66(3):217–236

Xie W, Nelson BL, Barton RR (2014) A Bayesian framework for quantifying uncertainty in stochastic simulation. Oper Res (in press)

Xu J, Zhang S, Huang E, Chen C-H, Lee H, Celik N (2014) Efficient multifidelity simulation optimization. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3940–3951

Yang X, Chen H, Liu MQ (2014) Resolvable orthogonal array-based uniform sliced Latin hypercube designs. Stat Probab Lett 93:108–115

Yin J, Ng SH, Ng KM (2009) A study on the effects of parameter estimation on Kriging model's prediction error in stochastic simulation. In: Rossini MD, Hill RR, Johansson B, Dunkin A, Ingalls RG (eds) Proceedings of the 2009 winter simulation conference, Austin, pp 674–685

Yin J, Ng SH, Ng KM (2010) A Bayesian metamodeling approach for stochastic simulations. In: Johansson B, Jain S, Montoya-Torres J, Hugan J, Yücesan E (eds) Proceedings of the 2010 winter simulation conference, Baltimore, pp 1055–1066

Yuan J, Ng SH (2015) An integrated approach to stochastic computer model calibration, validation and prediction. Trans Model Comput Simul **25**(3), Article No. 18

Zhang Z (2007) New modeling procedures for functional data in computer experiments. Doctoral dissertation, Department of Statistics, Pennsylvania State University, University Park

Zhang Z, Li R, Sudjianto A (2007) Modeling computer experiments with multiple responses. SAE Int 2007-01-1655

Zhou Q, Qian PZG, Zhou S (2011) A simple approach to emulation for computer models with qualitative and quantitative factors. Technometrics 53:266–273

Zuniga MM, Kucherenko S, Shah N (2013) Metamodelling with independent and dependent inputs. Comput Phys Commun 184(6):1570–1580

6
Simulation Optimization

This chapter is organized as follows. Section 6.1 introduces the optimization of real systems that are modeled through either deterministic or random simulation; this optimization we call *simulation optimization* or briefly *optimization*. There are many methods for this optimization, but we focus on methods that use specific metamodels of the underlying simulation models; these metamodels were detailed in the preceding chapters, and use either linear regression or Kriging. Section 6.2 discusses the use of linear regression metamodels for optimization. Section 6.2.1 summarizes basic *response surface methodology* (RSM), which uses linear regression; RSM was developed for experiments with real systems. Section 6.2.2 adapts this RSM to the needs of random simulation. Section 6.2.3 presents the *adapted steepest descent* (ASD) search direction. Section 6.2.4 summarizes *generalized RSM* (GRSM) for simulation with multiple responses. Section 6.2.5 summarizes a procedure for testing whether an estimated optimum is truly optimal—using the *Karush-Kuhn-Tucker* (KKT) conditions. Section 6.3 discusses the use of Kriging metamodels for optimization. Section 6.3.1 presents *efficient global optimization* (EGO), which uses Kriging. Section 6.3.2 presents *Kriging and integer mathematical programming* (KrIMP) for the solution of problems with constrained outputs. Section 6.4 discusses *robust optimization* (RO), which accounts for uncertainties in some inputs. Section 6.4.1 discusses RO using RSM, Sect. 6.4.2 discusses RO using Kriging, and Sect. 6.4.3 summarizes the *Ben-Tal* et al. approach to RO. Section 6.5

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8_6

summarizes the major conclusions of this chapter, and suggests topics for future research. The chapter ends with Solutions of exercises, and a long list of references.

6.1 Introduction

In *practice*, the optimization of engineered systems (man-made artifacts) is important, as is emphasized by Oden (2006)'s "National Science Foundation (NSF) Blue Ribbon Panel" report on simulation-based engineering. That report also emphasizes the crucial role of *uncertainty* in the input data for simulation models; we find that this uncertainty implies that robust optimization is important.

In *academic research*, the importance of optimization is demonstrated by the many sessions on this topic at the yearly Winter Simulation Conferences on discrete-event simulation; see
http://www.wintersim.org/.

The simplest type of optimization problems has no constraints for the input or the output, has no uncertain inputs, and concerns the expected value of a single (univariate) output; see the many test functions in Regis (2014). Obviously, in deterministic simulation the expected value is identical to the observed output of the simulation model for a given input combination. In random simulation, the expected value may also represent the probability of a binary variable having the value one, so $P(w = 1) = p$ and $P(w = 0) = 1 - p$ so $E(w) = p$. The expected value, however, excludes quantiles (e.g., the median and the 95 % quantile or percentile) and the mode of the output distribution. Furthermore, the simplest type of optimization assumes that the inputs are continuous (not discrete or nominal; see the various scales discussed in Sect. 1.3). The assumption of continuous inputs implies that there is an infinite number of systems, so we cannot apply so-called *ranking and selection* (R&S) and *multiple comparison* procedures (there are many publications on these procedures; see the next paragraph). We also refer to
http://simopt.org/index.php,
which is a testbed of optimization problems in discrete-event simulation. There are so many optimization methods that we do not try to summarize these methods. Neither do we refer to references that do summarize these methods—except for some very recent comprehensive references on simulation optimization that we list in the following note.

Note: Ajdari and Mahlooji (2014), Alrabghi and Tiwari (2015), Chau et al. (2014), Dellino and Meloni (2015), Figueira and Almada-Lobo (2014), Fu et al. (2014), Gosavi (2015), Homem-de-Mello and Bayraksan (2014), Hong et al. (2015), Jalali and Van Nieuwenhuyse (2015), Lee et al. (2013), Lee and Nelson (2014), Qu et al. (2015), Pasupathy and Ghosh (2014), Tenne and Goh (2010), Van der Herten et al. (2015) with its 800 pages, Xu et al. (2015) and Zhou et al. (2014).

In this chapter we focus on optimization that uses metamodels (approximations, emulators, surrogates); metamodels were introduced in Sect. 1.2. Moreover, we focus on metamodels that use either linear regression or Kriging; these two types of metamodels are detailed in the preceding four chapters. Jalali and Van Nieuwenhuyse (2015) claims that metamodel-based optimization is "relatively common" and that RSM is the most popular metamodel-based method, while Kriging is popular in theoretical publications. Like we did in the preceding chapters, we consider both deterministic and random simulation models in the present chapter. We define random simulation (including discrete event simulation) as simulation that uses pseudorandom numbers (PRN).

Note: Outside the discrete-event simulation area, some authors speak of RSM but they mean what we call the what-if regression-metamodeling approach, not the sequential (iterative) optimization approach. Other authors speak of RSM, but use global Kriging instead of local low-order polynomials. Many authors use the term "response surface" instead of "metamodel"; an example is Rikards and Auzins (2002).

Like in the preceding chapters, we focus on *expensive* simulation, in which it takes relatively much computer time for a single simulation run (such a run is a single realization of the time path of the simulated system). For example, 36 to 160 h of computer time were needed to simulate a crash model at Ford Motor Company; see the panel discussion reported in Simpson et al. (2004). This panel also reports the example of a (so-called "cooling") problem with 12 inputs, 10 constraints, and 1 objective function. For such expensive simulations, many simulation optimization methods are unpractical. An example is the popular software called *OptQuest* (which combines so-called tabu search, neural networks, and scatter search; it is an add-on to discrete-event simulation software such as Arena, CrystallBall, MicroSaint, ProModel, and Simul8); see

`http://www.opttek.com/OptQuest`.

OptQuest requires relatively many simulation replications and input combinations; see the inventory example in Kleijnen and Wan (2007). Fortunately, the mathematical and statistical computations required by optimization based on RSM or Kriging are negligible—compared with the computer time required by the "expensive" simulation runs.

In many OR applications, a single simulation run is computationally inexpensive, but there are extremely many input combinations; e.g., an M/M/1 model may have one input—namely, the traffic rate—that is continuous, so we can distinguish infinitely many input values but we can simulate only a fraction of these values in finite time. Actually, most simulation models have multiple inputs (say) k, so there is the "curse of dimensionality"; e.g., if we have $k = 7$ inputs (also see Miller 1956) and we experiment with only 10 values per input, then we still have 10^7 (10 million) combinations. Moreover, a single run may be expensive if we wish to estimate the steady-state performance of a queueing system with a high traffic rate; e.g. we might need to simulate one million customers. Finally, if we wish to estimate the failure probability of a *highly reliable* system, then we need to simulate extremely many customers—unless we apply importance sampling.

Note: This chapter is based on Kleijnen (2014).

6.2 Linear Regression for Optimization

Linear regression models are used in RSM. We shall discuss RSM in several subsections; namely Sect. 6.2.1 on basic RSM, Sect. 6.2.2 on RSM in random simulation, Sect. 6.2.3 on adapted steepest descent (ASD), Sect. 6.2.4 on generalized RSM (GRSM) for multiple responses, and Sect. 6.2.5 on testing the KKT conditions of an optimum estimated through GRSM. We shall return to RSM in the section on robust optimization; see especially Sect. 6.4.1.

6.2.1 Response Surface Methodology (RSM): Basics

Originally, RSM was developed for the optimization of *real* (physical) systems.

Note: The classic article is Box and Wilson (1951). The origin of RSM is nicely discussed in Box (1999), an overview of RSM publications during the period 1966–1988 is Myers et al. (1989) and a recent overview is Khuri and Mukhopadhyay (2010), a popular handbook is Myers et al. (2009), and recent RSM software can be found on the Web; e.g., the Design-Expert software and Minitab's "Response Optimizer" are found on

`www.statease.com`

`http://www.minitab.com/`.

RSM in *simulation* was first detailed in the monograph Kleijnen (1975). Unfortunately, RSM (unlike search heuristics such as OptQuest) has not yet been implemented as an add-on to *commercial off the shelf* (COTS) simulation software.

Note: One of the first case-studies on RSM in random simulation is Van den Bogaard and Kleijnen (1977), reporting on a computer center with two servers and three priority classes—with small, medium, and large jobs—estimating the 90 % quantiles of the waiting times per class for different class limits, and applying RSM to find the optimal class limits. RSM in random simulation is also discussed in Alaeddini et al. (2013), Barton and Meckesheimer (2006), Huerta and Elizondo (2014), Law (2015), and Rosen et al. (2008). Google gave more than two million results for the term "Response Surface Methodology", on 4 February 2014.

RSM treats the real system or its simulation model—either a deterministic or a random model—as a *black box*; i.e., RSM observes the input/output

(I/O) of the simulation model—but not the internal variables and specific functions implied by the simulation's computer modules. RSM is a *sequential* heuristic; i.e., it uses a sequence of local experiments that is meant to lead to the optimum input combination. Note that an input combination is also called a point or a scenario. RSM uses design of experiments (DOE) and the concomitant linear regression analysis. Though RSM is only a heuristic, it has gained a good track record, as we shall see in the next subsections.

Regarding this track record, we add that practitioners may not be interested in convergence proofs, because realistic experiments may be so expensive that large sample sizes are impossible; e.g., the computer budget may be so limited that only a small sample is possible (see the literature on *optimal computer budget allocation* or OCBA). Practitioners may be more interested in finding better solutions than the current one. Actually, we may claim that "the best is the enemy of the better" (this claim is inspired by Voltaire's expression "le mieux est l'ennemi du bien" or "perfect is the enemy of good"). Herbert Simon (1956) claims that humans strive for a "satisficing" solution instead of the optimal solution. Samuelson (2010) also emphasizes that it may be impractical to search for the very best. Furthermore, the website

http://simopt.org/index.php

states "We are particularly interested in increasing attention on the finite time performance of algorithms, rather than the asymptotic results that one often finds in related literature". Finally, we quote an anonymous source: "Unfortunately, these theoretical convergence results mean little in practice where it is more important to find high quality solutions within a reasonable length of time than to guarantee convergence to the optimum in an infinite number of steps."

We assume that RSM is applied, only after the important inputs and their experimental area have been identified; i.e., before RSM starts, we may need to use *screening* to identify the really important inputs among the many conceivably important inputs. Case studies illustrating screening followed by RSM are Morales-Enciso and Branke (2015) and Shi et al. (2014). In Chap. 4 we detailed various screening methods, focusing on sequential bifurcation. Chang et al. (2014) combines RSM with screening in a single method. We point out that RSM without a preceding screening phase may imply the simulation of extremely many combinations of simulation inputs, as we shall see in this section.

RSM starts with a sequence of local *metamodels* that are first-order polynomials in the inputs. Once the optimum seems close, RSM augments the latest first-order polynomial to a second-order polynomial. Basic RSM tries to minimize the expected value of a single output, with continuous inputs and without any constraints:

$$\min E(w_0|\mathbf{z}) \tag{6.1}$$

where $E(w_0|\mathbf{z})$ is the goal or objective output (in Sect. 6.2.4 we shall discuss multiple outputs w_h with $h = 0, 1, \ldots, r$), which is to be minimized through the choice of the input combinations $\mathbf{z} = (z_1, \ldots, z_k)'$ where z_j $(j = 1, \ldots k)$ denotes the j^{th} "original" input; i.e., the inputs are not standardized such that they lie between -1 and 1 (sometimes, the inputs are standardized such they lie between 0 and 1). Obviously, if we wish to maximize (instead of minimize) the output $E(w_0)$, then we simply add a minus sign in front of the output in Eq. (6.1) before we minimize it. If the output is deterministic, then $E(w_0) = w_0$.

Note: In random simulation, we may write $E(w_0|\mathbf{z})$ in Eq. (6.1) as

$$E(w_0|\mathbf{z}) = \int_0^1 \cdots \int_0^1 f_{\text{sim}}(\mathbf{z}, \mathbf{r}) d\mathbf{r}$$

where $f_{\text{sim}}(\mathbf{z}, \mathbf{r})$ denotes the computer simulation program, which is a mathematical function that maps the inputs $\mathbf{z}$ and the PRN vector $\mathbf{r}$ (with elements r that have a uniform marginal distribution on $(0, 1)$) to the random simulation response (output) w_0.

RSM has the following *characteristic*s, which we shall detail below.

- RSM is an *optimization heuristic* that tries to estimate the input combination that minimizes a given goal function; see again Eq. (6.1). Because RSM is only a heuristic, it does not guarantee success.

- RSM is a *stepwise* (multi-stage) method; see the steps below.

- In each step, RSM fits a local first-order *polynomial* regression (meta) model—except for the last step, in which RSM fits a second-order polynomial.

- To fit (estimate, calibrate) these first-order polynomials, RSM uses I/O data obtained through so-called *resolution-III (R-III) designs*; for the second-order polynomial, RSM uses a *central composite design (CCD)*; we have already detailed these R-III designs and CCDs in Chap. 2.

- Each step—except the last one—selects the direction for changing the inputs through the *gradient* implied by the first-order polynomial fitted in that step. This gradient is used in the mathematical (not statistical) technique of *steepest descent*—or steepest ascent, in case the output is to be maximized.

- In the final step, RSM takes the *derivatives* of the locally fitted *second-order polynomial* to estimate the optimum input combination. RSM may also apply the mathematical technique of *canonical analysis* to this polynomial, to examine the shape of the optimal subregion; i.e., does that region have a unique minimum, a saddle point, or a ridge with stationary points?

More specifically, the RSM algorithm (for either real or simulated systems) consists of the following *steps* (also see Fig. 6.1 in Sect. 6.2.4, which gives an example with a random goal output w_0 and two constrained random outputs w_1 and w_2; these constrained outputs vanish in basic RSM).

Algorithm 6.1

1. Initialize RSM; i.e., select a *starting point*.
   Comment: This starting point may be the input combination that is currently used in practice if the system already exists; otherwise, we should use intuition and prior knowledge (as in many other heuristics).

2. In the *neighborhood* of this starting point, approximate the I/O behavior through a local first-order polynomial metamodel augmented with additive white noise e:

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j z_j + e \tag{6.2}$$

   with the regression parameters $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_k)'$ where β_0 denotes the intercept and β_j denotes the first-order or "main" effect of input j with $j = 1, \ldots, k$.
   Comment: The first-order polynomial approximation may be explained by Taylor's series expansion. *White noise* (see Definition 2.3 in Chap. 2) means that e is normally, independently, and identically distributed (NIID) with zero mean and a constant variance (say) σ^2 in the local experimental area: $e \sim \text{NIID}(0, \sigma^2)$. However, when the next step moves to a new local area, RSM allows the variance to change.

   Compute the *best linear unbiased estimator* (BLUE) of $\boldsymbol{\beta}$; namely, the *least squares* (LS) estimator

$$\widehat{\boldsymbol{\beta}} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w} \tag{6.3}$$

   where $\mathbf{Z}$ denotes the $N \times (k+1)$ matrix determined by the R-III design and the m_i replications of combination i $(i = 1, \ldots, n)$ with $n \geq k+1$ and $\mathbf{w} = (w_1, \ldots w_N)'$ denotes the vector with the N outputs with $N = \sum_{i=1}^{n} m_i$ where $m_i \geq 1$ denotes the number of replications of combination i.
   Comment: $\mathbf{Z}$ has m_i identical rows where each row has as first element the value 1 which corresponds with the intercept β_0. Obviously, deterministic simulation implies $m_i = 1$ so $N = n$. Unfortunately, there are no general guidelines for determining the appropriate *size* of the local area in a step of RSM; again, intuition and prior knowledge are important. However, Chang et al. (2013) decides on the size of the local area, using a so-called trust region; we shall give some details

in Sect. 6.2.2. Furthermore, so-called "finite differencing" replaces the R-III design by a less efficient one-factor-at-a-time design (see again Sect. 2.3.2) and also faces the problem of selecting an appropriate size for the local area; the optimal size depends on the unknown variance and second-order derivatives; see Brekelmans et al. (2005), Safizadeh (2002), Saltelli et al. (2005), and Zazanis and Suri (1993).

3. Select the next subarea, following the *steepest descent* direction.
   Comment: For example, if the estimated local first-order polynomial is $\widehat{y} = \widehat{\beta}_0 + \widehat{\beta}_1 z_1 + \widehat{\beta}_2 z_2$, then a corresponding contour line is $\widehat{y} = a$ where a denotes some constant (if the goal output w_0 denotes costs, then the contour is also called the iso-costs line). The steepest descent path is *perpendicular* to the local contour lines. This path implies that if $\widehat{\beta}_1 \gg \widehat{\beta}_2$, then z_1 is decreased much more than z_2. Unfortunately, the steepest-descent method is *scale dependent*; i.e., linear transformations of the inputs affect the search direction; see Myers et al. (2009, pp. 193–195). We shall present a scale-independent variant in Sect. 6.2.3, which may interest both practitioners and researchers.

4. Take a step in the direction of steepest descent (estimated in step 3), experimenting with some intuitively selected values for the step size.
   Comment: If the intuitively selected step size yields an output that is significantly higher instead of lower, then we reduce the step size. Otherwise, we take one more step in the current steepest descent direction. A more sophisticated mathematical procedure for selecting the step size will follow in Sect. 6.2.4.

5. If the observed output w increases, then generate n outputs for a new local area centered around the best point found so far.
   Comment: After a number of steps in the steepest descent direction, the output will increase instead of decrease because the first-order polynomial in Eq. (6.2) is only a local approximation of the true I/O function. When such deterioration occurs, we simulate the $n > k$ combinations specified by a R-III design centered around the best point found so far; i.e., we use the same design as in step 2 (see Table 2.3 for an example), but we translate the standardized inputs x_j into different values for the original inputs z_j. One of the corner points of this R-III design may be the best combination found so far; see again Fig. 6.1 below.

6. Estimate the first-order effects in the new local polynomial approximation, using Eq. (6.3).

7. Return to step 3, if the latest locally fitted first-order polynomial is found to be adequate; else proceed to the next step.
   Comment: To test the *adequacy* of the fitted first-order polynomial,

we may apply one or more methods that we have already discussed for estimated linear regression metamodels in general; namely, the lack-of-fit F-statistic for testing whether all estimated first-order effects and hence the gradient are zero (see Sect. 2.2.2), and the coefficient of determination R^2 and cross-validation (see Sect. 3.6).

