Nader H. Bshouty
Gilles Stoltz
Nicolas Vayatis
Thomas Zeugmann (Eds.)

LNAI 7568

Algorithmic Learning Theory

23rd International Conference, ALT 2012
Lyon, France, October 2012
Proceedings

Springer

Lecture Notes in Artificial Intelligence 7568

Subseries of Lecture Notes in Computer Science

Nader H. Bshouty   Gilles Stoltz
Nicolas Vayatis   Thomas Zeugmann (Eds.)

Algorithmic Learning Theory

23rd International Conference, ALT 2012
Lyon, France, October 29-31, 2012
Proceedings

Springer

Volume Editors

Nader H. Bshouty
Technion, Haifa, Israel
E-mail: bshouty@cs.technion.ac.il

Gilles Stoltz
Ecole Normale Supérieure, CNRS, INRIA, Paris, France
E-mail: gilles.stoltz@ens.fr

Nicolas Vayatis
Ecole Normale Supérieure de Cachan, France
E-mail: vayatis@cmla.ens-cachan.fr

Thomas Zeugmann
Hokkaido University, Sapporo, Japan
E-mail: thomas@ist.hokudai.ac.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-34105-2 e-ISBN 978-3-642-34106-9
DOI 10.1007/978-3-642-34106-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012949059

CR Subject Classification (1998): I.2, F.4.1, F.1, F.2, I.2.3, I.2.6

LNCS Sublibrary: SL 7 – Artificial Intelligence

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

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

Preface

This volume contains the papers presented at the 23rd International Conference on Algorithmic Learning Theory (ALT 2012), which was held in Lyon, France, October 29–31, 2012. The conference was co-located and held in parallel with the 15th International Conference on Discovery Science (DS 2012). The technical program of ALT 2012 contained 23 papers selected from 47 submissions, and five invited talks. The invited talks were presented in joint sessions of both conferences.

ALT 2012 was dedicated to the theoretical foundations of machine learning and took place in the historical building of the Université Lumière Lyon 2 (berges du Rhônes). ALT provides a forum for high-quality talks with a strong theoretical background and scientific interchange in areas such as inductive inference, universal prediction, teaching models, grammatical inference, complexity of learning, online learning, semi-supervised and unsupervised learning, clustering, statistical learning, regression, bandit problems, Vapnik–Chervonenkis dimension, probably approximately correct learning, information-based methods, and applications of algorithmic learning theory.

The present volume contains the texts of the 23 papers presented at ALT 2012, divided into groups of papers on inductive inference, teaching and PAC–learning, statistical learning theory and classification, relations between models and data, bandit problems, online learning of individual sequences, and on other models of online learning. The volume also contains the texts or abstracts of the invited talks:

- Luc De Raedt (Katholieke Universiteit Leuven, Belgium), "Declarative Modeling for Machine Learning and Data Mining" (joint invited speaker for ALT 2012 and DS 2012)
- Shai Shalev-Shwartz (The Hebrew University of Jerusalem, Israel), "Learnability Beyond Uniform Convergence" (invited speaker for ALT 2012)
- Pascal Massart (Université Paris-Sud, France), "Some Rates of Convergence for the Selected Lasso Estimator" (tutorial speaker for ALT 2012)
- Toon Calders (Eindhoven University of Technology, The Netherlands), "Recent Developments in Pattern Mining" (invited speaker for DS 2012)
- Gilbert Ritschard (Université de Genève, Switzerland), "Exploring Sequential Data" (tutorial speaker for DS 2012).

Since 1999, ALT has been awarding the *E. M. Gold Award* for the most outstanding student contribution. This year, the award was given to Ziyuan Gao for his paper "Confident and Consistent Partial Learning of Recursive Functions" co-authored by Frank Stephan.