8. Fit the *second-order polynomial*

$$y = \beta_0 + \sum_{j=1}^{k} \beta_j z_j + \sum_{j=1}^{k} \sum_{j' \geq k}^{k} \beta_{j;j'} z_j z_{j'} + e, \tag{6.4}$$

where β_0 denotes the intercept, β_j $(j = 1, \ldots, k)$ the first-order effect of input j, $\beta_{j;j}$ the purely quadratic effect of input j, and $\beta_{j;j'}$ $(j < j')$ the interaction between inputs j and j'; estimate these $q = 1{+}2k{+}k(k{-}1)/2$ effects through a CCD with $n \geq q$ combinations Comment: It is intuitively clear that the *plane* implied by the most recently estimated local first-order polynomial cannot adequately represent a *hill top* when searching to maximize the output or—equivalently—minimize the output as in Eq. (6.1). So in the neighborhood of the optimum, a first-order polynomial is not adequate. We therefore fit the second-order polynomial defined in Eq. (6.4); RSM uses a CCD to generate the I/O data.

9. Use this fitted second-order polynomial, to estimate the optimal values of the inputs by straightforward *differentiation* or by more sophisticated *canonical analysis*; see Myers et al. (2009, pp. 224–242).

10. If time permits, then try to escape from a possible local minimum and *restart* the search; i.e., return to step 1 with a different initial local area.

    Comment: We shall discuss a *global* search method (namely, efficient global optimization, EGO) in Sect. 6.3.1.

We recommend not to eliminate inputs that have *nonsignificant* effects in a first-order polynomial fitted within the current local experimental area: these inputs may have significant effects in a next experimental area. The selection of the *number of replications* m_i is a moot issue in metamodeling, as we have already discussed for experimental designs in case of linear regression with heterogeneous variances (see Sect. 3.4.5) and for the selection of the number of replications through the sequential probability ratio test (SPRT) for sequential bifurcation (see Sect. 4.5), and for Kriging (see Sect. 5.6.2). For the time being, we recommend estimating the true mean response for a given input combination such that a relative precision of (say) 10 % has a (say) 90 % probability, using the method detailed in Law (2015).

The Taylor series argument suggests that a higher-order polynomial is more accurate than a lower-order polynomial. A statistical counterargument, however, is that *overfitting* gives less accurate estimators of the polynomial coefficients. Consequently, the higher-order polynomial may give a predictor $\widehat{y}$ with lower bias but higher variance such that its mean squared error (MSE) increases. Moreover, a higher-order polynomial requires the simulation of more input combinations.

In Sect. 3.4 we have already mentioned that a *deterministic simulation* model gives a fixed value for a given input combination, so we might assume white noise for the residuals e of the metamodel and apply basic RSM. In *random simulation*, however, we prefer the RSM variant detailed in the next section.

6.2.2 RSM in Random Simulation

We consider the following two characteristics of random simulation that violate the assumption of *white noise* within a given local area:

1. The constant variance assumption does not hold.

2. The independence assumption does not hold if common random numbers (CRN) are applied.

Sub 1: Many simulation models represent queueing systems; e.g., supply chains and telecommunication networks. The simplest queueing model is the so-called M/M/1 model (see Definition 1.4) for which we know that as its traffic rate increases, its mean steady-state waiting time increases and the variance increases even more; consequently, the assumption of a constant variance does not hold.

Sub 2: CRN are often applied in experiments with random simulation models, because CRN are the default option in many simulation software packages (e.g., Arena); moreover, CRN are a simple and intuitive variance reduction technique that gives more accurate estimators of the first-order or second-order polynomial metamodel in Eqs. (6.2) and (6.4). Obviously, the outputs of all input combinations that use CRN are statistically dependent; actually, we expect these outputs to be positively correlated.

Note: CRN are related to *blocking* in real-life experiments. In simulation experiments, we may use blocking when combining CRN and antithetic random numbers through the so-called Schruben-Margolin strategy; this strategy is recently discussed in Chih (2013).

Sub 1 and 2: The preceding two characteristics imply that ordinary LS (OLS) does not give the BLUE. As we have already discussed in Sect. 3.5, generalized LS (GLS) gives the BLUE, but assumes known response variances and covariances. We therefore recommend the following simple estimator, which we have already detailed in Sect. 3.5.

We assume a constant number of replications $m_i = m$ $(i = 1, \ldots, n)$, which is a realistic assumption if CRN are applied. We then compute the OLS estimator per replication replacing $\mathbf{w}$ in Eq. (6.3) by $\mathbf{w}_r$ to get the estimator $\widehat{\boldsymbol{\beta}}_r$ $(r = 1, \ldots, m)$. So, replication r gives an estimator of the steepest descent direction—if a first-order polynomial is used—or the optimum input combination—if a second-order polynomial is used. Together, the m replications give an estimator of the accuracy of this estimated direction or optimum. If we find the estimated accuracy to be too low, then we may simulate additional replications so m increases. Unfortunately, we have not yet any experience with this simple sequential approach for selecting the number of replications.

Actually, if we have $m_i > 1$ $(i = 1, \ldots, n)$ replications, then we can further explore the statistical properties of the OLS estimator of $\boldsymbol{\beta}$ through *distribution-free bootstrapping*, as we have already discussed in Sect. 3.3.5. We can also use the bootstrapped estimator $\widehat{\boldsymbol{\beta}}^*$ to derive confidence intervals (CIs) for the corresponding estimated steepest ascent direction and optimum.

Instead of distribution-free bootstrapping we can apply *parametric bootstrapping*, which assumes a specific type of distribution; e.g., a Gaussian distribution (also see the testing of the KKT conditions in Sect. 6.2.5 below). Parametric bootstrapping may be attractive if m_i is small and no CRN are used; e.g., the n expected values $E(w_i)$ and n variances σ_i^2 can be estimated if the weak condition $m_i > 1$ holds. If CRN are used, then the $n \times n$ covariance matrix $\boldsymbol{\Sigma}_{\mathbf{w}} = (\mathrm{cov}(w_i, w_{i'}))$ with $i, i' = 1, \ldots, n$ needs to be estimated; this estimation requires $m > n$, as proven in Dykstra (1970). So parametric bootstrapping may require fewer replications, but the assumed distribution may not hold for the simulated outputs.

Chang et al. (2013) presents *the stochastic trust-region response-surface method* (STRONG), which is a completely automated variant of RSM combined with so-called trust regions. STRONG is proven to converge to the true optimum (but see again our discussion of convergence, in Sect. 6.2.1). Originally, trust regions were developed in Conn et al. (2000) for deterministic nonlinear optimization. By definition, a *trust region* is a subregion in which the objective function is approximated such that if an adequate approximation is found within the trust region, then the region is expanded; else the region is contracted. STRONG uses these trust regions instead of the "local" regions of basic RSM, detailed in the preceding section. STRONG includes statistical tests to decide whether trust regions should be expanded or shrunken in the various steps, and to decide how much these areas should change. If necessary, the trust region is small and a second-order polynomial is used. Next, Chang et al. (2014) combines STRONG with *screening*, and calls the resulting procedure STRONG-S where S denotes screening. This method is applied to several test functions with multiple local minima. Contrary to the Taylor-series argument, STRONG may have a relatively large trust region that does not require a second-order polynomial metamodel but only a first-order polynomial metamodel. Chang and Lin (2015) applies STRONG—including some adaptation—to a renewable energy system. RSM in random simulation is also discussed in Law (2015, pp. 656–679). Ye and You (2015) uses trust regions, not applied to low-order polynomial metamodels but to deterministic Kriging metamodels of the underlying random simulation model.

Note: I/O data in RSM may contain outliers, which should be detected; for this detection, Huang and Hsieh (2014) presents so-called *influence analysis.*

Exercise 6.1 *Apply RSM to the following problem that is a simple Monte Carlo model of a random simulation:*

$$\mathit{min}\ E[5(z_1-1)^2+(z_2-5)^2+4z_1z_2+e]$$

where $\mathbf{z}=(z_1,z_2)'$ *and* $e\sim \mathit{NIID}(0,1)$. *RSM treats this example as a black box; i.e., you select the input combination* $\mathbf{z}$, *sample* e *from* $\mathit{NIID}(0,1)$, *and use these input data to compute the output (say)* w. *You (not RSM) may use the explicit function to derive the true optimum solution,* $\mathbf{z}_o$.

6.2.3 Adapted Steepest Descent (ASD) for RSM

Kleijnen et al. (2004) derives the so-called *adapted steepest descent* (ASD) direction that accounts for the covariances between the k components of the estimated gradient $\widehat{\boldsymbol{\beta}}_{-0}=(\widehat{\beta}_1,\ldots,\widehat{\beta}_k)'$ where the subscript -0 means that the intercept $\widehat{\beta}_0$ of the estimated first-order polynomial vanishes in the estimated gradient; i.e., $\widehat{\boldsymbol{\beta}}=(\widehat{\beta}_0,\widehat{\boldsymbol{\beta}}_{-0})'$ with $\widehat{\boldsymbol{\beta}}$ defined in Eq. (6.3). Obviously, *white noise* implies

$$\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}}=\sigma_w^2(\mathbf{Z}'\mathbf{Z})^{-1}=\sigma_w^2\begin{pmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{C}\end{pmatrix} \tag{6.5}$$

where σ_w^2 denotes the variance of the output w; $\mathbf{Z}$ is the $N\times(1+k)$ matrix of explanatory regression variables including the column with N one's; $N=\sum_{i=1}^{n}m_i$ where n is the number of different observed input combinations; m_i is the number of IID replications for combination i; a is a scalar; $\mathbf{b}$ is a k-dimensional vector; and $\mathbf{C}$ is a $k\times k$ matrix such that $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}}=\sigma_w^2\mathbf{C}$.

We notice that $\mathbf{Z}$'s first column corresponds with the intercept β_0. Furthermore, $\mathbf{Z}$ is determined by the R-III design, transformed into the original values of the inputs in the local area. To save computer time, we may replicate only the center of the local area; this center is not part of the R-III design.

The variance σ_w^2 in Eq. (6.5) is estimated through the *mean squared residuals* (MSR):

$$\widehat{\sigma}_w^2 = \frac{\sum_{i=1}^{n} \sum_{r=1}^{m_i} (w_{i;r} - \widehat{y}_i)^2}{N - (k+1)} \tag{6.6}$$

where $\widehat{y}_i = \mathbf{z}_i' \widehat{\boldsymbol{\beta}}$; also see Eq. (2.26).

It can be proven that the predictor variance $\mathrm{Var}(\widehat{y}|\mathbf{z})$ increases as $\mathbf{z}$—the point to be predicted—moves away from the local area where the gradient is estimated. The point with the minimum predictor variance is $-\mathbf{C}^{-1}\mathbf{b}$, where $\mathbf{C}$ and $\mathbf{b}$ were defined below Eq. (6.5). ASD means that the new point to be simulated is

$$\mathbf{d} = -\mathbf{C}^{-1}\mathbf{b} - \lambda \mathbf{C}^{-1} \widehat{\boldsymbol{\beta}}_{-0} \tag{6.7}$$

where $-\mathbf{C}^{-1}\mathbf{b}$ is the point where the local search starts (namely, the point with the minimum local variance), λ is the step size, $\widehat{\boldsymbol{\beta}}_{-0}$ is the steepest descent direction, and $\mathbf{C}^{-1}\widehat{\boldsymbol{\beta}}_{-0}$ is the steepest descent direction adapted for $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}}$. It is easy to see that if $\mathbf{C}$ is a diagonal matrix, then the higher the variance of an estimated input effect is, the less the search moves into the direction of that input.

Exercise 6.2 *Prove that the search direction in Eq. (6.7) does not change the steepest descent direction if the design matrix is orthogonal (so* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*).*

It can be proven that ASD, which accounts for $\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\beta}}_{-0}}$, gives a *scale-independent* search direction. Experimental results are presented in Kleijnen et al. (2004, 2006). These results imply that ASD performs "better" than steepest descent; i.e., the angle between the search direction based on the true $\boldsymbol{\beta}_{-0}$ and the search direction estimated in ASD is smaller. In one example this angle reduces from 89.87 for steepest descent to 1.83 for ASD.

Note: Fan and Huang (2011) derives another alternative for steepest ascent, using *conjugate gradients* (which were originally developed for unconstrained optimization in mathematical programming). Joshi et al. (1998) derives one more alternative, using *gradient deflection* methods. Safizadeh (2002) examines how to balance the variance and the bias via the MSE of the estimated gradient for different sizes of the local experimental area, assuming random simulation with CRN.

6.2.4 Multiple Responses: Generalized RSM (GRSM)

In practice, simulation models have *multiple* responses types (multivariate output); e.g., a realistic inventory simulation model may estimate (i) the sum of all inventory costs excluding the (hard-to-quantify) out-of-stock costs and (ii) the service rate (fill rate), and the goal of this simulation is to minimize this sum of inventory costs such that the service rate is not lower than (say) 90 %. Simulation software facilitates the collection of multiple outputs. There are several approaches to solve the resulting issues;

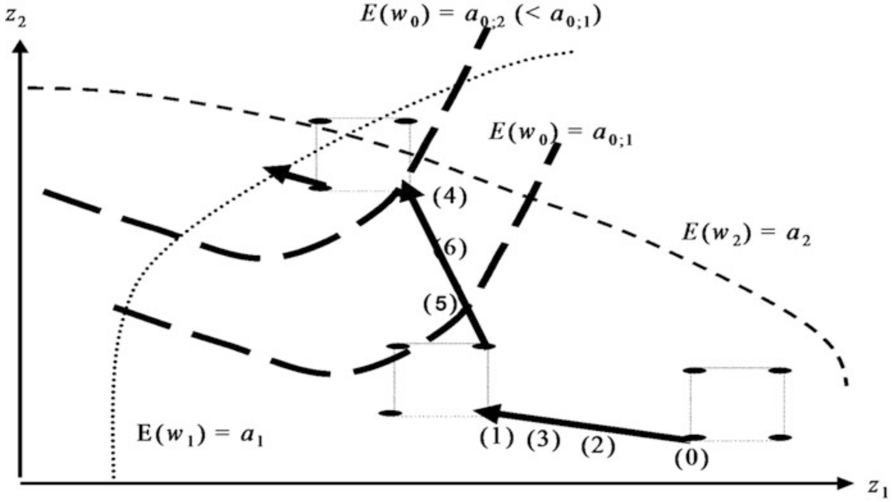


FIGURE 6.1. GRSM example with two inputs, two contour plots for the goal output, two constraints for the other outputs, three local areas, three search directions, and six steps in these directions

see the survey in Rosen et al. (2008). The RSM literature also offers several approaches for such situations, but we shall focus on GRSM.

Note: For RSM with multiple responses we refer to the surveys in Angün (2004), Khuri and Mukhopadhyay (2010), and Ng et al. (2007) and the recent case study in Shi et al. (2014) combining two output types into a single criterion. We shall discuss Kriging for simulation with multiple outputs, in Sect. 6.3.

GRSM is explained in Angün et al. (2009). Informally, we may say that GRSM is RSM for problems with multiple random outputs such that one goal output is minimized while the other outputs satisfy prespecified constraints (so GRSM does not use multi-objective optimization); moreover, the deterministic input variables may also be subjected to constraints. GRSM combines RSM and mathematical programming; i.e., GRSM generalizes the steepest descent direction of RSM through the *affine scaling search direction*, borrowing ideas from *interior point* methods (a variation on Karmarkar's algorithm) as explained in Barnes (1986). As Fig. 6.1 illustrates, the GRSM search avoids creeping along the boundary of the feasible area that is determined by the constraints on the random outputs and the deterministic inputs. So, GRSM moves faster to the optimum than steepest descent. Moreover, this search tries to stay inside the feasible area, so the simulation program does not crash. We shall discuss Fig. 6.1 in detail, at the end of this subsection. We point out that Angün et al. (2009) proves that the GRSM search direction is scale independent. Though we focus on *random* simulations, we might easily adapt GRSM for *deterministic* simulations and *real* systems.

Because GRSM is rather complicated, readers may wish to skip the rest of this subsection and also skip the next subsection (Sect. 6.2.5)—on testing an estimated optimum in GRSM through testing the Karush-Kuhn-Tucker conditions—without lessening their understanding of the rest of this book.

Formally, GRSM extends the basic RSM problem in Eq. (6.1) to the following *constrained nonlinear random optimization problem*:

$$\min E(w_0|\mathbf{z}) \tag{6.8}$$

such that the other $(r-1)$ random outputs satisfy the constraints

$$E(w_{h'}|\mathbf{z}) \geq a_{h'} \quad \text{with } h' = 1, \ldots, r-1, \tag{6.9}$$

and the k deterministic inputs z_j satisfy the *box constraints*

$$l_j \leq z_j \leq u_j \quad \text{with } j = 1, \ldots, k. \tag{6.10}$$

An example is an inventory simulation, in which the sum of the expected inventory carrying costs and ordering costs should be minimized while the expected service percentage should be at least 90 % so $a_1 = 0.9$ in Eq. (6.9); both the reorder quantity $z_1 = Q$ and the reorder level $z_2 = s$ should be non-negative so $z_1 \geq 0$ and $z_2 \geq 0$ in Eq. (6.10). A stricter input constraint may be that z_2 should at least cover the expected demand during the expected order lead time; obviously, these expectations are known inputs of the simulation. More complicated input constraints than Eq. (6.10)—namely, linear budget constraints—feature in a call-center simulation in Kelton et al. (2007).

Note: Optimization of simulated call-centers—but not using GRSM—is also studied in Atlason et al. (2008). Aleatory and epistemic uncertainties—discussed in Sect. 5.9 on risk analysis—in call-center queueing models are studied in Bassamboo et al. (2010). Geometry constraints are discussed in Stinstra and Den Hertog (2008). Input constraints resulting from output constraints are discussed in Ng et al. (2007).

Analogously to RSM's first steps using Eq. (6.2), GRSM locally approximates the multivariate I/O function by r univariate first-order polynomials augmented with white noise:

$$\mathbf{y}_h = \mathbf{Z}\boldsymbol{\beta}_h + e_h \quad \text{with } h = 0, \ldots r-1. \tag{6.11}$$

Analogously to RSM, GRSM assumes that locally the white noise assumption holds for Eq. (6.11), so the BLUEs are the following OLS estimators:

$$\widehat{\boldsymbol{\beta}}_h = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{w}_h \quad \text{with } h = 0, \ldots r-1. \tag{6.12}$$

The vector $\widehat{\boldsymbol{\beta}}_0$ (OLS estimator of first-order polynomial approximation of goal function) and the goal function in Eq. (6.8) result in

$$\min \widehat{\boldsymbol{\beta}}_{0;-0}\mathbf{z} \tag{6.13}$$

where $\widehat{\boldsymbol{\beta}}_{0;-0} = (\widehat{\beta}_{0;1}, \ldots, \widehat{\beta}_{0,k})'$ is the OLS estimator of the local gradient of the goal function. Combining Eq. (6.12) and the original output constraints in Eq. (6.9) gives

$$\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z} \geq c_{h'} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.14}$$

where $\widehat{\boldsymbol{\beta}}_{h';-0} = (\widehat{\beta}_{h';1}, \ldots, \widehat{\beta}_{h',k})'$ is the estimator of the local gradient of constraint function h', and $c_{h'} = a_{h'} - \widehat{\beta}_{h';0}$ is the modified right-hand side of this constraint function. The box constraints in Eq. (6.10) remain unchanged.