ALT 2012 was the 23rd in the ALT conference series, established in Japan in 1990. The ALT series is supervised by its Steering Committee: Naoki Abe (IBM Thomas J. Watson Research Center, Yorktown, USA), Shai Ben-David

(University of Waterloo, Canada), Nader Bshouty (Technion - Israel Institute of Technology, Israel) Marcus Hutter (Australian National University, Canberra, Australia), Jyrki Kivinen (University of Helsinki, Finland), Philip M. Long (Google, Mountain View, USA), Akira Maruoka (Ishinomaki Senshu University, Japan), Takeshi Shinohara (Kyushu Institute of Technology, Iizuka, Japan), Frank Stephan (National University of Singapore, Republic of Singapore), Gilles Stoltz (Ecole Normale Supérieure, France), Einoshin Suzuki (Kyushu University, Fukuoka, Japan), Csaba Szepesvári (University of Alberta, Canada), Eiji Takimoto (Kyushu University, Fukuoka, Japan), György Turán (University of Illinois at Chicago, USA and University of Szeged, Hungary), Osamu Watanabe (Tokyo Institute of Technology, Japan), Thomas Zeugmann (Chair, Hokkaido University, Japan), and Sandra Zilles (Publicity Chair, University of Regina, Saskatchewan, Canada).

We would like to thank the many people and institutions who contributed to the success of the conference. In particular, we want to thank our authors for contributing to the conference and for coming to Lyon in October 2012. Without their efforts and their willingness to choose ALT 2012 as a forum to report on their research, this conference would not have been possible.

ALT 2012 and DS 2012 were organized by the Université Lumière Lyon 2, France. We are very grateful to the General Chair Djamel Abdelkader Zighed and the General Local Arrangements Chair Stéphane Lallich. We would like to thank them and their team for the tremendous amount of work they have dedicated to making ALT 2012 and DS 2012 a success. We are grateful for the continuous collaboration with the series Discovery Science. In particular, we would like to thank the Conference Chair Jean-Gabriel Ganascia and the Program Committee Chairs Philippe Lenca and Jean-Marc Petit of Discovery Science 2012.

We are also grateful that we could use the excellent conference management system EasyChair for putting together the program for ALT 2011; EasyChair was developed mainly by Andrei Voronkov and is hosted at the University of Manchester. The system is cost-free.

We are grateful to the members of the Program Committee for ALT 2012 and the subreferees for their hard work in selecting a good program for ALT 2012. Reviewing papers and checking the correctness of results is demanding in time and skills and we very much appreciate this contribution to the conference. Last but not least we thank Springer for their support in preparing and publishing this volume in the *Lecture Notes in Artificial Intelligence* series.

August 2012

Nader H. Bshouty
Gilles Stoltz
Nicolas Vayatis
Thomas Zeugmann

Organization

General Chair (ALT 2012 and DS 2012)

Djamel Zighed	Université Lumière Lyon 2, France

General Local Chair (ALT 2012 and DS 2012)

Stéphane Lallich	Université Lumière Lyon 2, France

ALT 2012 Conference Chair

Nicolas Vayatis	Ecole Normale Supérieure de Cachan, France

Program Committee

Jake Abernethy	University of Pennsylvania, USA
András Antos	MTA SZTAKI, Hungary
Shai Ben-David	University of Waterloo, Canada
Avrim Blum	Carnegie Mellon University, USA
Nader H. Bshouty (Chair)	Technion–Israel Institute of Technology, Israel
Koby Crammer	Technion–Israel Institute of Technology, Israel
Sanjoy Dasgupta	UC San Diego, USA
Vitaly Feldman	IBM Research, USA
Claudio Gentile	Università dell'Insubria, Varese, Italy
Timo Kötzing	Max-Planck-Institut für Informatik, Germany
Wouter M. Koolen	Royal Holloway, University of London, UK
Phil M. Long	Google, Mountain View, USA
Claire Monteleoni	George Washington University, USA
Lev Reyzin	Georgia Institute of Technology, USA
Steven de Rooij	CWI Amsterdam, The Netherlands
Daniil Ryabko	INRIA, France
Rocco Servedio	Columbia University, USA
Hans Ulrich Simon	Ruhr-Universität Bochum, Germany
Frank Stephan	National University of Singapore
Gilles Stoltz (Chair)	Ecole Normale Supérieure, Paris, France
Csaba Szepesvári	University of Alberta, Edmonton, Canada
Eiji Takimoto	Kyushu University, Japan
Ambuj Tewari	University of Texas at Austin, USA
Vladimir Vovk	Royal Holloway, University of London, UK
Bob Williamson	Australian National University, Australia
Sandra Zilles	University of Regina, Canada