Now we collect the k-dimensional vectors $\widehat{\boldsymbol{\beta}}_{h';-0}$ $(h' = 1, \ldots, r-1)$ in Eq. (6.14) in the $(r-1) \times k$ matrix denoted by (say) $\mathbf{B}$. Likewise, we collect the $(r-1)$ elements $c_{h'}$ in the vector $\mathbf{c}$. Furthermore, we define $\mathbf{l}$ as the vector with the k elements l_j, and $\mathbf{u}$ as the vector with the k elements u_j. Finally, we introduce the k-dimensional vectors with the non-negative *slack variables* $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$, to get the following problem formulation that is the equivalent of the problem formulated in Eq. (6.8) through Eq. (6.10):

$$\begin{array}{ll} \text{minimize} & \widehat{\boldsymbol{\beta}}'_{0;-0}\mathbf{z} \\ \text{subject to} & \mathbf{Bz} - \mathbf{s} = \mathbf{c} \\ & \mathbf{z} + \mathbf{r} = \mathbf{u} \\ & \mathbf{z} - \mathbf{v} = \mathbf{l}. \end{array} \tag{6.15}$$

Obviously, the constrained optimization problem in Eq. (6.15) is linear in the inputs $\mathbf{z}$ (the OLS estimates $\widehat{\boldsymbol{\beta}}_{0;-0}$ and $\widehat{\boldsymbol{\beta}}_{h';-0}$ in $\mathbf{B}$ use the property that this problem is also linear in the regression parameters). Angün et al. (2009) uses this problem formulation to derive the following *GRSM search direction*:

$$\mathbf{d} = -(\mathbf{B}'\mathbf{S}^{-2}\mathbf{B} + \mathbf{R}^{-2} + \mathbf{V}^{-2})^{-1}\widehat{\boldsymbol{\beta}}_{0;-0} \tag{6.16}$$

where $\mathbf{S}$, $\mathbf{R}$, and $\mathbf{V}$ are diagonal matrixes with as main-diagonal elements the current estimated slack vectors $\mathbf{s}$, $\mathbf{r}$, and $\mathbf{v}$ in Eq. (6.15). Note that $\widehat{\boldsymbol{\beta}}_{0;-0}$ in Eq. (6.16) is the estimated steepest ascent direction in basic RSM. As the value of a slack variable in Eq. (6.16) decreases—so the corresponding constraint gets tighter—the GRSM search direction deviates more from the steepest descent direction. Possible singularity of the various matrices in Eq. (6.16) is discussed in Angün (2004).

Following the GRSM direction defined by Eq. (6.16), we must decide on the *step size* (say) λ along this path. Angün et al. (2009) selects

$$\lambda = 0.8 \min \left[\frac{c_{h'} - \widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{z}_c}{\widehat{\boldsymbol{\beta}}'_{h';-0}\mathbf{d}} \right] \tag{6.17}$$

where the factor 0.8 decreases the probability that the *local* metamodel in Eq. (6.14) is misleading when applied *globally*; $\mathbf{z}_c$ denotes the current (see the subscript c) input combination.

Combining the search direction in Eq. (6.16) and the step size in Eq. (6.17) gives the new combination $\mathbf{z}_c + \lambda\mathbf{d}$. The box constraints in Eq. (6.10) for the deterministic inputs hold globally, so it is easy to check whether this new combination $\mathbf{z}_c + \lambda\mathbf{d}$ satisfies these constraints.

Analogously to basic RSM, GRSM proceeds *stepwise*. After each step along the search path, GRSM tests the following two null-hypotheses $H_0^{(1)}$ and $H_0^{(2)}$:

1. Pessimistic null-hypothesis: $w_0(\mathbf{z}_c+\lambda\mathbf{d})$ (output of new combination) is *no improvement* over $w_0(\mathbf{z}_c)$ (output of old combination):

$$H_0^{(1)} : E[w_0(\mathbf{z}_c + \lambda\mathbf{d})] \geq E[w_0(\mathbf{z}_c)]. \tag{6.18}$$

2. Optimistic null-hypothesis: this step is *feasible*; i.e., $w_{h'}(\mathbf{z}_c + \lambda\mathbf{d})$ satisfies the $(r-1)$ constraints in Eq. (6.9):

$$H_0^{(2)} : E[w_{h'}(\mathbf{z}_c + \lambda\mathbf{d})] \geq a_{h'} \quad \text{with } h' = 1, \ldots, r-1. \tag{6.19}$$

To test these two hypotheses, we may apply the following simple statistical procedures; more complicated parametric bootstrapping is used in Angün (2004), permitting *non*normality and testing the *relative* improvement $w_0(\mathbf{z}_c + \lambda\mathbf{d})/w_0(\mathbf{z}_c)$ and slacks $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$.

Exercise 6.3 *Which statistical problem arises when testing the ratio of the slack at the new solution and the slack at the old solution,* $s_{h'}(\mathbf{z}_c + \lambda\mathbf{d})/s_{h'}(\mathbf{z}_c)$*?*

To test $H_0^{(1)}$ defined in Eq. (6.18), we apply the paired Student statistic t_{m-1}; we use the "paired" statistic because we assume that CRN are used. We reject the hypothesis if significant improvement is observed. To test $H_0^{(2)}$ in Eq. (6.19), we again apply a t_{m-1} -statistic; because we test multiple hypotheses, we apply Bonferroni's inequality so we divide the classic α value by $(r-1)$ (number of tests).

Actually, a better solution may lie somewhere between $\mathbf{z}_c$ (old combination) and $\mathbf{z}_c + \lambda\mathbf{d}$ (new combination). Therefore GRSM uses *binary search*; i.e., GRSM simulates a combination that lies halfway between these two combinations—and is still on the search path. This halving of the step size may be applied several times; also see Fig. 6.1.

Next, GRSM proceeds analogously to basic RSM; i.e., around the best combination found so far, GRSM selects a new local area. Again a R-III design specifies the new simulation input combinations, and r first-order polynomials are fitted, which gives a *new* search direction, etc. Note that we might use the m replications $\widehat{\boldsymbol{\beta}}_r$ to estimate the accuracy of the search direction; to test the accuracy of the estimated optimum, we shall present a test in the next subsection.

Now we discuss Fig. 6.1 in more detail. This plot illustrates GRSM for a problem with simple known test functions (in practice, we use simulation to estimate the true outputs of the various implicit I/O functions of the simulation model). This plot shows two inputs, corresponding to the two axes labeled z_1 and z_2. Because the goal function is to be minimized, the plot shows two *contour plots* or *iso-costs functions* defined by $E(w_0) = a_{0;1}$ and $E(w_0) = a_{0;2}$ with $a_{0;2} < a_{0;1}$. The plot also shows two constraints; namely, $E(w_1) = a_1$ and $E(w_2) = a_2$. The search starts in the lower-right local area of the plot, using a 2^2 design; see the four elongated points. Together with the replications that are not shown, the I/O data give the search direction that is shown by the arrow leaving from point (0). The maximum step-size along this path takes the search from point (0) to point (1). The binary search takes the search back to point (2), and next to point (3). Because the best point so far turns out to be point (1), the 2^2 design is again used to select four points in this new local area; point (1) is selected as one of these four points. Simulation of the four points of this 2^2 design gives a new search direction, which indeed avoids the boundary. The maximum step-size now takes the search to point (4). The binary search takes the search back to point (5), and next to point (6). Because the best point so far turns out to be point (4), the 2^2 design is simulated in a new local area with point (4) as one of its points. A new search direction is estimated, etc.

Angün (2004) gives details on two examples, illustrating and evaluating GRSM. One example is an inventory simulation with a service-level constraint specified in Bashyam and Fu (1998); no analytical solution is known. The other example is a test function with a known solution. The results for these examples are encouraging, as GRSM finds solutions that are both feasible and give low values for the goal functions. Leijen (2011) applies GRSM to a bottle-packaging line at Heineken with nine inputs and one stochastic output constraint besides several deterministic input constraints; the analysis of the solutions generated by GRSM indicates that GRSM can find good estimates of the optimum. Mahdavi et al. (2010) applies GRSM to a job-shop manufacturing system. We shall briefly return to GRSM when discussing Eq. (6.35).

Exercise 6.4 *Apply GRSM to the following artificial example reproduced from Angün et al. (2009):*

$$\begin{array}{ll} \textit{Minimize} & E[5(z_1-1)^2+(z_2-5)^2+4z_1z_2+e_0] \\ \textit{subject to} & E[(z_1-3)^2+z_2^2+z_1z_2+e_1] \leq 4 \\ & E[z_1^2+3\,(z_2+1.061)^2+e_2] \leq 9 \\ & 0 \leq z_1 \leq 3,\ -2 \leq z_2 \leq 1 \end{array} \tag{6.20}$$

where e_0, e_1, and e_2 are the components of a multivariate normal variate with mean $\mathbf{0}$, variances $\sigma_{0;0} = 1$ (so $\sigma_0 = 1$), $\sigma_{1;1} = 0.0225$ (so $\sigma_1 = 0.15$), and $\sigma_{2;2} = 0.16$ (so $\sigma_2 = 0.4$), and correlations $\rho_{0;1} = 0.6$, $\rho_{0;2} = 0.3$, $\rho_{1;2} = -0.1$.

6.2.5 Testing a GRSM Optimum: Karush-Kuhn-Tucker (KKT) conditions

Obviously, it is uncertain whether the optimum estimated by the GRSM heuristic is close enough to the true optimum. In *deterministic* nonlinear mathematical programming, the first-order necessary optimality-conditions are known as the KKT conditions; see Gill et al. (2000). First we present the basic idea behind these conditions; next, we explain how to test these conditions in random simulation.

To explain the basic idea of the KKT conditions, we use Fig. 6.2 that illustrates the same type of problem as the one in Fig. 6.1. Figure 6.2 shows a goal function $E(w_0)$ with three contour plots that correspond with the threshold values 66, 76, and 96; also see Eq. (6.8). Furthermore, there are two constrained simulation outputs; namely, $E(w_1) \geq 4$ and $E(w_2) \geq 9$; also see Eq. (6.9). So, the plot shows the boundaries of the feasible area that is determined by the equalities $E(w_1) = 4$ and $E(w_2) = 9$. Obviously, the optimum combination is point A. The two points B and C lie on the same boundary; namely, the boundary $E(w_2) = 9$. Point D lies on the other boundary; namely, the boundary $E(w_1) = 4$. Obviously, the optimal point A and the point D lie far away from each other. The plot also displays the local gradients at the four points A through D for the goal function and for the *binding constraint*, which is the constraint with a zero slack value in Eq. (6.9). These gradients are *perpendicular* to the local tangent lines; those lines are shown only for the binding constraint—not for the goal function. These tangent lines are first-order polynomials; see Eq. (6.11). (Obviously, the estimated gradient is biased if second-order effects are important and yet a first-order polynomial is fitted.)

Note: There is a certain constraint qualification that is relevant when there are nonlinear constraints in the problem; see Gill et al. (2000, p. 81). There are several types of constraint qualification, but many are only of theoretical interest; a practical constraint qualification for nonlinear constraints is that the $r-1$ constraint gradients at the locally optimal combination be linearly independent.

Now we present the statistical procedure for testing the KKT conditions in random simulation that was derived in Bettonvil et al. (2009). Before we shall discuss the technical details of this procedure, we point out that the empirical results for this procedure are encouraging; i.e., the classic t-test for zero slacks performs as expected and the new bootstrap tests give observed type-I error rates close to the prespecified (nominal) rates, while the type-II error rate decreases as the tested input combination is farther away from the true optimum; see the points A through D in Fig. 6.2.

Note: We add that Kasaie et al. (2009) also applies this procedure to an agent-based simulation model of epidemics; this model is also discussed in Kasaie and Kelton (2013). Furthermore, Wan and Li (2008) applies the asymptotic variant of this procedure to the (s, S) inventory problem formulated in Bashyam and Fu (1998) with good results.

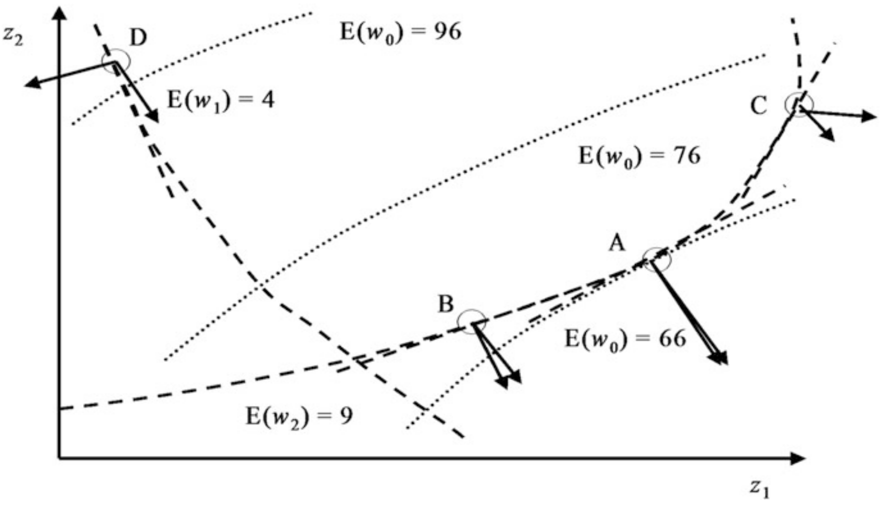


FIGURE 6.2. A constrained nonlinear random optimization problem: three contour plots with goal values 66, 76, and 96; two other outputs with lower bounds 4 and 9; optimal point A; points B and C on bound 9; point D on bound 4; local gradients at A through D for goal function and binding constraint, perpendicular to local tangent lines for binding constraint

Let $\mathbf{z}_o$ denote the input combination that gives a local minimum (or optimum; see the subscript o) for the deterministic variant of the problem defined in Eq. (6.8) through Eq. (6.10). The KKT conditions for $\mathbf{z}_o$ are then(besides some regularity conditions)

$$\begin{aligned} &\boldsymbol{\beta}_{0;-0} = \sum_{h \in A(\mathbf{z}_o)} \lambda_h \boldsymbol{\beta}_{h;-0} \\ &\lambda_h \geq 0 \\ &h \in A(\mathbf{z}_o) \end{aligned} \tag{6.21}$$

where $\boldsymbol{\beta}_{0;-0}$ denotes the k-dimensional vector with the gradient of the goal function, as we have already seen in Eq. (6.13); $A(\mathbf{z}_o)$ is the index set with the indices of those constraints that are binding at $\mathbf{z}_o$; λ_h is the Lagrangian multiplier for binding constraint h; $\boldsymbol{\beta}_{h;-0}$ is the gradient of the output in that binding constraint. Now we give two examples illustrating that Eq. (6.21) implies that the gradient of the objective is a nonnegative linear combination of the gradients of the binding constraints, at $\mathbf{z}_o$.

Example 6.1 *Figure 6.2 has only one binding constraint at the point A, so Eq. (6.21) then stipulates that the goal gradient $\boldsymbol{\beta}_{0;-0}$ and the gradient of the output with a binding constraint (namely, output $h = 2$) are two*

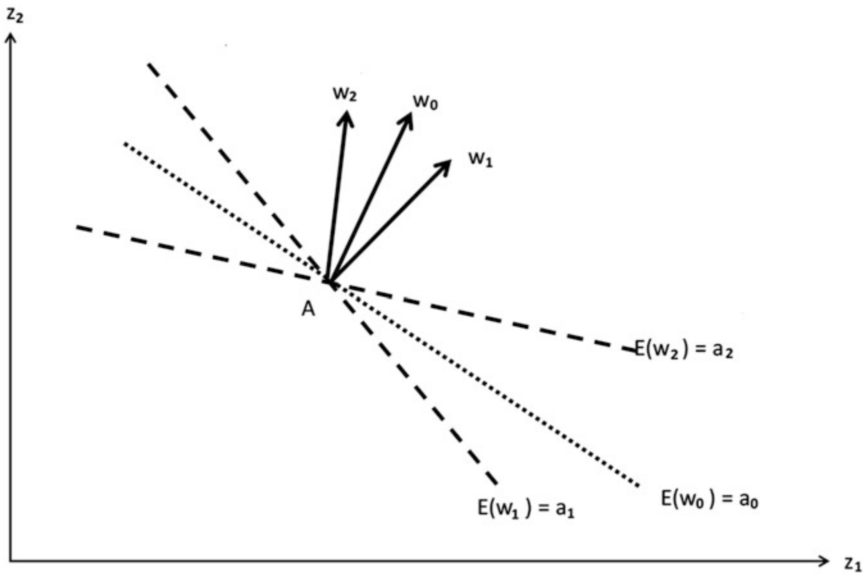


FIGURE 6.3. A LP problem: one contour line with goal value $w_0 = a_0$; two other outputs with upper bounds a_1 and a_2; optimal point A; local gradients at A for goal function and two binding constraints

vectors that point in the same direction. Indeed, point B has two gradients that point in different but similar directions—and so does C—whereas D has two gradients that point in completely different directions.

Example 6.2 *Figure 6.3 is actually a linear programming (LP) problem. One contour line for the goal output w_0 shows the input combination (z_1, z_2) that result in $w_0(z_1, z_2) = a_0$; the two other outputs are w_1 and w_2, which should satisfy the constraints $w_1 \leq a_1$ and $w_2 \leq a_2$; point A is the optimal input combination $\mathbf{z}_o$; the local gradients at point A are displayed for the goal function and the two binding constraints. Obviously, the goal gradient is a linear combination with positive coefficients of the two other gradients.*

Note: If the optimum occurs *inside* the feasible area, then there are no binding constraints so the KKT conditions reduce to the condition that the goal gradient be zero. Basic RSM includes tests for a zero gradient estimated from a second-order polynomial; see again Sect. 6.2.1.

In random simulation we must *estimate* the gradients; moreover, to check which constraints are binding, we must estimate the slacks of the constraints. This estimation changes the KKT conditions into a problem of nonlinear statistics. An asymptotic test is presented in Angün (2004), using the so-called *Delta method* and a generalized form of the so-called *Wald statistic*. A small-sample bootstrap test is presented in Bettonvil et al. (2009), which we now present because it is simpler and it suits expensive simulation. Nevertheless, this bootstrap test is still rather complicated, so readers may skip to the next section (Sect. 6.3, on Kriging for optimization)—without lessening their understanding of the rest of this book.

As in basic RSM, we assume *locally constant variances and covariances* for each of the r simulation outputs w_h $(h = 0, 1, \ldots, r-1)$. OLS per univariate simulation output gives $\widehat{\boldsymbol{\beta}}_h$ defined in Eq. (6.12). These estimators have the following estimated covariance matrix:

$$\widehat{\boldsymbol{\Sigma}}_{\widehat{\boldsymbol{\beta}}_h, \widehat{\boldsymbol{\beta}}_{h'}} = \widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}} \otimes (\mathbf{Z}'\mathbf{Z})^{-1} \; (h, h' = 0, \ldots, r-1) \tag{6.22}$$

where $\otimes$ denotes the *Kronecker product* and $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is the $r \times r$ matrix with the classic estimators of the (co)variances based on the m replications at the local center so the replication number l runs from 1 through m (we use the symbol l instead of our usual symbol r, because r now stands for the number of output types); so $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is defined by

$$\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}} = (\widehat{\sigma}_{h;h'}) = (\frac{\sum_{l=1}^{m} w_{h;l} - \overline{w}_h)(w_{h';l} - \overline{w}_{h'})}{m-1}). \tag{6.23}$$

The Kronecker product implies that $\widehat{\boldsymbol{\Sigma}}_{\widehat{\boldsymbol{\beta}}_h, \widehat{\boldsymbol{\beta}}_{h'}}$ is an $rq \times rq$ matrix where q denotes the number of regression parameters (so $q = 1 + k$ in a first-order polynomial); this matrix is formed from the $r \times r$ matrix $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ by multiplying each of its elements by the entire $q \times q$ matrix $(\mathbf{Z}'\mathbf{Z})^{-1}$ (e.g., $\mathbf{Z}$ is an $N \times (1 + k)$ matrix in Eq. (6.5)). The matrix $\widehat{\boldsymbol{\Sigma}}_{w_h, w_{h'}}$ is singular if $m \leq r$; e.g., the case study in Kleijnen (1993) has $r = 2$ output types and $k = 14$ inputs so $m \geq 3$ replications of the center point are required. Of course, the higher m is, the higher is the power of the tests that use these replications. Bettonvil et al. (2009) does not consider cases with all n local points replicated or with CRN; these cases require further research.

Basic RSM (explained in Sect. 6.2.1) assumes that the output is Gaussian, and now in GRSM we assume that the r-variate simulation output is *multivariate Gaussian*. We use the *center* point to test whether a constraint is binding in the current local area, because this point is more representative of the local behavior than the extreme points of the R-III design applied in this area. To save simulation runs, we should start a local experiment at its center point including replications; if it turns out that either no constraint is binding or at least one constraint is violated in Eq. (6.24) below, then we do not need to test the other two hypotheses given in Eq. (6.25) and Eq. (6.26) and we do not need to simulate the remainder of the local design.

Like we do in basic RSM, we should test the *validity* of the local metamodel. GRSM assumes multiple outputs, so we may apply *Bonferroni's inequality.* If we reject a metamodel, then we have two options:

- Decrease the local area; e.g., halve the range of each input.

- Increase the order of the polynomial; e.g., switch from a first-order to a second-order polynomial.

We do not explore these options any further, but refer back to Sect. 6.2.2.

To test the KKT conditions, we test the following three null-hypotheses denoted by the superscripts (1) through (3):

1. The current solution is feasible and at least one constraint is binding; see Eq. (6.9):

$$H_0^{(1)} : E(w_{h'} | \mathbf{x} = \mathbf{0}) = a_{h'} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.24}$$

   where $\mathbf{x} = \mathbf{0}$ corresponds with the center of the local area expressed in the standardized inputs.

2. The expected value of the estimated goal gradient may be expressed as the expected value of a linear combination of the estimated gradients of the simulation outputs in the binding constraints; i.e., in Eq. (6.21) we replace the deterministic quantities by their estimators:

$$H_0^{(2)} : E(\widehat{\boldsymbol{\beta}}_{0;-0}) = E(\sum_{h \in A(\mathbf{z}_o)} \widehat{\lambda}_h \widehat{\boldsymbol{\beta}}_h). \tag{6.25}$$

3. The Lagrangian multipliers in Eq. (6.25) are nonnegative:

$$H_0^{(3)} : E(\widehat{\boldsymbol{\lambda}}) \geq \mathbf{0}. \tag{6.26}$$

Each of these three hypotheses requires multiple tests, so we apply Bonferroni's inequality. Moreover, we test these three hypotheses sequentially, so it is hard to control the final type-I and type-II error probabilities (basic RSM has the same type of problem, but that RSM has nevertheless acquired a track record in practice).

Sub 1: To test $H_0^{(1)}$ in Eq. (6.24), we use the classic t-statistic:

$$t_{m-1}^{(h')} = \frac{\overline{w}_{h'}(\mathbf{x} = \mathbf{0}) - a_{h'}}{\sqrt{\widehat{\sigma}_{h';h'}/m}} \quad \text{with } h' = 1, \ldots, r-1 \tag{6.27}$$

where both the numerator and the denominator use the m replications at the local center point; see Eq. (6.23). This t-statistic may give the following three results:

(i) The statistic is *significantly positive*; i.e., the constraint for output h' is not binding. If none of the $(r-1)$ constraints is binding, then we have not yet found the optimal solution—assuming that at the optimum at least one constraint is binding; otherwise, we apply basic RSM. The search for better solutions continues; see again Sect. 6.2.4.

(ii) The statistic is *significantly negative*; i.e., the current local area does not give feasible solutions so we have not yet found the optimal solution. The search should back-up into the feasible area.

(iii) The statistic is *nonsignificant*; i.e., the current local area gives feasible solutions and the constraint for output h' is binding. We should then include the index of this gradient in $A\,(\mathbf{z}_o)$; see Eq. (6.25). And the KKT test proceeds as follows.

Sub 2 and 3: To estimate the *linear* combination in Eq. (6.25), we apply OLS with as explanatory variables the estimated gradients of the (say) J binding constraints; obviously, these explanatory variables are random. We collect these J estimated gradients in the $k \times J$ matrix $\widehat{\mathbf{B}}_{J;-0}$. These explanatory variables have linear weights $\boldsymbol{\lambda}$ that equal the parameters that are estimated through OLS, denoted by $\widehat{\boldsymbol{\lambda}}$. Let $\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0}$ denote the OLS estimator of the goal gradient, so

$$\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}(\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0} = \widehat{\mathbf{B}}_{J;-0}\widehat{\boldsymbol{\lambda}} \tag{6.28}$$

with $\widehat{\boldsymbol{\lambda}} = (\widehat{\mathbf{B}}'_{J;-0}\widehat{\mathbf{B}}_{J;-0})^{-1}\widehat{\mathbf{B}}'_{J;-0}\widehat{\boldsymbol{\beta}}_{0;-0}$; also see the general formula for OLS in Eq. (2.13). To quantify the *validity* of this linear approximation, we use the k-dimensional vector with the residuals

$$\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}_{0;-0} - \widehat{\boldsymbol{\beta}}_{0;-0}. \tag{6.29}$$

$H_0^{(2)}$ in Eq. (6.25) implies that $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0})$ in Eq. (6.29) should satisfy $E[\widehat{\mathbf{e}}$ $(\widehat{\widehat{\boldsymbol{\beta}}}_{0;-0})] = \mathbf{0}$. Furthermore, $H_0^{(2)}$ involves a product of multivariates, so standard tests do not apply; therefor we use *bootstrapping*. We do not apply distribution-free bootstrapping, because in expensive simulation only the center point is replicated a few times. Instead, we apply *parametric bootstrapping*; i.e., we assume a Gaussian distribution (like we do in basic RSM), and we estimate its parameters from the simulation's I/O data. The resulting bootstrap algorithm consists of the following four steps, where the superscript $*$ is the usual symbol for a bootstrapped value.

Algorithm 6.2

1. Use the Monte Carlo method to sample

$$vec(\widehat{\boldsymbol{\beta}}^{*}_{0;-0}, \widehat{\mathbf{B}}^{*}_{J;-0}) \sim N(vec(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0}), \widehat{\boldsymbol{\Sigma}}_{vec(\widehat{\boldsymbol{\beta}}_{0;-0},\widehat{\mathbf{B}}_{J;-0})}) \tag{6.30}$$

where $\mathrm{vec}(\widehat{\boldsymbol{\beta}}^{*}_{0;-0}, \widehat{\mathbf{B}}^{*}_{J;-0})$ is a $(k + kJ)$-dimensional vector formed by stapling (stacking) the estimated k-dimensional goal gradient vector and the J k-dimensional vectors of the $k \times J$ matrix $\widehat{\mathbf{B}}^{*}_{J;-0}$; $\mathrm{vec}(\widehat{\boldsymbol{\beta}}_{0;-0}, \widehat{\mathbf{B}}_{J;-0})$ is defined analogously to $\mathrm{vec}(\widehat{\boldsymbol{\beta}}^{*}_{0;-0}, \widehat{\mathbf{B}}^{*}_{J;-0})$ but uses

Eq. (6.12), and $\widehat{\boldsymbol{\Sigma}}_{\mathrm{vec}(\widehat{\boldsymbol{\beta}}_{0;-0},\widehat{\mathbf{B}}_{J;-0})}$ is the $(k+kJ)\times(k+kJ)$ matrix computed through Eq. (6.22).

2. Use the bootstrap values sampled in step 1 to compute the OLS estimate of the bootstrapped goal gradient where this OLS uses the bootstrapped gradients of the binding constraints as explanatory variables; i.e., use Eq. (6.28) adding the superscript $*$ to all random variables resulting in $\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ and $\widehat{\boldsymbol{\lambda}}^{*}$.

3. Use $\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ from step 2 and $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$ from step 1 to compute the bootstrap residual $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}) = \widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0}$ - $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$, analogously to Eq. (6.29); if any of the bootstrapped Lagrangian multipliers $\widehat{\boldsymbol{\lambda}}^{*}$ found in step 2 is negative, then increase the counter (say) c^{*} with the value 1.

4. Repeat the preceding three steps (say) 1,000 times, to obtain the estimated density function (EDF) of $\widehat{\mathbf{e}}(\widehat{\widehat{\boldsymbol{\beta}}}^{*}_{0;-0;j})$—which denotes the bootstrapped residuals per input j $(j = 1, \ldots, k)$—and the final value of the counter c^{*}. Reject $H_0^{(2)}$ in Eq. (6.25) if this EDF implies a two-sided $(1 - \alpha/(2k))$ CI that does not cover the value 0, where the factor k is explained by Bonferroni's inequality. Reject $H_0^{(3)}$ in Eq. (6.26) if the fraction $c^{*}/1{,}000$ is significantly higher than 50 %. To test the fraction $c^{*}/1{,}000$, approximate the binomial distribution through the normal distribution with mean 0.50 and variance $(0.50\times 0.50)/1{,}000 = 0.00025$.
   Comment: If the true Lagrangian multiplier is only "slightly" larger than zero, then "nearly" 50 % of the bootstrapped values is negative.

Altogether, this KKT test-procedure uses the following three models:

1. The *simulation* model, which is treated as a black box in GRSM.

2. The *regression* metamodel, which uses the simulation I/O data $(\mathbf{Z}, \mathbf{w})$ as input and gives the estimates of the gradients for the goal response $(\widehat{\boldsymbol{\beta}}_{0;-0})$ and the constrained responses with binding constraints $(\widehat{\mathbf{B}}_{J;-0})$. The regression analysis also gives the estimator $\widehat{\boldsymbol{\Sigma}}_{\mathrm{vec}(\widehat{\boldsymbol{\beta}}_{0;-0},\widehat{\mathbf{B}}_{J;-0})}$ (estimated covariance matrix of estimated gradients).

3. The *bootstrap* model, which uses the regression output $(\widehat{\boldsymbol{\beta}}_{0;-0}$, $\widehat{\mathbf{B}}_{J;-0}$, $\widehat{\boldsymbol{\Sigma}}_{\mathrm{vec}\widehat{\boldsymbol{\beta}}_{(0;-0},\widehat{\mathbf{B}}_{J;-0)}}$ as parameters of the multivariate normal distribution of its output $\widehat{\boldsymbol{\beta}}^{*}_{0;-0}$ and $\widehat{\mathbf{B}}^{*}_{J;-0}$.

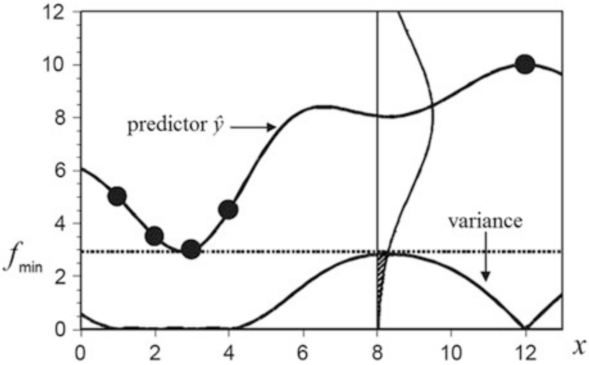


FIGURE 6.4. Expected improvement (EI) at $x = 8$: see *shaded area*; five observations on $f(x)$: see *dots*; Kriging predictor $\widehat{y}$ and variance of $\widehat{y}$

6.3 Kriging Metamodels for Optimization

In Sect. 6.2 we discussed optimization through RSM, which uses linear regression metamodels; namely, first-order and second-order polynomials fitted locally. Now we discuss optimization through Kriging metamodels, which are fitted globally. In Sect. 6.3.1 we shall discuss so-called *efficient global optimization* (EGO), which was originally developed for the minimization of the unconstrained output of a deterministic simulation model. In Sect. 6.3.2 we shall discuss constrained optimization in random simulation, using a combination of Kriging and *integer mathematical programming* (IMP) called KrIMP. We shall use the symbol x (not z) to denote the input (ignoring standardization), as the Kriging literature usually does.

6.3.1 Efficient Global Optimization (EGO)

EGO is a well-known *sequential* method; i.e., EGO selects the next input combination or "point" as experimental I/O results become available.Typically, EGO balances *local* and *global* search; i.e., EGO combines *exploitation* and *exploration.* More precisely, when selecting a new point, EGO estimates the maximum of the *expected improvement* (EI) comparing this new point and the best point that was found so far. EI uses the global Kriging metamodel to predict the output of a new point, while accounting for the predictor variance; this variance increases as a new point does not lie in a local subarea formed by some old points; also see Fig. 6.4. Obviously, EI is large if either the predicted value $\widehat{y}$ is much smaller than the minimum found so far denoted by $f_{\min} = \min w(\mathbf{x}_i)$, or the estimated predictor variance $\widehat{\sigma}(\mathbf{x})$ is large so the prediction shows much uncertainty. We shall further explain and formalize EGO in Algorithm 6.3 below.

The classic reference for EGO is Jones et al. (1998), which includes references to older publications that inspired EGO. In practice, EGO has shown to perform well when optimizing the unconstrained output of a deterministic simulation model; its theoretical convergence properties are analyzed in Bull (2011) and Vazquez and Bect (2010). EGO has also been implemented in software; see

http://cran.r-project.org/web/packages/DiceOptim/index.html.

We present only the *basic* EGO algorithm. There are many *variants* of EGO for deterministic and random simulations, constrained optimization, multi-objective optimization including Pareto frontiers, the "admissible set" or "excursion set", robust optimization, estimation of a quantile (instead of the mean), and Bayesian approaches.

Note: For these variants we list only the most recent publications plus some classic publications: Binois et al. (2015), Chevalier et al. (2014), Davis and Ierapetritou (2009), Feng et al. (2015), Forrester and Jones (2008), Forrester and Keane (2009), Forrester et al. (2008, pp. 90–101, 125–131, 141–153), Frazier (2010), Frazier et al. (2009), Gano et al. (2006), Gorissen (2010), Gramacy et al. (2015), Gramacy and Lee (2010), Huang et al. (2006), Jala et al. (2014), Jalali and van Nieuwenhuyse (2014), Janusevskis and Le Riche (2013), Kleijnen et al. (2012), Koch et al. (2015), Marzat et al. (2013), Mehdad and Kleijnen (2015), Morales-Enciso and Branke (2015), Müller and Shoemaker (2014), Nakayama et al. (2009), Picheny et al. (2013a), Picheny et al. (2013b), Preuss et al. (2012), Quan et al. (2013), Razavi et al. (2012), Regis (2014), Roustant et al. (2012), Salemi et al. (2014), Sasena et al. (2002), Scott et al. (2011), Scott et al. (2010), Sun et al. (2014), Svenson and Santner (2010), Tajbakhsh et al. (2013), Tong et al. (2015), Ur Rehman et al. (2014), Villemonteix et al. (2009a), Villemonteix et al. (2009b), Wagner (2013), Wiebenga (2014), and Williams et al. (2010).

We present a basic EGO algorithm for minimizing w, which denotes the output of a given deterministic simulation model. Our algorithm consists of the following five steps.

Algorithm 6.3

1. Fit a Kriging metamodel $y(\mathbf{x})$ to the old I/O simulation data $(\mathbf{X}, \mathbf{w})$. Comment: In Sect. 5.2 we presented details on Kriging metamodels for deterministic simulation, where $\mathbf{X}$ denoted the $n \times k$ matrix with the n combinations of the k simulation inputs, $\mathbf{w}$ denoted the n-dimensional vector with simulation outputs, and we speak of n "old" I/O data and a "new" input combination that is yet to be simulated.

2. Find the minimum output simulated so far: $f_{\min} = \min_{1 \le i \le n} w(\mathbf{x}_i)$.

3. Defining EI at a point $\mathbf{x}$ as

$$\mathrm{EI}(\mathbf{x}) = E\left[\max\left(f_{\min} - y(\mathbf{x}), 0\right)\right], \tag{6.31}$$

Jones et al. (1998) derives the following closed-form expression for its estimate:

$$\widehat{\mathrm{EI}}(\mathbf{x}) = (f_{\min} - \widehat{y}(\mathbf{x}))\, \Phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{\widehat{\sigma}(\mathbf{x})}\right) + \widehat{\sigma}(\mathbf{x})\phi\left(\frac{f_{\min} - \widehat{y}(\mathbf{x})}{\widehat{\sigma}(\mathbf{x})}\right) \tag{6.32}$$

where $\widehat{y}(\mathbf{x})$ is the Kriging predictor with plugged-in estimates defined in Eq. (5.19); $\widehat{y}(\mathbf{x})$ is assumed to be normally distributed with mean $\widehat{y}(\mathbf{x})$ and standard deviation $\widehat{\sigma}(\mathbf{x})$ which is the square root of $\widehat{\sigma}^2(\mathbf{x})$; Φ and ϕ are the usual symbols for the cumulative distribution function and probability density function of the "standard" normal variable, which has zero mean and unit variance. Using Eq. (6.32), find $\widehat{\mathbf{x}}_o$, which denotes the estimate of $\mathbf{x}$ that maximizes $\widehat{\mathrm{EI}}(\mathbf{x})$.
Comment: To find the *maximizer* of Eq. (6.32), we may apply a *global optimizer* such as the *genetic algorithm* (GA) in Forrester et al. (2008, p. 78), the branch-and-bound algorithm in Jones et al. (1998), the genetic optimization using derivatives in Picheny et al. (2013b), or the evolutionary algorithm in Viana et al. (2013). Obviously, a *local* optimizer is undesirable, because $\mathrm{EI}(\mathbf{x})$ has many local optima; e.g., if $\mathbf{x} = \mathbf{x}_i$, then $\widehat{\sigma}^2(\mathbf{x}) = 0$ so $\mathrm{EI}(\mathbf{x}) = 0$. Instead of a global optimizer, we may use a set of *candidate points* selected through Latin hypercube sampling (LHS), and select the candidate point that maximizes $\widehat{\mathrm{EI}}(\mathbf{x})$; see Boukouvalas et al. (2014), Echard et al. (2011), Kleijnen and Mehdad (2013), Scott et al. (2012), and Taddy et al. (2009). Obviously, we may use *parallel* computer hardware to compute $\mathrm{EI}(\mathbf{x})$ for different candidate points $\mathbf{x}$, if we have such hardware available; also see Ginsbourger et al. (2010).

4. Run the simulation model with the input $\widehat{\mathbf{x}}_o$ found in step 3, to find the corresponding output $w(\widehat{\mathbf{x}}_o)$.

5. Fit a new Kriging metamodel to the old I/O data of step 1 and the new I/O of step 4. Update n and return to step 2 if the stopping criterion is not yet satisfied.
Comment: Sun et al. (2014) presents a fast approximation for re-estimation of the Kriging metamodel in exploitation versus exploration in discrete optimization via random simulation. Kamiński (2015) also presents several methods for avoiding re-estimation of the Kriging parameters. A stopping criterion may be $\max \widehat{\mathrm{EI}}(\mathbf{x})$ is "close" to zero. Different stopping criteria are discussed in Razavi et al. (2012), Sun et al. (2014).

DiceOptim, which is an R package, implements EGO and enables the evaluation of multiple new points instead of a single new point. For details on DiceOptim we refer to Roustant et al. (2012).

Note: Mehdad and Kleijnen (2015) considers EGO with the predictor variance estimated through either *bootstrapped* Kriging (BK) or *conditional simulation* (CS); these two methods were discussed in Sect. 5.3. Several experiments suggest that BK and CS give predicted variances that do not differ significantly from each other, but that may be significantly bigger than the classic estimate (nevertheless, BK and CS do not give CIs that are significantly better than classic Kriging). Experiments with EGO using these alternative predictor variances suggest that EGO with BK or CS may or may not perform better than classic Kriging (CK). So, EGO may not be a good heuristic if the problem becomes complicated; also see Yarotsky (2013). More precisely, EGO with a specific correlation function and the classic estimator of the Kriging predictor variance replaced by the BK or CS estimators may be a refinement that does not improve EGO drastically. We might therefore stick to CK if we accept some possible inefficiency and prefer the simple analytical computations in Eq. (6.32).