ALT 2012 Local Arrangements Chair

Fabrice Muhlenbach	Université Jean Monnet de Saint-Etienne, France

Subreferees

Mohammad Gheshlaghi Azar
Peter Antal
Frank J. Balbach
Sébastien Bubeck
John Case
Nicolò Cesa-Bianchi
Karthekeyan Chandrasekaran
Archie Chapman
François Coste
Sanmay Das
John Duchi
Rafael M. Frongillo
András György
Zaïd Harchaoui
Kohei Hatano
David P. Helmbold
Daniel Hsu
Benjamin Jourdain
Satyen Kale
Yuri Kalnishkan
Nikos Karampatziakis
Azadeh Khaleghi
Roni Khardon
Aryeh Kontorovich
Nathaniel Korda
Luigi Malagò
Scott McQuade
Samuel E. Moelius III
Edward Moroshko
Francesco Orabona
Ronald Ortner
Vikas Sindhwani
István Szita
Matus Telgarsky
Nina Vaitz
Vladimir Vovk
Andre Wibisono
Thomas Zeugmann

Sponsoring Institutions

eric
UNIVERSITÉ DE LYON
UNIVERSITÉ LUMIÈRE LYON 2 UNIVERSITÉ DE LYON
TELECOM Bretagne Institut Mines-Télécom
INSA LYON

ENS CACHAN
UCB Lyon 1
Université Jean Monnet SAINT-ETIENNE
LABORATOIRE Hubert CURIEN UMR • CNRS • 5516 • ST-ETIENNE
Lab-STICC

PASCAL Pattern Analysis, Statistical Modelling and Computational Learning
arc COMMUNAUTÉS DE RECHERCHE ACADÉMIQUE RhôneAlpes
T.I.C. ET USAGES INFORMATIQUES INNOVANTS
EGC

VILLE DE LYON
Région RhôneAlpes
RHÔNE LE DÉPARTEMENT

Table of Contents

Invited Papers

Regular Contributions

Inductive Inference

Teaching and PAC-Learning

Statistical Learning Theory and Classification

Relations between Models and Data

Bandit Problems

Online Prediction of Individual Sequences

Other Models of Online Learning

Editors' Introduction

Nader H. Bshouty, Gilles Stoltz, Nicolas Vayatis, and Thomas Zeugmann

The ALT-conference series is dedicated to studies on learning from an algorithmic and mathematical perspective. In the following, the five invited lectures and the regular contributions are introduced in some more detail.

Invited Talks. It is now a tradition of the co-located conferences ALT and DS to have a joint invited speaker—namely this year, Luc De Raedt. Since 2006 he is a full research professor at the Department of Computer Science of the Katholieke Universiteit Leuven (Belgium). His research interests are in artificial intelligence, machine learning and data mining, as well as their applications. He is currently working on probabilistic logic learning (sometimes called statistical relational learning), which combines probabilistic reasoning methods with logical representations and machine learning, the integration of constraint programming with data mining and machine learning principles, the development of programming languages for machine learning, and analyzing graph and network data. In his talk *Declarative Modeling for Machine Learning and Data Mining* he notes that despite the popularity of machine learning and data mining today, it remains challenging to develop applications and software that incorporates machine learning or data mining techniques. This is because machine learning and data mining have focused on developing high-performance algorithms for solving particular tasks rather than on developing general principles and techniques. He thus proposes to alleviate these problems by applying the constraint programming methodology to machine learning and data mining and to specify machine learning and data mining problems as constraint satisfaction and optimization problems. The aim is that the user be provided with a way to declaratively specify what the machine learning or data mining problem is rather than having to outline how that solution needs to be computed.

Four other invited talks are also given by eminent researchers in their fields, who present either an introduction to their specific research area or give a lecture of wide general interest.

Shai Shalev-Shwartz is the ALT invited speaker; since 2009 he is a senior lecturer at the School of Computer Science and Engineering of The Hebrew university (Jerusalem, Israel). His research interests include machine learning and learning theory at broad, with an emphasis on online algorithms, large-scale learning, information retrieval, and optimization. In his talk *Learnability Beyond Uniform Convergence* he discusses the problem of characterizing learnability, which in his view is the most basic question of statistical learning theory. He indicates that a fundamental result is that learnability is equivalent to uniform convergence of the empirical risk to the population risk, and that if a problem is learnable, it is learnable via empirical risk minimization. However, the equivalence of uniform convergence and learnability was formally established only in

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 1–11, 2012.

the supervised classification and regression setting. He then shows that in (even slightly) more complex prediction problems learnability does not imply uniform convergence. He thus presents several alternative attempts to characterize learnability. The results obtained are based on joint researches with Ohad Shamir, Nati Srebro, Karthik Sridharan, and with Amit Daniely, Sivan Sabato, and Shai Ben-David, respectively.

Pascal Massart is the ALT tutorial speaker; since 1990 he is a full professor at the Department of Mathematics of the Université Paris-Sud (Orsay, France). He dedicated most of his work in the past 20 years to elaborate a non-asymptotic theory for model selection and made contributions also to related fields, like the theory of empirical processes, concentration-of-the-measure inequalities, and non-parametric statistics. He also established connections between model selection theory and statistical learning theory. His tutorial is based on the paper *Some Rates of Convergence for the Selected Lasso Estimator* co-authored with Caroline Meynet. He illustrates on the example of the Lasso estimator how the theory of model selection in statistics can shed some light and improve some results in learning. More precisely he considers the estimation of a function in some ordered finite or infinite dictionary, that is, in some (non necessarily orthonormal) family of elements in a Hilbert space. He focuses on a variant of the Lasso, the selected Lasso estimator, which he introduced in an earlier paper with Caroline Meynet. This estimator is an adaptation of the Lasso suited to infinite dictionaries. He uses the oracle inequality established therein to derive rates of convergence of this estimator on a wide range of function classes (Besov-type spaces). The results highlight that the selected Lasso estimator is adaptive to the smoothness of the function to be estimated, contrary to the classical Lasso or to other algorithms considered in the literature.

Toon Calders, the invited speaker for DS, received his PhD in Mathematics in 2003 from the University of Antwerp. Since 2006 he is assistant professor in the Information Systems Group at the Department of Mathematics and Computer Science of the Eindhoven University of Technology. His lecture *Recent Developments in Pattern Mining* gives an overview of the many techniques developed to solve pattern mining problems. Many methods have been proposed to enumerate all frequent itemsets. The basic difficulty is the pattern explosion problem, i.e., millions of patterns may be generated. Though this problem is widely recognized, it still lacks a satisfactory solution. Toon Calders surveys promising methods based upon the minimal description length principle, information theory, and statistical models, and discusses the advantages and disadvantages of these new methods. The final part of his lecture addresses more complex patterns such as sequences and graphs, and concludes with important open problems in this challenging area.

Gilbert Ritschard, the DS tutorial speaker, graded in econometrics and got his PhD in Econometrics and Statistics at the University of Geneva in 1979. He also taught as invited professor in Toronto, Montreal, Lyon, Lausanne and Fribourg and participated as a statistical expert in several large statistical modeling projects of International Organizations. He is a full professor of statistics at the

Department of Economics of the University of Geneva, where he is responsible for the program of statistics and quantitative methods for the social sciences and runs his researches within the Institute for Demographic and Life Course Studies and acts as vice-dean of the Faculty of Economics and Social Sciences since 2007. Several funded applied researches were headed or co-headed by him. He also published papers in economics as well as on more applied topics in the field of social sciences, especially in demography, sociology and social science history. With his team he developed the world wide used TraMineR toolbox for exploring and analyzing sequence data in R. His present research interests are in categorical and numerical longitudinal data analysis and their application to life course analysis. His tutorial *Exploring Sequential Data* gives an introduction to sequence analysis as it is practiced for life course analysis. Examples comprise successive buys of customers, working states of devices, visited web pages, or professional careers, and addressed topics are the rendering of state and event sequences, longitudinal characteristics of sequences, measuring pairwise dissimilarities and dissimilarity-based analysis of sequence data such as clustering, representative sequences, and regression trees. All the methods employed are available in TraMineR R-package.

We now turn our attention to the regular contributions contained in this volume.