6.3.2 Kriging and Integer Mathematical Programming (KrIMP)

Kleijnen et al. (2010) derives a heuristic that is not guided by EGO, but is more related to classic operations research (OR); this heuristic is called "Kriging and integer mathematical programming (KrIMP)". The heuristic addresses constrained optimization in random simulation, but may be easily adjusted (simplified) for deterministic simulation. Applications include an (s, S) inventory system with random lead times and a service level constraint that was originally investigated in Bashyam and Fu (1998), and a complicated call-center simulation in Kelton et al. (2007), which also minimizes costs while satisfying a service constraint; moreover, the call-center simulation must satisfy a budget constraint for the deterministic inputs (namely, resources such as personnel with specific skills) and these inputs must be nonnegative integers.

These two applications are examples of the *constrained nonlinear random optimization problem* that we have already presented in Eq. (6.8) through Eq. (6.10), but that we now augment with constraints for the deterministic inputs $\mathbf{z}$ that must satisfy s constraints f_g (e.g., budget constraints), and must belong to the set of non-negative integers $\mathbf{N}$:

$$\begin{aligned} &\min_{\mathbf{x}} E(w_0|\mathbf{x}) \\ &E(w_{h'}|\mathbf{x}) \geq c_h \ (h' = 1, \ldots, r-1) \\ f_g(\mathbf{x}) &\geq c_g \ (g = 1, \ldots, s) \\ x_j &\in \mathbf{N} \ (j = 1, \ldots, d). \end{aligned} \tag{6.33}$$

To solve this problem, KrIMP combines the following three methodologies:

1. *sequentialized* DOE to specify the next simulation combination (EGO also uses a sequential design);

2. *Kriging* to analyze the simulation I/O data that result from methodology #1 (like EGO does), and obtain explicit functions for $E(w_h|\mathbf{x})$ $(h = 0, 1, \ldots, r - 1)$ instead of the implicit (black box) functions of simulation;

3. *integer nonlinear programming* (INLP) to estimate the optimal solution from the explicit Kriging metamodels that result from methodology #2; obviously INLP is a part of integer mathematical programming (IMP).

KrIMP comprises modules that use free off-the-shelf software. We may replace these modules, as we learn more about DOE, Kriging, and INLP. For example, we may replace Kriging by intrinsic Kriging (IK); we mentioned IK in Sect. 5.4. If our application has continuous inputs, then we may replace INLP by a solver that uses the gradients; these gradients are estimated by Kriging "for free", as we discussed in Sect. 5.2 (after Exercise 5.2). In future research we may adapt KrIMP for deterministic simulations with constrained multiple outputs and inputs.

Kleijnen et al. (2010) compares the results of KrIMP with those of OptQuest, which is the popular commercial heuristic embedded in discrete-event simulation software such as Arena; see Kelton et al. (2007). In the two applications mentioned above, KrIMP turns out to require fewer simulated input combinations and to give better estimated optima than OptQuest does.

Now we discuss some salient characteristics of KrIMP that are summarized in Fig. 6.5; readers may wish to skip to the next section (Sect. 6.4, on robust optimization). KrIMP simulates a new input combination and uses the augmented I/O data either to improve the Kriging metamodel or to find the optimum—similar to "exploration" and "exploitation" in EGO. The r global Kriging metamodels should be accurate enough to enable INLP to identify either infeasible points (which violate the constraints on the $r - 1$ random outputs $E(w_{h'})$) or suboptimal points (which give a too high goal output $E(w_0)$ when trying to minimize $E(w_0)$). KrIMP may add a new point throughout the entire input-feasible area, which implies exploration. The global Kriging metamodel for output w_h $(h = 0, 1, \ldots, r - 1)$ uses all observations for this output, obtained so far. To guide the INLP search, KrIMP simulates each point with a given relative precision so KrIMP is reasonably certain of the objective values and the possible violation of the constraints; i.e., KrIMP selects the number of replications m_i such that the halfwidth of the 90 % CI for the average simulation output is within

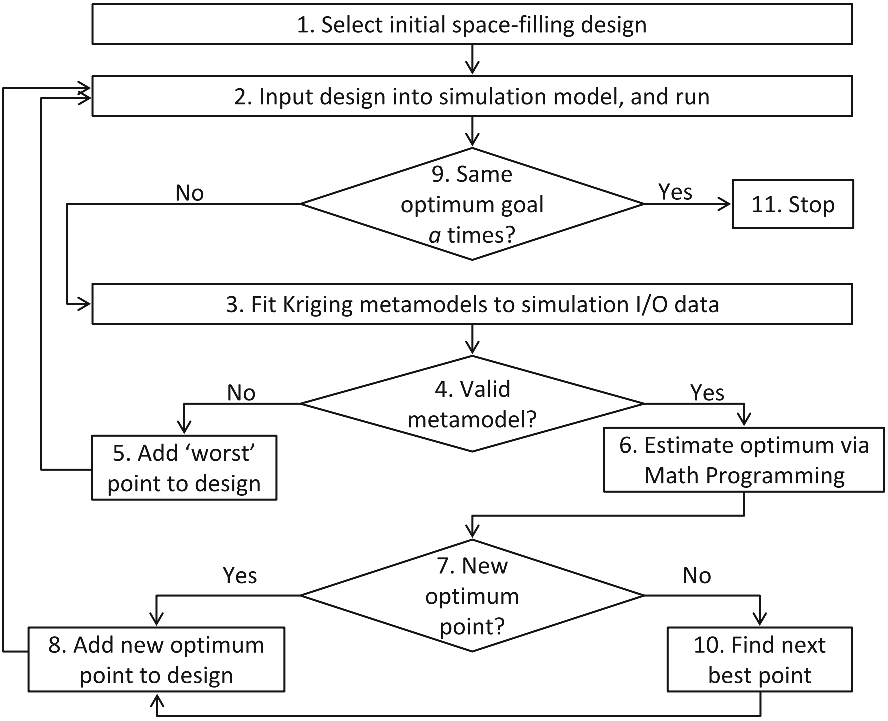

1. Select initial space-filling design
2. Input design into simulation model, and run
9. Same optimum goal a times?
No
Yes
11. Stop
3. Fit Kriging metamodels to simulation I/O data
4. Valid metamodel?
No
Yes
5. Add 'worst' point to design
6. Estimate optimum via Math Programming
7. New optimum point?
Yes
No
8. Add new optimum point to design
10. Find next best point

FIGURE 6.5. Overview of the KrIMP heuristic, combining Kriging and integer mathematical programming (IMP)

15 % of the true mean for all r outputs; also see our discussion on designs for linear regression metamodels with heterogeneous response variances, in Sect. 3.4.5. Furthermore, KrIMP uses CRN to improve the estimate of the optimum solution. KrIMP applies Kriging to the average output per simulated input combination, and does so for each of the r types of output; i.e., KrIMP does not use stochastic Kriging (SK) discussed in Sect. 5.6 and does not apply multivariate Kriging discussed in Sect. 5.10. KrIMP also uses distribution-free bootstrapping, combined with cross-validation. This bootstrapping gives an estimate of the predictor variance for output h at the deleted combination x_i, denoted by $\widehat{\sigma}^2(\widehat{y}_h^*(\mathbf{x}_i))$. Actually, the bootstrap in KrIMP accounts for multivariate (namely, r-variate) output created through CRN and for nonconstant replication numbers m_i This bootstrap and cross-validation give the following *Studentized* prediction errors for output h of deleted combination i with $i = 1, \ldots, n_{\text{cv}}$ where n_{cv} denotes the number of cross-validated combinations ($n_{\text{cv}} < n$ because

KrIMP avoids extrapolation in its Kriging):

$$t_{m_i-1}^{h,i} = \frac{\overline{w}_h(\mathbf{x}_i) - \widehat{y}_h(-\mathbf{x}_i)}{\{\widehat{\sigma}^2[\overline{w}_h(\mathbf{x}_i)] + \widehat{\sigma}^2[\widehat{y}_h^*(\mathbf{x}_i)]\}^{1/2}}$$
$$(h = 0, \ldots, r-1) \ (i = 1, \ldots, n_{\text{cv}}) \qquad (6.34)$$

where

$$\widehat{\sigma}^2[\overline{w}_h(\mathbf{x}_i)] = \frac{\widehat{\sigma}^2[w_h(\mathbf{x}_i)]}{m_i}$$

with

$$\widehat{\sigma}^2[w_h(\mathbf{x}_i)] = \frac{\sum_{r=1}^{m_i}[w_{h;r}(\mathbf{x}_i) - \overline{w}_h(\mathbf{x}_i)]^2}{m_i - 1}.$$

The highest absolute value of the $t_{m_i-1}^{h,i}$ in Eq. (6.34) over all r outputs and all n_{cv} cross-validated combinations is denoted by max $|t_{m_i-1}^{h,i}|$. Bonferroni's inequality implies that KrIMP divides the traditional type-I error rate α by $r \times n_{\text{cv}}$. If max $|t_{m_i-1}^{h,i}|$ is significant, then KrIMP rejects all r Kriging metamodels; else, KrIMP uses the metamodels in its INLP, to estimate the constrained optimum.

Actually, we think that it is not good enough that KrIMP simulates each point with a given relative precision; i.e., we think that KrIMP should treat the $r-1$ constraints $E(w_{h'}|\mathbf{x}) \geq c_h$ in Eq. (6.33)—or Eq. (6.9) in case of GRSM—more rigorously such that

$$P[\forall h' : E(w_{h'}|\mathbf{x}) \geq c_h] \leq p \qquad (6.35)$$

where p is a given small number; e.g., $p = 0.05$ is the probability that all $r-1$ constraints are satisfied. Obviously, this *chance-constrained* formulation concerns the $1-p$ quantile of the output $w_{h'}$ given the input combination $\mathbf{x}$: $P[(w_{h'}|\mathbf{x}) < c_h] = 1-p$. Similar quantiles are used in Feyzioğlu et al. (2005) applying second-order polynomials (instead of Kriging) to solve a multi-objective optimization problem (instead of a constrained optimization problem such as Eq. (6.33)); Kleijnen et al. (2011) also uses quantiles in a similar approach. Hong et al. (2015) also considers chance- constrained optimization in case of a given limited number of alternative simulated systems. Furthermore—inspired by EGO—we may adapt KrIMP such that it does not minimize the expected value $E(w_0|\mathbf{x})$ in Eq. (6.33), but it minimizes a preselected quantile—namely, the q-quantile—of the goal output: $\min_{\mathbf{x}} \ (w_{0;q}|\mathbf{x})$ where $P[(w_0|\mathbf{x}) < w_{0;q}] = q$. Obviously, if $q = 0.50$ and w_0 has a symmetric distribution (as the Gaussian assumption in Kriging implies), then $w_{0;q} = E(w_0)$. Various choices of q are discussed in Picheny et al. (2013a). Finally, to predict the joint probability in Eq. (6.35), KrIMP may use SK defined in Sect. 5.6.

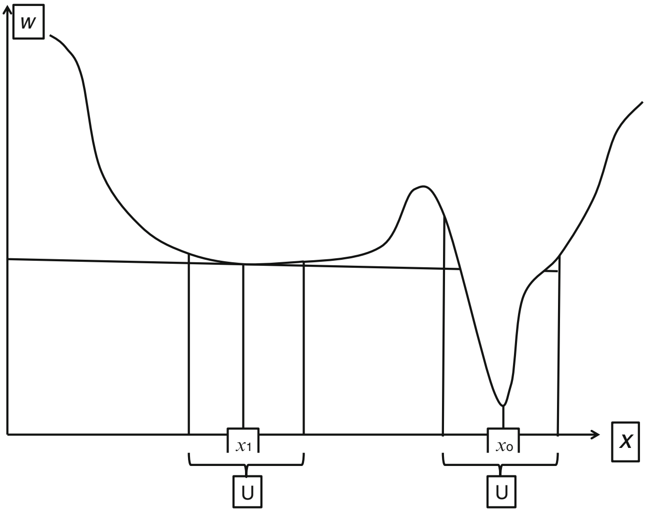


FIGURE 6.6. Robust solution x_1 in case of implementation error within range U, and nominally optimal solution x_o for simulation output $w = f_{\text{sim}}(\mathrm{x})$

6.4 Robust Optimization

We start with a simple artificial example; see Fig. 6.6. In this example we assume that an implementation error (say) e occurs when a recommended solution is realized in the system being simulated; the possible values of this error fall within a range denoted by U, so $e \in U$ where the symbol U stands for the *uncertainty set* in the mathematical programming approach to robust optimization (see the next paragraph). The "nominally" optimal solution ignores this implementation error, so in the plot the global optimum is x_o. A better solution accounting for this implementation error is x_1, which is the best worst-case or min-max solution. In Taguchian robust optimization (also introduced in the next paragraph) we assume a probability density function (PDF) for e; e.g., we assume a Gaussian PDF with a mean $E(e) = 0$ and a variance such that $x + e$—the realized value of the implemented solution—has a 99 % probability of falling within the range U around the recommended solution x. Obviously, this PDF together with the curvature of the simulation's I/O function $w = f_{\text{sim}}(x)$ implies that in this example the simulation output w has $\text{Var}(w|x_o) > \text{Var}(w|x_1)$. A Taguchian solution tries to balance the mean and the variance of the output w through a robust solution for the decision variable x.

In general, the practical importance of robust optimization is emphasized by the panel reported in Simpson et al. (2004). Indeed, we think that robustness is crucial, given today's increased uncertainty in organizations and their environment; e.g., robust optimization may guide strategic decisions on supply chains that are meant to be "agile" or "resilient". More specifically, the optimum solution for the decision variables—that we may estimate through local linear regression metamodels or global Kriging metamodels, as we explained in the preceding sections—may turn out to be inferior when ignoring uncertainties in the noncontrollable environmental variables; i.e., these uncertainties create a *risk*. Taguchi (1987) discusses "robust optimization" for the design of products. Ben-Tal and Nemirovski (1998) discusses robust optimization in mathematical programming models with uncertain coefficients.

Note: Taguchi (1987) is updated in Myers et al. (2009) and Wu and Hamada (2009). Furthermore, Ben-Tal and Nemirovski (1998) is updated in Ben-Tal and Nemirovski (2008), Gabrel et al. (2014), Wiesemann et al. (2014), and Yanikoğlu et al. (2015). Finally, robust optimization in simulation is also discussed in Hamarat et al. (2014) and Jalali and Van Nieuwenhuyse (2015). Robust decision-making is discussed in Grubler et al. (2015).

Taguchi (1987) emphasizes that in practice some inputs of a manufactured product are under complete control of the engineers, whereas other inputs are not; e.g., the design of a car engine is completely controlled by the engineers, but the driving style is not. Consequently, an engineering design—in this chapter we should distinguish between an engineering design and a statistical design—that allows some flexibility in its use is "better"; e.g., a car optimized only for the race circuit does not perform well in the city streets. Likewise, in simulation—either deterministic or random—our estimated optimum solution may be completely wrong when we ignore uncertainties in some inputs; e.g., the nominally optimal decision on the inventory control limits s (reorder level) and S (order-up-to level) may be completely wrong if we ignore the uncertainty in the parameters that we assumed for the random demand and delivery time distributions. Taguchi (1987) therefore distinguishes between two types of inputs:

- *decision variables*, which we now denote by d_j $(j = 1, \ldots, k)$ so $\mathbf{d} = (d_1, \ldots, d_k)'$, and

- *environmental inputs* or *noise factors* e_g $(g = 1, \ldots, c)$ so $\mathbf{e} = (e_1, \ldots, e_c)'$.

Note: Stinstra and Den Hertog (2008) points out that a source of uncertainty may be *implementation error*, which occurs whenever recommended values of decision variables are to be realized in practice; e.g., continuous values are hard to realize in practice, because of limited accuracy (see again Fig. 6.6). Besides implementation errors, there are validation errors of the simulation model (compared with the real system) and the metamodel (compared with the simulation model); also see the discussion on the validation of metamodels in simulation, in Kleijnen and Sargent (2000).

We perceive the following major differences between Taguchi's and Ben-Tal et al.'s approaches. Originally, Ben-Tal et al. assumed static deterministic linear problems solved by LP, whereas we assume dynamic nonlinear problems solved by either deterministic or random simulation. Ben -Tal et al. assume that uncertainty implies that the coefficients of the LP problem lie in a mathematical set called the *uncertainty set*; see the example in Fig. 6.6. We, however, assume that in deterministic or random simulation some inputs have a given statistical distribution; also see Sect. 5.9, in which we discussed risk analysis, uncertainty propagation, epistemic uncertainty, etc. Currently, Ben-Tal et al. also consider multi-stage nonlinear problems and uncertainty sets based on historical data. Another essential characteristic of simulation is that the objective and constrained functions are not known explicitly; actually, these functions are defined implicitly by the simulation model (we may replace these implicit functions by explicit metamodels, which are linear in the inputs if we use first-order polynomials or nonlinear if we use either higher-order polynomials or Kriging; metamodels treat the simulation model as a black box, as we explained in Sect. 2.1). Moreover, a random simulation model gives random outputs, which only estimate the true outputs (these outputs may be expected values or specific quantiles).

The goal of robust optimization is the design of robust products or systems, whereas the goal of *risk analysis* is to quantify the risk of a given engineering design; that design may turn out to be not robust at all. For example, Kleijnen and Gaury (2003) presents a random simulation of production-management (through methods such as Kanban, Conwip, and related methods), using RSM to estimate an optimal solution assuming a specific—namely the most likely—combination of environmental input values. Next, the robustness of this solution is estimated when the environment changes; technically, these environments are generated through LHS. In robust optimization, however, we wish to find a solution that—from the start of the analysis—accounts for all possible environments, including their likelihood; i.e., whereas Kleijnen and Gaury (2003) performs an *ex post* robustness analysis, we wish to perform an *ex ante* analysis.

Note: Whereas optimization is a "hot" topic in simulation (either deterministic or random), robust optimization is investigated in only a few publications; see the older references in Kleijnen (2008, pp. 131–132) and also Bates et al. (2006), Dengiz (2009), Kenett and Steinberg (2006), Meloni and Dellino (2015), Wiebenga (2014), and the references in the next subsections.

Next we shall discuss Taguchi's approach, using RSM in Sect. 6.4.1 and Kriging in Sect. 6.4.2; we shall discuss Ben-Tal et al.'s approach in Sect. 6.4.3.

6.4.1 Taguchian Robust Optimization Through RSM

Taguchi (1987) assumes a single output—which we denote by w—focusing on its mean μ_w and its variance caused by the noise factors $\mathbf{e}$ so $\sigma^2(w|\mathbf{d}) > 0$. These two outputs are combined in a *scalar loss function* such as the *signal-to-noise* or *mean-to-variance* ratio μ_w/σ_w^2; also see the discussion of these functions in Myers et al. (2009, pp. 486–488). Instead of this scalar function, we use both μ_w and σ_w^2 separately and formulate the following mathematical problem:

$$\min E(w|\mathbf{d}) \quad \text{such that } \sigma(w|\mathbf{d}) \leq T \tag{6.36}$$

where $E(w|\mathbf{d})$ is the mean of the simulation output w determined by the distribution function of the environmental variables $\mathbf{e}$ and controlled through the decision factors $\mathbf{d}$; the constraint concerns $\sigma(w|\mathbf{d})$, which is the standard deviation of the goal output w, and has a given upper threshold T. We also refer to Myers et al. (2009, pp. 488–495) and the surveys on robust optimization in Beyer and Sendhoff (2007) and Park et al. (2006).