Inductive Inference. One of the classical areas of algorithmic learning is inductive inference of recursive functions. In this setting the learner is usually fed augmenting finite sequences $f(0), f(1), f(2), \ldots$ of the target function f. For each finite sequence the learner has to compute a hypothesis, i.e., a natural number. These numbers are interpreted with respect to a given enumeration of partial recursive functions comprising the target function. Then the number j output by the learner is interpreted as a program computing the jth function enumerated. The sequence of all hypotheses output by the learner has then to converge (to stabilize) on a program that, under the given correctness criterion, correctly computes the target function. This learning scenario is commonly called *explanatory inference* or *learning in the limit*. Since only finitely many values of the function have been seen by the learner up to the unknown point of convergence, some form of learning must have taken place. Usually, the goal is then to construct a learner that can infer all functions from a given target class U. Many variations of this model are possible. For finite learning one requires the point of convergence to be decidable. Another variation is to allow the learner to converge semantically, i.e., instead of stabilizing to a correct program, the learner is allowed to output infinitely many *different programs* which, however, beyond some point, all must correctly compute the target function. This model is commonly referred to as *behaviorally correct learning*. In the context of the first paper introduced below also the notions of *confidence* and *reliability* are of particular interest. A confident learner is required to converge on every function, even it is not in the target class (but may stabilize to a special symbol "?"). In contrast, a reliable learner must signal its inability to learn a target function (which may be again outside the class) by performing infinitely many mind changes. Thus, if a reliable learner converges then it learns. In this context it remains to specify what is meant by “outside” the target class. Above all total functions (including the non-computable ones) are considered. If one allows only the total recursive functions then the resulting models are called *weakly confident* and *weakly reliable*, respectively.

The problems studied by Sanjay Jain, Timo Kötzing, and Frank Stephan in their paper *Enlarging Learnable Classes* are easily described as follows. Suppose we have already a learner for a target class U_1 and another one for a class U_2. Then it is only natural to ask under which circumstances one can obtain a more powerful learner that simultaneously infers all functions from $U_1 \cup U_2$. A classical result shows that this is not always possible, even for behaviorally correct learning. If one can obtain such a more powerful learner then it is also interesting to ask whether or not it can be done effectively. That is, given programs for learners M_1 and M_2 inferring U_1 and U_2, respectively, one studies the problem whether or not one can compute from these programs a learner M for the union $U_1 \cup U_2$. Still, it is imaginable that one cannot compute such a learner but show it to exist (this is the non-effective case). The interesting new modification of this problem introduced by the authors is to ask which classes U_1 have the property that $U_1 \cup U_2$ is learnable for *all* classes U_2. As shown by Jain *et al.*, if U_1 has a weakly confident and reliable learner then the union $U_1 \cup U_2$ is always explanatory learnable and the learner is effective. Moreover, they show the effective case and the non-effective case separate and a sufficient criterion is shown for the effective case. A closely related problem is to ask the same questions when the second class is restricted to be any singleton class. In this case it suffices that the learner for U_1 is weakly confident to obtain the effective case. In contrast, for finite learning there is no non-empty class U_1 satisfying the non-constructive case for classes and the constructive case for singleton classes. Furthermore, the authors investigate the problem how much information is needed to enlarge a learnable class by infinitely many functions while maintaining its learnability. In this context, two questions that remained open in a paper by Mark Fulk and John Case in 1999 are completely answered.

The next paper in this section is the Gold Award winning paper *Confident and Consistent Partial Learning of Recursive Functions* by Ziyuan Gao and Frank Stephan for the best paper co-authored by a student, who is Ziyuan Gao. As the discussion above shows there is no single learner that can infer the whole class of recursive functions. Therefore, it is interesting to consider further variations. Osherson, Stob, and Weinstein (1986) considered partial learning, where the learner is required to output a correct program for the target function infinitely often and any other hypothesis only finitely often. Gao and Stephan refine this model by combining it with the confidence demand discussed above and with consistent learning. A consistent learner has to correctly reflect the information already obtained, and this demand is posed to all but finitely many of the hypotheses output. The resulting models are called confident partial learning and consistent partial learning, respectively. The paper contains many interesting results and masters several complicated proof techniques. In particular, it is shown that confident partial learning is more powerful than explanatory learning. On the other hand, the authors show that there are behaviorally correct learnable classes which are *not* confidently partially learnable. So, the learning model is also not trivial in the sense that it can infer every recursive function. Moreover, confident partial learning has another interesting property, i.e., it is closed under finite unions. The authors then study confident partial learning with respect to oracles, and obtain some deep results. That is, in addition to the successively fed graph of the the target function, the learner has access to an oracle. The second part of the paper combines partial learning with consistency. Since a consistent learner is preferable, these results deserve attention. On the positive site it is shown that every behaviorally correct learnable class is also is essentially class consistently partially learnable. On the other hand, the set of all recursive predicates is not essentially class consistently partially learnable. Finally, it is shown that PA-oracles are sufficient in order to partially learn every recursive function essentially class consistently.