Note: An alternative for the standard deviation $\sigma(w|\mathbf{d})$ in Eq. (6.36) may be the variance $\sigma^2(w|\mathbf{d})$, but the standard deviation uses the same measurement unit as the mean $(w|\mathbf{d})$. Kleijnen and Gaury (2003) uses the probability of a specific disastrous event happening; e.g., $P(w > c|\mathbf{d})$.

Taguchi's worldview has been very successful in production engineering, but statisticians have seriously criticized his statistical techniques; see the panel report in Nair (1992). To this report we add that in simulation we can experiment with many more inputs, levels (values), and combinations than we can in real-life experiments; Taguchians and many statisticians focus on real-life experiments. Myers et al. (2009, pp. 502–506) combines Taguchi's worldview with the statisticians' RSM. Whereas Myers et al. (2009) assumes that the multivariate noise $\mathbf{e}$ has the covariance matrix $\mathbf{\Omega_e} = \sigma_e^2\mathbf{I}$—and the mean $\mu_\mathbf{e}$— we assume a general $\mathbf{\Omega_e}$. Whereas Myers et al. (2009) superimposes contour plots for the mean and variance of the output to find a robust solution, we use more general and flexible mathematical programming. This mathematical programming, however, requires specification of threshold values such as T in Eq. (6.36). Unfortunately, managers may find it hard to select specific values such as T, so we may try different values and estimate the corresponding Pareto-optimal efficiency frontier. Decreasing T in Eq. (6.36) increases $E(w|\mathbf{d})$ if the constraint with the old T was binding. So, changing T gives an estimate of the Pareto-optimal efficiency frontier; i.e., $E(w|\mathbf{d})$ and $\sigma(w|\mathbf{d})$ are criteria requiring a trade-off. To estimate the variability of this frontier resulting from the various estimators, we may use bootstrapping. For details on our adaptation of the approach in Myers et al. (2009) we also refer to Dellino et al. (2010).

More precisely, Myers et al. (2009) fits a *second-order polynomial* for the decision variables $\mathbf{d}$ that are to be optimized. Possible effects of the environmental variables $\mathbf{e}$ are modelled through a first-order polynomial

	e combination			
d combination	1	2	...	n_e
1				
2				
...				
n_d				

TABLE 6.1. A crossed design combining a design for the decision variables d and a design for the environmental inputs e

in these variables **e**. *Control-by-noise* two-factor interactions (between **d** and **e**) are also considered. Altogether, the following "incomplete" second-order polynomial is fitted:

$$\begin{aligned} y &= \beta_0 + \sum_{j=1}^{k} \beta_j d_j + \sum_{j=1}^{k} \sum_{j' \geq j}^{k} \beta_{j;j'} d_j d_{j'} + \sum_{g=1}^{c} \gamma_g e_g + \sum_{j=1}^{k} \sum_{g=1}^{c} \delta_{j;g} d_j e_g + \epsilon \\ &= \beta_0 + \boldsymbol{\beta}'\mathbf{d} + \mathbf{d}'\mathbf{B}\mathbf{d} + \boldsymbol{\gamma}'\mathbf{e} + \mathbf{d}'\boldsymbol{\Delta}\mathbf{e} + \epsilon \end{aligned} \tag{6.37}$$

where we now denote the regression residual through the symbol ϵ (instead of e); we denote the first-order effects by $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)'$ for **d** and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_c)'$ for **e**; we let **B** denote the $k \times k$ symmetric matrix with on the main diagonal the purely quadratic effects $\beta_{j;j}$ of **d** and off the diagonal half the interactions $\beta_{j;j'}/2$ of **d**; and we let $\boldsymbol{\Delta}$ denote the $k \times c$ matrix with the interactions $\delta_{j;g}$ between decision variable d_j and environmental variable e_g.

If $E(\epsilon) = 0$, then Eq. (6.37) implies the following regression predictor for μ_w (true mean of output w):

$$\mu_y = \beta_0 + \boldsymbol{\beta}'\mathbf{d} + \mathbf{d}'\mathbf{B}\mathbf{d} + \boldsymbol{\gamma}'\boldsymbol{\mu}_{\mathbf{e}} + \mathbf{d}'\boldsymbol{\Delta}\boldsymbol{\mu}_{\mathbf{e}}. \tag{6.38}$$

Because the covariance matrix of the noise variables **e** is $\boldsymbol{\Omega}_{\mathbf{e}}$, the regression predictor for σ_w^2 (true variance of w) is

$$\sigma_y^2 = (\boldsymbol{\gamma}' + \mathbf{d}'\boldsymbol{\Delta})\boldsymbol{\Omega}_{\mathbf{e}}(\boldsymbol{\gamma} + \boldsymbol{\Delta}'\mathbf{d}) + \sigma_\epsilon^2 = \mathbf{l}'\boldsymbol{\Omega}_{\mathbf{e}}\mathbf{l} + \sigma_\epsilon^2 \tag{6.39}$$

where $\mathbf{l} = (\boldsymbol{\gamma} + \boldsymbol{\Delta}'\mathbf{d}) = (\partial y/\partial e_1, \ldots, \partial y/\partial e_c)'$ so **l** is the *gradient* with respect to **e**. Consequently, the larger the gradient's elements are, the larger σ_y^2 is—which stands to reason. Furthermore, if there are no control-by-noise interactions so $\boldsymbol{\Delta} = \mathbf{0}$, then we cannot control σ_y^2 through **d**.

To enable estimation of the regression parameters in Eq. (6.37), we follow the usual Taguchian approach and use a *crossed design*; i.e., we combine the design or *inner array* for **d** with n_d combinations and the design or *outer array* for **e** with n_e combinations such that the crossed design has $n_d \times n_e$ combinations as in Table 6.1. To estimate the optimal **d** through the second-order polynomial in Eq. (6.37), we use a CCD; also see again our discussion below Eq. (6.4). For the first-order polynomial in $\mathbf{e}$, we use a R-III design; see the discussion below Eq. (6.3). Obviously, the combination of these two designs enables the estimation of the two-factor interactions $\delta_{j;g}$.

Note: Designs that are more efficient than crossed designs are discussed in Dehlendorff et al. (2011), Dellino et al. (2010), Khuri and Mukhopadhyay (2010), Kolaiti and Koukouvinos (2006), and Myers et al. (2009).

To use *linear regression analysis* for the estimation of the parameters in Eq. (6.37), we reformulate that equation as

$$y = \boldsymbol{\zeta}'\mathbf{x} + \epsilon \tag{6.40}$$

with the q-dimensional vector $\boldsymbol{\zeta} = (\beta_0, \ldots, \delta_{k;c})'$ and $\mathbf{x}$ defined in the obvious way; e.g., the element corresponding with $\beta_{1;2}$ (interaction between d_1 and d_2) is $d_1 d_2$. Obviously, Eq. (6.40) is linear in $\boldsymbol{\zeta}$, but not in $\mathbf{d}$.

The OLS estimator $\widehat{\boldsymbol{\zeta}}$ of $\boldsymbol{\zeta}$ in Eq. (6.40) is

$$\widehat{\boldsymbol{\zeta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{w} \tag{6.41}$$

where $\mathbf{X}$ is the $N \times q$ matrix of explanatory variables with $N = \sum_{i=1}^{n} m_i$ where n denotes the number of different combinations of $\mathbf{d}$ and $\mathbf{e}$, and m_i denotes the number of replications in combination i (obviously, $m_i = 1$ in deterministic simulation); $\mathbf{w}$ is the vector with the N "stapled" (or "stacked") outputs $w_{i;r}$ where $r = 1, \ldots, m_i$.

The covariance matrix of the OLS estimator $\widehat{\boldsymbol{\zeta}}$ defined in Eq. (6.41) is

$$\boldsymbol{\Sigma}_{\widehat{\boldsymbol{\zeta}}} = (\mathbf{X}'\mathbf{X})^{-1}\sigma_w^2 \tag{6.42}$$

where σ_w^2 equals σ_ϵ^2 because we assume the metamodel in Eq. (6.37) to be valid and ϵ to be white nose so $\epsilon \sim \text{NIID}(0, \sigma_\epsilon^2)$. This variance is estimated by the *mean squared residuals* (MSR), which we have already defined in Eq. (2.20) and we repeat here for convenience:

$$\text{MSR} = \frac{(\widehat{\mathbf{y}} - \mathbf{w})'(\widehat{\mathbf{y}} - \mathbf{w})}{N - q} \tag{6.43}$$

where $\widehat{\mathbf{y}} = \widehat{\boldsymbol{\zeta}}'\mathbf{x}$; also see Eq. (6.6).

Note: Santos and Santos (2011) allows σ_w^2 to be nonconstant, and estimates a metamodel for σ_w—besides a metamodel for μ_w. Shin et al. (2011) also estimates one metamodel for the mean and one for the variance.

To estimate the predictor mean μ_y in the left-hand side of Eq. (6.38), we simply plug $\widehat{\boldsymbol{\zeta}}$ defined in Eq. (6.41) into the right-hand side of Eq. (6.38), which also contains the known $\mathbf{d}$ and $\boldsymbol{\mu}_{\mathbf{e}}$. We also estimate the predictor variance σ_y^2 by plugging $\widehat{\boldsymbol{\zeta}}$ into Eq. (6.39), where $\boldsymbol{\Omega}_{\mathbf{e}}$ is known. We point out that Eq. (6.39) involves products of unknown parameters, so it implies a *nonlinear* estimator $\widehat{\sigma}_y^2$; plugged-in estimators certainly create bias, but we ignore this bias.

Note: Apley and Kim (2011) follows a Bayesian approach—called "cautious robust design"—which does account for the uncertainty of the parameter estimator $\widehat{\boldsymbol{\zeta}}$, and gives an analytical (instead of a simulation) solution.

Our final goal is to solve Eq. (6.36). We solve this constrained minimization problem through a mathematical programming solver; e.g., Matlab's "fmincon"—but a different solver might be used; see Gill et al. (2000). This solution estimates the robust optimal solution for the decision variables and the resulting mean and variance.

Dellino et al. (2010) presents an example; namely, the *economic order quantity* (EOQ) for an environment with a demand rate that is uncertain—but this rate has a known distribution (implying "uncertainty propagation" of "epistemic" uncertainty; see again Sects. 1.1 and 5.9). This example demonstrates that if management prefers low variability of inventory costs, then they must pay a price; i.e., the expected costs increases. Furthermore, different values are indeed found for the robust EOQ and the classic EOQ; this classic EOQ assumes a known fixed demand rate. More examples are referenced in Yanikoğlu et al. (2015).

Note: The solution estimated through robust optimization is a nonlinear function of the simulation output so there are no standard CIs for this solution. We may therefore evaluate the reliability of the estimated solution through bootstrapping. The final decision on the preferred solution is up to management; they should select a compromise combination of the decision variables depending on their risk attitude. Shang et al. (2004) uses plots to decide on a compromise solution; also see Fig. 6.7 where the horizontal double-pointed arrows denote the (bootstrap) CIs for the optimal solutions for the mean and variance, respectively, which do not overlap in this example. However, we leave this bootstrapping for future research. We also refer to Apley and Kim (2011), discussed in the immediately preceding Note.

Note: Future research may also address the following issues. Instead of minimizing the mean under a standard-deviation constraint as in Eq. (6.36), we may minimize a specific quantile of the simulation output distribution or minimize the *conditional value at risk* (CVaR); CVaR considers only one-sided deviations from the mean (whereas the standard deviation and the variance consider deviations on both sides of the mean). Indeed, Angün (2011) replaces the standard deviation by the CVaR and considers random simulation of the (s, S) inventory system in Bashyam and Fu (1998) and the call center in Kelton et al. (2007); in case the problem is found to be convex, this problem can be solved very efficiently. Instead of Eq. (6.36), Broadie et al. (2011) estimates the probability of a large loss in financial risk management, for various "scenarios"—these scenarios correspond with the combinations of environmental variables $\mathbf{e}$ in our approach—and examines the sequential allocation of the computer budget to estimate this loss, allowing for variance heterogeneity; we also refer to Sun et al. (2011), which we shall briefly discuss in Sect. 6.4.2 (last Note). Other risk measures are the *expected shortfall*, which is popular in the actuarial literature;

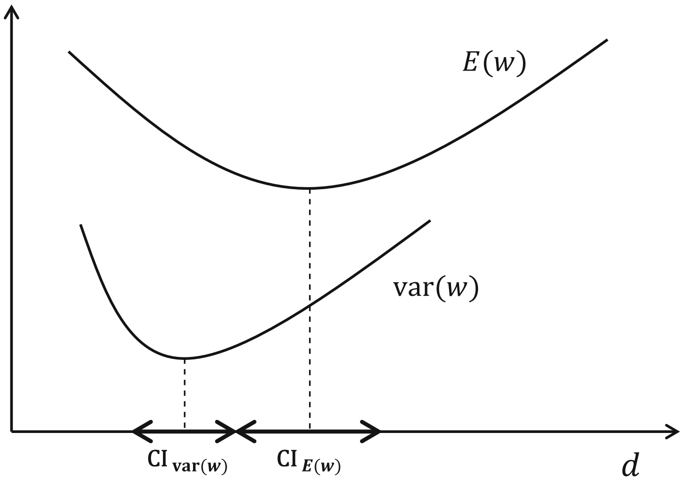


FIGURE 6.7. Example of robust optimization of a simulation model with output w, a single controllable input d; and one or more uncontrollable inputs $\mathbf{e}$ so $\mathrm{Var}(w|\mathbf{d}) > 0$

see again Angün (2011) and also Gordy and Juneja (2010) and Lan et al. (2010). Furthermore, multi-objective optimization and genetic algorithms for estimating Pareto frontiers are discussed in Koziel et al. (2014) and Shahraki and Noorossana (2014). Another methodology for estimating the Pareto frontier is developed in Shin et al. (2011), solving a bi-objective robust design problem considering two quality characteristics. Rashid et al. (2013) also presents a method for the estimation of the efficiency frontier. Ardakani and Wulff (2013) gives an extensive overview of various optimization formulations in case of multiple outputs, using a multi-objective decision-making perspective; these formulations include our Eq. (6.36), the Pareto frontier, so-called *desirability functions*, etc.; an application of this desirability function—combining two outputs into a single criterion—is presented in Yalçinkaya and Bayhan (2009).

6.4.2 Taguchian Robust Optimization Through Kriging

Dellino et al. (2012) combines the world view of Taguchi (1987) and Kriging metamodels, for robust optimization in deterministic simulation. This approach is illustrated through the EOQ example with uncertain demand rate that was also used in Dellino et al. (2010) (discussed in the preceding subsection, Sect. 6.4.1).

More precisely, Taguchi's low-order polynomial metamodels are replaced by ordinary Kriging (OK) metamodels. Moreover, bootstrapping is applied to quantify the variability in the estimated Kriging metamodels. Instead of Taguchi's signal-noise criterion μ_w/σ_w^2, now Kriging is combined with nonlinear programming (NLP) (NLP is also discussed in the subsection on KrIMP, Sect. 6.3.2). Changing the threshold values in the NLP model—that will be defined in Eq. (6.44)—enables the estimation of the Pareto frontier. The EOQ example shows that robust optimization may require an order quantity that differs from the classic EOQ (such a difference is also found through the RSM approach in Sect. 6.4.1).

Specifically, Dellino et al. (2012) uses the following NLP model:

$$\min E(w|\mathbf{d}) \text{ such that } \sigma(w|\mathbf{d}) \leq T \tag{6.44}$$

where $E(w|\mathbf{d})$ is the mean of the simulation output w determined by the distribution function of the environmental variables $\mathbf{e}$ and controlled through the decision factors $\mathbf{d}$; the constraint concerns $\sigma(w|\mathbf{d})$, which is the standard deviation of the goal output w, and has a given upper threshold T. The same problem was defined in Eq. (6.36).

Next, $E(w|\mathbf{d})$ and $\sigma(w|\mathbf{d})$ are replaced by their Kriging metamodels. Obviously, the constrained minimization problem in Eq. (6.44)—combined with the explicit Kriging approximations—is nonlinear in the decision variables $\mathbf{d}$.

We point out that we are *not* interested in the functional relationship between the output w and the environmental inputs $\mathbf{e}$; in the RSM approach—in Eq. (6.37)—we do estimate a low-order polynomial in $\mathbf{e}$ and $\mathbf{d}$. Following Taguchi (1987), we consider the inputs $\mathbf{e}$ as noise. Unlike Taguchi, we now use LHS to sample (say) n_e combinations of the environmental inputs $\mathbf{e}$. For the decision variables $\mathbf{d}$ we do not use a CCD, whereas we did use a CCD in the RSM approach in Sect. 6.4.1 (between Eqs. (6.39) and (6.40)). LHS does not impose a relationship between n_e (number of combinations of $\mathbf{e}$) and c (number of environmental inputs), as we explained in our discussion of LHS in Sect. 5.5.1. If we do not have prior information about the likelihood of specific values for $\mathbf{e}$, then we might use independent uniform distributions per environmental input e_g $(g = 1, \ldots, c)$ (also see our brief discussion of Bayesian prior distributions at the end of Sect. 5.9 on risk analysis). Whereas classic optimization assumes a single "scenario" (e.g., the most likely combination of environmental inputs), we now estimate the parameters in the Kriging metamodel for the decision variables $\mathbf{d}$ from the simulation outputs averaged over all simulated combinations of $\mathbf{e}$; these combinations are sampled through LHS accounting for the distribution of $\mathbf{e}$. We now explain this Kriging approach to Taguchian optimization, in more detail.

In general, if we wish to fit a Kriging metamodel to obtain an explicit approximation for the I/O function of a simulation model, then we often use LHS to obtain the I/O simulation data—as we have already discussed in detail in Sect. 5.5. Dellino et al. (2012) also uses LHS, as part of the following two approaches, especially developed for robust optimization:

1. Analogously to Dellino et al. (2010), fit two Kriging metamodels; namely, one model for $E(w|\mathbf{d})$ and one for $\sigma(w|\mathbf{d})$—both estimated from the *simulation* I/O data.

2. Analogously to Lee and Park (2006), fit a single Kriging metamodel to a relatively small number (say) n of combinations of $\mathbf{d}$ and $\mathbf{e}$; next use this metamodel to compute the *Kriging predictions* for the simulation output w for $N \gg n$ combinations of $\mathbf{d}$ and $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$.

First we summarize approach 1, then approach 2, and finally the two approaches together.

Sub 1: We start with selecting the input combinations for the simulation model through a *crossed* design for $\mathbf{d}$ and $\mathbf{e}$; see again Table 6.1. Such crossed designs are traditional in Taguchian design (as we discussed between Eqs. (6.39) and (6.40)). To facilitate the fitting of a Kriging metamodel in $\mathbf{d}$, we select the n_d combinations of $\mathbf{d}$ *space-filling*; e.g., we use a maximin LHS, as we discussed in Sect. 5.5.1. The n_e combinations of $\mathbf{e}$, however, we *sample* from the distribution of $\mathbf{e}$; we may use LHS for this (stratified) sampling. The resulting I/O data form an $n_d \times n_e$ matrix. Such a crossed design enables the following estimators of the n_d conditional means and variances where $i = 1, \ldots, n_d$:

$$\overline{w}_i = \frac{\sum_{j=1}^{n_e} w_{i;j}}{n_e} \quad \text{and} \quad s_i^2(w) = \frac{\sum_{j=1}^{n_e} (w_{i;j} - \overline{w}_i)^2}{n_e - 1}. \tag{6.45}$$

These two estimators are unbiased, as they do not use any metamodels.

Sub 2: We start with a relatively small number (say) n of combinations of the $k + c$ inputs $\mathbf{d}$ and $\mathbf{e}$; we select these combinations through a space-filling design (so we not yet sample $\mathbf{e}$ from its distribution). Next, we use this $n \times (k+c)$ matrix with the simulation input data and the n-dimensional vector with the corresponding simulation outputs w, to fit a Kriging metamodel that approximates w as a function of $\mathbf{d}$ and $\mathbf{e}$. Finally, we use a design with $N \gg n$ combinations, crossing a space-filling design with N_d combinations of $\mathbf{d}$ and LHS with N_e combinations of $\mathbf{e}$ accounting for the distribution of $\mathbf{e}$. We use this Kriging metamodel to compute the predictors $\widehat{y}$ of the N outputs. We then derive the N_d conditional means and standard deviations using Eq. (6.45) replacing n_d and n_e by N_d and N_e and replacing the simulation output w by the Kriging predictor $\widehat{y}$. We use these predictions to fit two Kriging metamodels; namely, one Kriging model for the mean output and one for the standard deviation of the output.

Sub 1 and 2: Next we use the two Kriging metamodels—namely, one model for the mean and one model for the standard deviation of the simulation output—as input for the NLP model in Eq. (6.44) to estimate the robust optimal I/O combination. Finally, we vary the threshold T to estimate the Pareto frontier. We call this frontier the "original" frontier, to be distinguished from the bootstrapped frontier (discussed in the next Note).

Note: The original frontier is built on estimates of the mean and standard deviation of the simulation output. To quantify the variability in the estimated mean and standard deviation, we apply *distribution-free bootstrapping*. Moreover, bootstrapping assumes that the original observations are IID; however, the crossed design for $\mathbf{d}$ and $\mathbf{e}$ (see again Table 6.1) implies that the n_d observations on the output for a given combination of the c environmental factors $\mathbf{e}$ are not independent; we might compare this dependence with the dependence created by CRN. Therefore, we sample the n_d-dimensional vectors $\mathbf{w}_j$ $(j = 1, \ldots, n_e)$ n_e times with replacement. This resampling gives the n_e bootstrapped observations $\mathbf{w}_j^*$. This gives the bootstrapped conditional means $\overline{w}_i^*$ and standard deviations s_i^*. To these $\overline{w}_i^*$ and s_i^*, we apply Kriging. These two Kriging metamodels together with the NLP model in Eq. (6.44) give the predicted optimal bootstrapped mean and standard deviation. Repeating this bootstrap sampling (say) B times gives CIs. More research is needed to discover how exactly to use these CIs to account for management's risk attitude; also see Zhang and Ma (2015). Furthermore, Simar and Wilson (1998) studies bootstrapping for estimating the variability of a frontier; namely, the *efficiency frontier* in *data envelop analysis* (DEA), estimated through a LP model. We also refer to Dellino and Meloni (2013) for quantifying the variability of a fitted metamodel, using bootstrapping and cross-validation.

To compare (validate) the robust solution and the classic (nominally optimal) solution, we may sample new combinations of the environmental inputs; i.e., we replace the old LHS combinations by new combinations, because the old combinations favor the robust solution which uses estimates based on these old combinations.

Note: Using a Bayesian approach to the analysis of the I/O data from simulation, Tan (2014a) first fits a Kriging model to the I/O data, then approximates this Kriging model through a so-called *orthonormal polynomial* (which is more complicated than the polynomial models that we discussed in Sect. 2.1), and finally uses this polynomial for "functional analysis of variance" or FANOVA (we discussed FANOVA in Sect. 5.8). This FANOVA can decompose $\sigma^2(w|\mathbf{d})$ (the response variance at a given combination of the decision variables $\mathbf{d}$) into a sum of variances due to the main effects and interactions among the environmental variables $\mathbf{e}$; several sensitivity indexes within the context of robust optimization can be defined. We also refer to Tan (2014b).

Note: EGO (with its EI criterion and Kriging metamodeling, explained in Sect. 6.3.1) may also be used for robust optimization. Actually, Marzat et al. (2013) refers to several publications that extend EGO accounting for a probability distribution of $\mathbf{e}$ such that it minimizes a weighted average of the response w over a discrete set of values for these $\mathbf{e}$. Marzat et al. (2013) combines EGO with algorithms for solving the following *minimax* problem: estimate the combination of $\mathbf{d}$ that minimizes the maximum response when the worst combination of $\mathbf{e}$ occurs; several test functions are investigated. Furthermore, Ur Rehman et al. (2014) extends EGO accounting for implementation errors within an "uncertainty set" (see Sect. 6.4.3 below) and estimating the "best worst-case" or "min-max" solution. Janusevskis and Le Riche (2013) also applies Kriging and EGO for robust optimization.

Note: In Sect. 5.6 on stochastic Kriging (SK) we have already mentioned that the simulation response may be a quantile, which may be relevant in chance-constrained (probabilistically constrained) optimization. Simulation optimization with probabilistic constraints—namely, min $E(w_0)$ such that $P(w_1 \leq c) \geq p$—is discussed in Andrieu et al. (2011) and Sakallı and Baykoç (2011); we also refer back to the references on EGO adapted for chance-constrained optimization in Sect. 6.3.1, and Eq. (6.35) in Sect. 6.3.2 on KrIMP. Stochastically constrained optimization in a R&S context is discussed in Hong et al. (2015). We also refer back to the "expected shortfall", discussed in Sect. 6.4.1 (last Note in that subsection) including references to Broadie et al. (2011) and Sun et al. (2011); those references and also Chen and Kim (2014) and Gan and Lin (2015) use *nested simulation*, which should be distinguished from the crossed designs—as we briefly discusses in the Note after Eq. (6.39). Furthermore, NLP may be replaced by some other optimizer; e.g., an evolutionary algorithm. Finally, we may also apply Dellino et al. (2012)'s methodology to random simulation models, replacing ordinary Kriging (OK) by stochastic Kriging (SK) or stochastic intrinsic kriging (SIK); see the discussions on SK and SIK in Chap. 5. Yin et al. (2015) use simulation of finite element models with uncertain environmental inputs. This simulation is followed by univariate Kriging metamodels. These metamodels are the inputs for a multicriteria optimization problem that combines the means and standard deviations of the multiple simulation outputs. This problem is solved through particle-swarm heuristics.

6.4.3 Ben-Tal et al.'s Robust Optimization

If the mathematical programming (MP) solution ignores the uncertainty in the coefficients of the MP model, then the so-called *nominal solution* may easily violate the constraints in the given model. The *robust solution* may result in a slightly worse value for the goal variable, but it increases the probability of satisfying the constraints; i.e., a robust solution is "immune" to variations of the variables within the *uncertainty set*. Given historical data on the environmental variables $\mathbf{e}$, Yanikoğlu et al. (2015) derives a specific uncertainty set for p where p denotes the unknown density function of $\mathbf{e}$ that is compatible with the historical data on $\mathbf{e}$ (more precisely, p belongs to this set with confidence $1 - \alpha$ if we select some phi-divergence measure such as the well-known chi-square distance). The mathematical challenge in robust optimization of MP models is to develop a computationally tractable so-called *robust counterpart* of the original problem. In this section we do not present the mathematical details of the derivation of tractable robust counterparts, but refer to the references that we gave above.

Note: Taguchians assume a specific distribution for the environmental variables $\mathbf{e}$, which—in case of a multivariate Gaussian distribution—implies a mean vector $\boldsymbol{\mu}_{\mathbf{e}}$ and a covariance matrix $\boldsymbol{\Omega}_{\mathbf{e}}$; see Eqs. (6.38) and (6.39). We may estimate this distribution from historical data. However, Yanikoğlu et al. (2015) develops an approach that uses only the original observed data on $\mathbf{e}$; several numerical examples demonstrate the effectiveness of this novel combination of the two approaches originated by Taguchi and Ben-Tal et al. The uncertainty (or "ambiguity") of the estimated mean vector $\boldsymbol{\mu}_{\mathbf{e}}$ and covariance matrix $\boldsymbol{\Omega}_{\mathbf{e}}$ is also considered in Hu et al. (2012), assuming a multivariate normal distribution for the parameters $\mathbf{e}$ of the underlying simulation model and ambiguity sets for $\boldsymbol{\mu}_{\mathbf{e}}$ and $\boldsymbol{\Omega}_{\mathbf{e}}$ with the corresponding worst-case performance.

The examples in Yanikoğlu et al. (2015) include a deterministic simulation of the television example in Myers et al. (2009, p. 512) and a random simulation of a distribution-center example in Shi (2011); details on the latter example are also given in Shi et al. (2014). The latter example has as response the total throughput, and has five decision variables (e.g., number of forklifts) and two environmental variables (e.g., delay probabilities of suppliers); the incomplete second-order polynomial of Eq. (6.37) is fitted. Yanikoğlu et al. (2015) replaces Eq. (6.36) by the following related problem:

$$\min \sigma_w^2 \quad \text{such that } \mu_w \leq T \tag{6.46}$$

where the statistical parameters μ_w and σ_w^2 are based on the historical data (using the phi-divergence criterion). These two examples demonstrate that robust solutions may have better worst-case performance and also better average performance than the nominal solutions have.

6.5 Conclusions

In this chapter we started with basic RSM, which minimizes the expected value of a single response variable in real-life experiments or deterministic simulation. Next we considered RSM in random simulation. We then presented the ASD search direction, which improves the classic steepest descent direction. We also summarized GRSM for simulation with multivariate responses, assuming that one response is to be minimized while all the other responses and deterministic inputs should satisfy given constraints. Furthermore, we discussed the KKT conditions in constrained minimization, and presented a parametric bootstrap procedure for testing these conditions in random simulation. Next we discussed Kriging for optimization. We detailed EGO for unconstrained optimization in deterministic simulation, and KriMP for constrained optimization in random simulation. Finally, we considered robust optimization, using either the linear regression metamodels of RSM or Kriging metamodels; we also briefly discussed Ben-Tal et al.'s approach to robust optimization.

Future research may study the selection of the required number of replications, and the use of replications to estimate the accuracy of the resulting estimated search direction or optimum. Bootstrapping might solve this problem, but more research is needed. Numerical evaluation of the adapted steepest descent method would benefit from more applications in practice. We also see a need for more research on the KKT testing procedure when all local points (not only the center) are replicated and CRN are used; more practical applications are also needed. Various EGO variants and KriMP need more research. In Taguchian robust optimization we may vary the threshold values, to estimate the Pareto frontier; bootstrapping this frontier might enable management to make the final compromise decision—but more research and applications are needed.

Solutions of Exercises

Solution 6.1 $\mathbf{z}_o = (-5,\ 15)$*; also see Angün et al. (2009).*

Solution 6.2 *If* $\mathbf{Z}'\mathbf{Z} = N\mathbf{I}$*, then Eq. (6.5) implies* $\mathbf{C} = \mathbf{I}/N$*. Hence, Eq. (6.7) does not change the steepest descent direction.*

Solution 6.3 *The ratio of two normal variables has a Cauchy distribution so its expected value does not exist; its median does.*

Solution 6.4 $(z_{o1},\ z_{o2}) = (1.24,\ 0.52)$*; also see Angün et al. (2009).*

References

Ajdari A, Mahlooji H (2014) An adaptive hybrid algorithm for constructing an efficient sequential experimental design in simulation optimization. Commun Stat Simul Comput 43:947–968

Alaeddini A, Yang K, Mao H, Murat A, Ankenman B (2013) An adaptive sequential experimentation methodology for expensive response surface optimization—case study in traumatic brain injury modeling. Qual Reliab Eng Int 30(6): 767–793

Alrabghi A, Tiwari A (2015) State of the art in simulation-based optimisation for maintenance systems. Comput Ind Eng (in press)

Andrieu L, Cohen G, Vázquez-Abad FJ (2011) Gradient-based simulation optimization under probability constraints. Eur J Oper Res 212:345–351

Angün ME (2004) Black box simulation optimization: generalized response surface methodology. CentER dissertation series, Tilburg University, Tilburg, Netherlands (also published by VDM Verlag Dr. Müller, Saarbrücken, Germany, 2011)

Angün E (2011) A risk-averse approach to simulation optimization with multiple responses. Simul Model Pract Theory 19:911–923

Angün E, den Hertog D, Gürkan G, Kleijnen JPC (2009) Response surface methodology with stochastic constraints for expensive simulation. J Oper Res Soc 60(6):735–746

Apley DW, Kim J (2011) A cautious approach to robust design with model parameter uncertainty. IIE Trans 43(7):471–482

Ardakani MK, Wulff SS (2013) An overview of optimization formulations for multiresponse surface problems. Qual Reliab Eng Int 29:3–16

Atlason J, Epelman MA, Henderson SG (2008) Optimizing call center staffing using simulation and analytic center cutting-plane methods. Manag Sci 54(2):295–309

Barnes ER (1986) A variation on Karmarkar's algorithm for solving linear programming problems. Math Program 36:174–182

Barton RR, Meckesheimer M (2006) Metamodel-based simulation optimization. In: Simulation. Handbooks in operations research and management science, vol 13. Elsevier/North Holland, Amsterdam, pp 535–574

Bashyam S, Fu MC (1998) Optimization of (s, S) inventory systems with random lead times and a service level constraint. Manag Sci 44:243–256

Bassamboo A, Randhawa RS, Zeevi A (2010) Capacity sizing under parameter uncertainty: safety staffing principles revisited. Manag Sci 56(10):1668–1686

Bates RA, Kenett RS, Steinberg DM, Wynn HP (2006) Achieving robust design from computer simulations. Qual Technol Quant Manag 3(2):161–177

Ben-Tal A, Nemirovski A (1998) Robust convex optimization. Math Oper Res 23(4):769–805

Ben-Tal A, Nemirovski A (2008) Selected topics in robust convex optimization. Math Program 112(1):125–158

Bettonvil BWM, del Castillo E, Kleijnen JPC (2009) Statistical testing of optimality conditions in multiresponse simulation-based optimization. Eur J Oper Res 199(2):448–458

Beyer H, Sendhoff B (2007) Robust optimization—a comprehensive survey. Comput Methods Appl Mech Eng 196(33–34):3190–3218

Binois M, Ginsbourger D, Roustant O (2015) Quantifying uncertainty on Pareto fronts with Gaussian process conditional simulations. Eur J Oper Res 243: 386–394

Boukouvalas A, Cornford D, Stehlík M (2014) Optimal design for correlated processes with input-dependent noise. Comput Stat Data Anal 71:1088–1102

Box GEP (1999) Statistics as a catalyst to learning by scientific method, part II—a discussion. J Qual Technol 31(1):16–29

Box GEP, Wilson KB (1951) On the experimental attainment of optimum conditions. J R Stat Soc Ser B 13(1):1–38

Brekelmans R, Driessen L, Hamers H, den Hertog D (2005) Gradient estimation schemes for noisy functions. J Optim Theory Appl 126(3): 529–551

Broadie M, Du Y, Moallemi CC (2011) Efficient risk estimation via nested sequential simulation. Manag Sci 57:1172–1194

Bull AD (2011) Convergence rates of efficient global optimization algorithms. J Mach Learn Res 12:2879–2904

Chang K-H, Hong J, Wan H (2013) Stochastic trust-region response-surface method (STRONG)—a new response-surface framework for simulation optimization. INFORMS J Comput 25(2):230–243

Chang K-H, Li M-K, Wan H (2014) Combining STRONG with screening designs for large-scale simulation optimization. IIE Trans 46(4):357–373

Chang K-H, Lin G (2015) Optimal design of hybrid renewable energy systems using simulation optimization. Simul Model Pract Theory 52:40–51

Chau M, Fu MC, Qu H, Ryzhov I (2014) Simulation optimization: a tutorial overview and recent developments in gradient-based and sequential allocation methods. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 21–35

Chen X, Kim K-K (2014) Stochastic kriging with biased sample estimates. ACM Trans Model Comput Simul 24(2):8:1–8:23

Chevalier C, Ginsbourger D, Bect J, Vazquez E, Picheny V, Richet Y (2014) Fast parallel Kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. Technometrics 56(4): 455–465

Chih M (2013) A more accurate second-order polynomial metamodel using a pseudo-random number assignment strategy. J Oper Res Soc 64: 198–207

Conn AR, Gould NLM, Toint PL (2000) Trust-region methods. SIAM, Philadelphia

Davis E, Ierapetritou M (2009) A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. J Glob Optim 43:191–205

Dehlendorff C, Kulahci M, Andersen K (2011) Designing simulation experiments with controllable and uncontrollable factors for applications in health care. J R Stat Soc Ser C (Appl Stat) 60:31–49

Dellino G, Kleijnen JPC, Meloni C (2010) Robust optimization in simulation: Taguchi and response surface methodology. Int J Prod Econ 125(1):52–59

Dellino G, Kleijnen JPC, Meloni C (2012) Robust optimization in simulation: Taguchi and Krige combined. INFORMS J Comput 24(3):471–484

Dellino G, Meloni C (2013) Quantitative methods to analyze simulation metamodels variability. In: Spitaleri RM (ed) Proceedings of the 11th meeting on applied scientific computing and tools. IMACS series in computational and applied mathematics, vol 17, pp 91–100

Dellino G, Meloni C (eds) (2015) Uncertainty management in simulation-optimization of complex systems. Algorithms and applications. Springer, New York

Dengiz B (2009) Redesign of PCB production line with simulation and Taguchi design. In: Rossetti MD, Hill RR, Johansson B, Dunkin A, Ingalls RG (eds) Proceedings of the 2009 winter simulation conference, Austin, pp 2197–2204

Dykstra RL (1970) Establishing the positive definiteness of the sample covariance matrix. Ann Math Stat 41(6):2153–2154

Echard B, Gayton N, Lemaire M (2011) Ak-mcs: an active learning reliability method combining Kriging and Monte Carlo simulation. Struct Saf 33(2):145–154

Fan S-KS, Huang K-N (2011) A new search procedure of steepest ascent in response surface exploration. J Stat Comput Simul 81(6):661–678

Feyzioğlu O, Pierreval H, Deflandre D (2005) A simulation-based optimization approach to size manufacturing systems. Int J Prod Res 43(2): 247–266

Feng Z, Zhang Q, Tang Q, Yang T, Ma Y (2015) A multiobjective optimization based framework to balance the global exploration and local exploitation in expensive optimization. J Glob Optimi, 61(4):677–694

Figueira G, Almada-Lobo B (2014) Hybrid simulation-optimization methods: a taxonomy and discussion. Simul Model Pract Theory 46:118–134

Forrester AIJ, Jones DR (2008) Global optimization of deceptive functions with sparse sampling. In: 12th AIAA/ISSMO multidisciplinary analysis and optimization conference, Victoria, pp 10–12

Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1–3):50–79

Forrester AIJ, Sóbester A, Keane AJ (2008) Engineering design via surrogate modelling; a practical guide. Wiley, Chichester, pp 79–102

Frazier PI (2010) Learning with dynamic programming. In: Cochran JJ, Cox LA, Keskinocak P, Kharoufeh JP, Smith JC (eds) Wiley encyclopedia of operations research and management science. Wiley, New York

Frazier P, Powell W, Dayanik S (2009) The knowledge-gradient policy for correlated normal beliefs. INFORMS J Comput 21:599–613

Fu MC, Bayraksan G, Henderson SG, Nelson BL, Powell WB, Ryzhov IO, Thengvall B (2014) Simulation optimization: a panel on the state of the art in research and practice. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3696–3706

Gabrel V, Murat C, Thiele A (2014) Recent advances in robust optimization: an overview. Eur J Oper Res 235(3):471–483

Gano SE, Renaud JE, Martin JD, Simpson TW (2006) Update strategies for Kriging models for using in variable fidelity optimization. Struct Multidiscip Optim 32(4):287–298

Gan G, Lin XS (2015) Valuation of large variable annuity portfolios under nested simulation: a functional data approach. Insurance: Math Econ 62:138–150

Gill PE, Murray W, Wright MH (2000) Practical optimization, 12th edn. Academic, London

Ginsbourger D, Le Riche R, Carraro L (2010) Kriging is well-suited to parallelize optimization. In: Tenne Y, Goh C-K (eds) Computational intelligence in expensive optimization problems. Springer, Berlin, pp 131–162

Gordy MB, Juneja S (2010) Nested simulation in portfolio risk measurement. Manag Sci 56(11):1833–1848

Gorissen D (2010) Grid-enabled adaptive surrogate modeling for computer aided engineering. Ph. D. dissertation Ghent University, Ghent, Belgium

Gosavi A (2015) Simulation-based optimization: parametric optimization techniques and reinforcement learning, 2nd edn. Springer, Boston

Gramacy RB, Gray GA, Le Digabel S, Lee HKH, Ranjan P, Wells G, Wild SM (2015) Modeling an augmented Lagrangian for blackbox constrained optimization. Technometrics (in press)

Gramacy RB, Lee HKH (2010) Optimization under unknown constraints. Bayesian Stat 9:1–18

Grubler A, Ermoliev Y, and Kryazhimskiy A (2015) Coping with uncertainties examples of modeling approaches at IIASA. Technological Forecasting and Social Change (in press)

Hamarat C, Kwakkel JH, Pruyt E, Loonen ET (2014) An exploratory approach for adaptive policymaking by using multi-objective robust optimization. Simul Model Pract Theory 46:25–39

Homem-de-Mello T, Bayraksan G (2014) Monte Carlo sampling-based methods for stochastic optimization. Surv Oper Res Manag Sci 19(1): 56–85

Hong LJ, Luo J, Nelson BL (2015) Chance constrained selection of the best. INFORMS J Comput 27(2):317–334

Hu Z, Cao J, Hong LJ (2012) Robust simulation of global warming policies using the DICE model. Manag Sci 58(12):2190–2206

Huang D, Allen TT, Notz W, Zheng N (2006) Global optimization of stochastic black-box systems via sequential Kriging meta-models. J Glob Optim 34:441–466

Huang Y, Hsieh C-Y (2014) Influence analysis in response surface methodology. J Stat Plan Inference 147:188–203

Huerta A, Elizondo M (2014) Analysis of scientific collaboration patterns in co-authorship network of simulation-optimization of supply chains. Simul Model Pract Theory 46:135–148

Jala M, Lévy-Leduc C, Moulines É, Conil E, Wiart J (2014) Sequential design of computer experiments for the assessment of fetus exposure to electromagnetic fields. Technometrics (in press)

Jalali H, van Nieuwenhuyse I (2014) Evaluation of Kriging-based methods for simulation optimization with homogeneous noise. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, pp 4057–4058

Jalali H, Van Nieuwenhuyse I (2015, accepted) Simulation optimization in inventory replenishment: a classification. IIE Trans

Janusevskis J, Le Riche R (2013) Simultaneous kriging-based estimation and optimization of mean response. J Glob Optim 55(2):313–336

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Glob Optim 13:455–492

Joshi S, Sherali HD, Tew JD (1998) An enhanced response surface methodology (RSM) algorithm using gradient deflection and second-order search strategies. Comput Oper Res 25(7/8):531–541

Kamiński B (2015) A method for updating of stochastic Kriging metamodels. Eur J Oper Res (accepted)

Kasaie P, Kelton WD (2013) Simulation optimization for allocation of epidemic-control resources. IIE Trans Healthc Syst Eng 3(2):78–93

Kasaie P, Vaghefi A, Naieni G (2009) Optimal resource allocation for control of epidemics: an agent based simulation approach. Working Paper, Dept. of Industrial Engineering, Iran University of Science & Technology, Tehran, 16844, Iran

Kelton WD, Sadowski RP, Sturrock DT (2007) Simulation with Arena, 4th edn. McGraw-Hill, Boston

Kenett R, Steinberg D (2006) New frontiers in design of experiments. Qual Progress 61–65

Khuri AI, Mukhopadhyay S (2010) Response surface methodology. Wiley Interdiscip Rev Comput Stat 2:128–149

Kleijnen JPC (1975) Statistical techniques in simulation, part II. Dekker, New York

Kleijnen JPC (1993) Simulation and optimization in production planning: a case study. Decis Support Syst 9:269–280

Kleijnen JPC (2008) Design and analysis of simulation experiments. Springer, New York

Kleijnen JPC (2014) Response surface methodology. In: Fu MC (ed) Handbook of simulation optimization. Springer, New York

Kleijnen JPC, Den Hertog D, Angün E (2004) Response surface methodology's steepest ascent and step size revisited. Eur J Oper Res 159:121–131

Kleijnen JPC, Den Hertog D, Angün E (2006) Response surface methodology's steepest ascent and step size revisited: correction. Eur J Oper Res 170:664–666

Kleijnen JPC, Gaury EGA (2003) Short-term robustness of production-management systems: a case study. Eur J Oper Res 148(2):452–465

Kleijnen JPC, Mehdad E (2013) Conditional simulation for efficient global optimization. In: Proceedings of the 2013 winter simulation conference, Washington, pp 969–979

Kleijnen JPC, Pierreval H, Zhang J (2011) Methodology for determining the acceptability of system designs in uncertain environments. Eur J Oper Res 209(2):176–183

Kleijnen JPC, Sargent RG (2000) A methodology for the fitting and validation of metamodels in simulation. Eur J Oper Res 120(1):14–29

Kleijnen JPC, Van Beers WCM, van Nieuwenhuyse I (2010) Constrained optimization in simulation: a novel approach. Eur J Oper Res 202: 164–174

Kleijnen JPC, Van Beers W, Van Nieuwenhuyse I (2012) Expected improvement in efficient global optimization through bootstrapped Kriging. J Glob Optim 54:59–73

Kleijnen JPC, Wan J (2007) Optimization of simulated systems: OptQuest and alternatives. Simul Model Pract Theory 15:354–362

Koch P, Wagner T, Emmerich MTM, Bäck T, Konen W (2015) Efficient multi-criteria optimization on noisy machine learning problems. Appl Soft Comput (in press)

Kolaiti E, Koukouvinos C (2006) On the use of three level orthogonal arrays in robust parameter design. Stat Probab Lett 76(3):266–273

Koziel S, Bekasiewicz A, Couckuyt I, Dhaene T (2014) Efficient multi-objective simulation-driven antenna design using co-Kriging. IEEE Trans Antennas Propag 62(11):5901–5915

Lan H, Nelson BL, Staum J (2010) A confidence interval procedure for expected shortfall risk measurement via two-level simulation. Oper Res 58(5):1481–1490

Law AM (2015) Simulation modeling and analysis, 5th edn. McGraw-Hill, Boston

Lee KH, Park GJ (2006) A global robust optimization using Kriging based approximation model. J Jpn Soc Mech Eng 49:779–788

Lee LH, Chew EP, Frazier PI, Jia Q-S, Chen C-H (2013) Foreword: advances in simulation optimization and its applications. IIE Trans 45(7):683–684

Lee S, Nelson BL (2014) Bootstrap ranking & selection revisited. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3857–3868

Leijen MCF (2011) Response surface methodology for simulation optimization of a packaging line. Master's thesis, Eindhoven University of Technology, Department of Mechanical Engineering, Systems Engineering Group, Eindhoven

Mahdavi I, Shirazi B, Solimanpur M (2010) Development of a simulation-based decision support system for controlling stochastic flexible job shop manufacturing systems. Simul Model Pract Theory 18:768–786

Marzat J, Walter E, Piet-Lahanie H (2013) Worst-case global optimization of black-box functions through Kriging and relaxation. J Glob Optim 55:707–727

Mehdad E, Kleijnen JPC (2015) Classic Kriging versus Kriging with bootstrapping or conditional simulation: classic Kriging's robust confidence intervals and optimization. J Oper Res Soc (in press)

Mehdad E, Kleijnen JPC (2014) Global optimization for black-box simulation through sequential intrinsic Kriging. CentER Discussion Paper 2014-063, Tilburg University, Tilburg, Netherlands

Meloni C, Dellino G (eds) (2015) Uncertainty management in simulation-optimization of complex systems; algorithms and applications. Springer

Miller GA (1956) The magical number seven plus or minus two: some limits on our capacity for processing information. Psychol Rev 63:81–97

Montevechi JAB, de Almeida Filho RG, Paiva AP, Costa RFS, and A.L. Medeiros (2010) Sensitivity analysis in discrete-event simulation using fractional factorial designs. J Simul 4(2):128–142

Morales-Enciso S and Branke J (2015) Tracking global optima in dynamic environments with efficient global optimization. Eur J Oper Res 242(3):744–755

Müller J, Shoemaker CA (2014) Influence of ensemble surrogate models and sampling strategy on the solution quality of algorithms for computationally expensive black-box global optimization problems. J Glob Optim 60(2):123–144

Myers RH, Khuri AI, Carter WH (1989) Response surface methodology: 1966–1988. Technometrics 31(2):137–157

Myers RH, Montgomery DC, Anderson-Cook CM (2009) Response surface methodology: process and product optimization using designed experiments, 3rd edn. Wiley, New York

Nair VN (ed) (1992) Taguchi's parameter design: a panel discussion. Technometrics 34(2):127–161

Nakayama H, Yun Y, Yoon M (2009) Sequential approximate multiobjective optimization using computational intelligence. Springer, Berlin, pp 133–141

Ng SH, Xu K, Wong WK (2007) Optimization of multiple response surfaces with secondary constraints for improving a radiography inspection process. Qual Eng 19(1):53–65

Oden JT (2006) Revolutionizing engineering science through simulation. National Science Foundation (NSF), Blue Ribbon Panel on Simulation-Based Engineering Science

Park G-J, Lee T-H, Lee KH, Hwang K-H (2006) Robust design: an overview. AIAA J 44(1):181–191

Pasupathy R, Ghosh S (2014) Simulation optimization: a concise overview and implementation guide. INFORMS Tutorials in Operations Research, pp 122–150. http://pubsonline.informs.org/doi/book/10.1287/educ.2014#Chapters

Picheny V, Ginsbourger D, Richet Y, Caplin G (2013a) Quantile-based optimization of noisy computer experiments with tunable precision (including comments and rejoinder). Technometrics 55(1):1–36

Picheny V, Wagner T, Ginsbourger D (2013b) A benchmark of kriging-based infill criteria for noisy optimization. Struct Multidiscip Optim 48:607–626

Preuss M, Wagner T, Ginsbourger D (2012) High-dimensional model-based optimization based on noisy evaluations of computer games. In: Hamadi Y, Schoenauer M (eds) Learning and intelligent optimization: 6th international conference (LION 6), Paris. Springer, Berlin, pp 145–159

Quan N, Yin J, Ng SH, Lee LH (2013), Simulation optimization via kriging: a sequential search using expected improvement with computing budget constraints. IIE Trans 45:763–780

Qu H, Ryzhov IO, Fu MC, Ding Z (2015) Sequential selection with unknown correlation structures. Oper Res 63(4):931–948

Rashid K, Bailey, WJ Couet B, and Wilkinson D (2013) An efficient procedure for expensive reservoir-simulation optimization under uncertainty. SPE Econ Manage 5(4):21–33

Razavi S, Tolson BA, Burn DH (2012) Review of surrogate modeling in water resources. Water Resour Res 48, W07401:1–322

Regis RG (2014) Locally-biased efficient global optimization using Kriging metamodels Working paper, Department of Mathematics, Saint Joseph's University, Philadelphia

Rikards R, Auzins J (2002) Response surface method for solution of structural identification problems. In: Fourth international conference on inverse problems in engineering, Rio de Janeiro

Rosen SC, Harmonosky CM, Traband MT (2008) Optimization of systems with multiple performance measures via simulation: survey and recommendations. Comput Ind Eng 54(2):327–339

Roustant O, Ginsbourger D, Deville Y (2012) DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by Kriging-based metamodeling and optimization. J Stat Softw 51(1):1–55

Safizadeh MH (2002) Minimizing the bias and variance of the gradient estimate in RSM simulation studies. Eur J Oper Res 136(1):121–135

SakallıÜS, Baykoç ÖF (2011) An optimization approach for brass casting blending problem under aletory and epistemic uncertainties. Int J Prod Econ 133(2):708–718

Salemi P, Nelson BL, Staum J (2014) Discrete optimization via simulation using Gaussian Markov random fields. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3809–3820

Saltelli A, Ratto M, Tarantola S, Campolongo F (2005) Sensitivity analysis of chemical models. Chem Rev 105(7):2811–2827

Samuelson D (2010) When close is better than optimal: combining simulation and stochastic optimization for better risk management. OR/MS Today 37(6):38–41

Santos MI, Santos PM (2011) Construction and validation of distribution-based regression simulation metamodels. J Oper Res Soc 62:1376–1384

Sasena MJ, Papalambros P, Goovaerts P (2002) Exploration of metamodeling sampling criteria for constrained global optimization. Eng Optim 34(3):263–278

Scott W, Frazier P, Powell W (2011) The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression. SIAM J Optim 21(3):996–1026

Scott WR, Powell WB, Simao HP (2010) Calibrating simulation models using the knowledge gradient with continuous parameters. In: Proceedings of the 2010 winter simulation conference, Baltimore, pp 1099–1109

Shang JS, Li S, Tadikamalla P (2004) Operational design of a supply chain system using the Taguchi method, response surface methodology, simulation, and optimization. Int J Prod Res 42(18):3823–3849

Shahraki AF, Noorossana R (2014) Reliability-based robust design optimization: a general methodology using genetic algorithm. Comput Ind Eng 74:199–207

Shi W (2011) Design of pre-enhanced cross-docking distribution center under supply uncertainty: RSM robust optimization method. Working Paper, Huazhong University of Science & Technology, China

Shi W, Shang J, Liu Z, Zuo X (2014) Optimal design of the auto parts supply chain for JIT operations: sequential bifurcation factor screening and multi-response surface methodology. Eur J Oper Res 236(2):664–676

Shin S, Samanlioglu F, Cho BR, Wiecek MM (2011) Computing trade-offs in robust design: perspectives of the mean squared error. Comput Ind Eng 60(2):248–255

Simar L, Wilson PW (1998) Sensitivity analysis of efficiency scores: how to bootstrap in nonparametric frontier models. Manag Sci 44(1):49–61

Simon HA (1956) Rational choice and the structure of the environment. Psychol Rev 63(2):129–138

Simpson TW, Booker AJ, Ghosh D, Giunta AA, Koch PN, Yang R-J (2004) Approximation methods in multidisciplinary analysis and optimization: a panel discussion. Struct Multidiscip Optim 27(5):302–313

Stinstra E, den Hertog D (2008) Robust optimization using computer experiments. Eur J Oper Res 191(3):816–837

Sun L, Hong LJ, Hu Z (2014) Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. Oper Res 62(6):1416–1438

Sun Y, Apley DW, Staum J (2011) Efficient nested simulation for estimating the variance of a conditional expectation. Oper Res 59(4):998–1007

Svenson JD, Santner TJ (2010) Multiobjective optimization of expensive black-box functions via expected maximin improvement. The Ohio State University, Columbus, Ohio

Taddy MA, Lee HKH, Gray GA, Griffin JD (2009) Bayesian guided pattern search for robust local optimization. Technometrics 5(4):389–401

Taguchi G (1987) System of experimental designs, vols 1 and 2. UNIPUB/ Krauss International, White Plains

Tajbakhsh S, del Castillo E, Rosenberger JL (2013) A fully Bayesian approach to the efficient global optimization algorithm. Working Paper, Pennsylvania State University

Tan MHY (2014a) Robust parameter design with computer experiments using orthonormal polynomials. Technometrics (in press)

Tan MHY (2014b) Stochastic polynomial interpolation for uncertainty quantification with computer experiments. Technometrics (in press)

Tenne Y, Goh C-K (eds) (2010) Computational intelligence in expensive optimization problems. Springer, Berlin

Tong C, Sun Z, Zhao Q, Wang Q, Wang S (2015) A hybrid algorithm for reliability analysis combining Kriging and subset simulation importance sampling. J Mech Sci Technol 29(8):3183–3193

Ur Rehman S, Langelaar M, van Keulen F (2014) Efficient Kriging-based robust optimization of unconstrained problems. J Comput Sci (in press)

Van den Bogaard W, Kleijnen JPC (1977) Minimizing waiting times using priority classes: a case study in response surface methodology. Discussion Paper FEW 77.056. http://arno.uvt.nl/show.cgi?fid=105001. Accessed 12 Mar 2014

Van der Herten J, Couckuyt I, Deschrijver D, Dhaene T (2015) A fuzzy hybrid sequential design strategy for global surrogate modeling of high-dimensional computer experiments. SIAM J Sci Comput 37(2):A1020–A1039

Vazquez E, Bect J (2010) Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. J Stat Plan Inference 140(11):3088–3095

Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogate techniques. J Glob Optim 56(2):669–689

Villemonteix J, Vazquez E, Sidorkiewicz M, Walter E (2009a) Global optimization of expensive-to-evaluate functions: an empirical comparison of two sampling criteria. J Glob Optim 43:373–389

Villemonteix J, Vazquez E, Walter E (2009b) An informational approach to the global optimization of expensive-to-evaluate functions. J Glob Optim 44(4):509–534

Wagner T (2013) Planning and multi-objective optimization of manufacturing processes by means of empirical surrogate models. Doctoral dissertation, Technische Universität Dortmund, Dortmund, Germany

Wan J, Li L (2008) Simulation for constrained optimization of inventory system by using Arena and OptQuest. In: 2008 international conference on computer science and software engineering (CSSE 2008). IEEE, Wakefield, MA, pp 202–205

Wiebenga JH (2014) Robust design and optimization of forming processes. Ph.D. thesis, University of Twente, Enschede, Netherlands

Wiesemann W, Kuhn D, Sim M (2014) Distributionally robust convex optimization. Oper Res (in press)

Williams BJ, Santner TJ, Notz WI, Lehman JS (2010) Sequential design of computer experiments for constrained optimization. In: Kneib T, Tutz G (eds) Festschrift for Ludwig Fahrmeir, Springer, Berlin, pp 449–471

Wu CFJ, Hamada M (2009) Experiments; planning, analysis, and parameter design optimization, 2nd edn. Wiley, New York

Xu J, Huang E, Chen C-H, Lee LH (2015) Simulation optimization: a review and exploration in the new era of cloud computing and big data. Asia Pacific J Oper Res (in press)

Yalçinkaya, Ö, Bayhan GM (2009) Modelling and optimization of average travel time for a metro line by simulation and response surface methodology. Eur J Oper Res 196(1):225–233

Yanikoğlu İ, den Hertog D, and Kleijnen JPC (2015), Robust dual response optimization. IIE Trans (in press)

Yarotsky D (2013) Examples of inconsistency in optimization by expected improvement. J Glob Optim 56, pp. 1773–1790

Yin H, Fang H, Xiao Y, Wen G, Qing Q (2015) Multi-objective robust optimization of foam-filled tapered multi-cell thin-walled structures. Struct Multidiscip Optim (in press)

Ye W, You F (2015) A fast simulation-based optimization method for inventory control of general supply chain networks under uncertainty American control conference, Palmer House Hilton, Chicago, 1–3 July 2015, pp 2001–2006

Zazanis MA, Suri R (1993) Convergence rates of finite-difference sensitivity estimates for stochastic systems. Oper Res 41(4):694–703

Zhang J, Ma Y (2015) Stochastic Kriging-assisted multi-objective simulation optimization and uncertainty analysis. Simul: Trans Soc Model Simul Int (in press)

Zhou E, Bhatnagar S, Chen X (2014) Simulation optimization via gradient-based stochastic search. In: Tolk A, Diallo SY, Ryzhov IO, Yilmaz L, Buckley S, Miller JA (eds) Proceedings of the 2014 winter simulation conference, Savannah, pp 3869–3879

Author Index

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8

Subject Index

J.P.C. Kleijnen, *Design and Analysis of Simulation Experiments*,
International Series in Operations Research & Management
Science 230, DOI 10.1007/978-3-319-18087-8