The paper *Automatic Learning from Positive Data and Negative Counterexamples* by Sanjay Jain and Efim Kinber deals with the inductive inference of languages. So the target is a formal language and the information given to the learner may be eventually all strings in the language (positive examples only), all strings over the underlying alphabet which are then marked with respect to their containment in the target language, or, as in the present paper, positive examples and negative counterexamples (but not all). This source of information is justified by two facts. First, learning from positive examples only is often too weak, and receiving potentially all strings does not reflect, e.g., natural language acquisition. Again, one has to study the problem of inferring all languages from a given target class of languages by one learner. In their paper Jain and Kinber consider classes of target languages that are required to be automatic ones. That is, the authors consider classes of regular languages of the form $(L_i)_{i\in I}$ such that $\{(i,x) \mid x \in L_i\}$ and I itself are regular sets. So automatic classes of languages are a particular type of an automatic structure. In this context it should be noted that the family of automatic classes which are inferable from positive examples only is rather restricted. Thus, it is natural to ask under which conditions *all* automatic classes are automatically learnable. Here automatically learnable means that the learner itself must be describable by an automatic structure. The rest of the model is *mutatis mutandis* the same as in the inductive inference of recursive functions. However, since the learner is required to be automatic, it can obviously not memorize all data. So, the constraint to learn iteratively and/or with a bounded long term memory is very natural in this context. Here iterative means that the learner can just store the last example seen. The authors distinguish between least counterexamples (a shortest possible one), bounded counterexamples (bounded in the size of the longest positive example seen so far) and arbitrary counterexamples. The first main result is that all automatic classes are automatically learnable (iteratively and with bounded long term memory, respectively) from positive examples and arbitrary counterexamples. Furthermore, there are automatic classes that *cannot* be learned from positive examples and bounded counterexamples. The authors show many more results for which we refer the reader to the paper.

Christophe Costa Florêncio and Sicco Verwer in *Regular Inference as Vertex Coloring* also study a problem that belongs to the inductive inference of formal languages, i.e., learning the class of all regular languages from complete data in the limit. The hypothesis space chosen is the set of all deterministic finite automata (abbr. DFA). In this context it is known that it suffices to output in each learning step a minimal DFA that is consistent with all the data seen so far. This is, however, easier said than done, since the problem is known to be *NP*-complete. Thus the idea is to reduce the learning problem to satisfiability and to exploit the enormous progress made for satisfiability solvers. The approach undertaken previously is to perform this in two steps, i.e., first the learning problem is translated into a graph coloring problem, and second the graph coloring problem obtained is translated into a satisfiability problem. Here the first step included some inequality constraints (requiring the constraint vertices to have a different color) as well as some equality constraints. So, these constraints had to be translated into the resulting satisfiability problem. The main contribution of the present paper is an improvement for the first step that allows for a direct translation of the inference problem into a graph coloring problem. In this way, one can also directly use sophisticated solvers for graph coloring.

Teaching and PAC–Learning. Each learning model specifies the learner, the learning domain, the source of information, the hypothesis space, what background knowledge is available and how it can be used, and finally, the criterion of success. While the learner is always an algorithm, it may also be restricted in one way or another, e.g., by requiring it to be space and/or time efficient.

A significant line of work over the past decade studies combinatorial measures of the complexity of teaching. In this framework a helpful teacher chooses an informative sequence of labeled examples and provides them to the learner, with the goal of uniquely specifying the target concept from some a priori concept class of possible target functions. Several different combinatorial parameters related to this framework have been defined and studied, including the worst-case teaching dimension, the average teaching dimension, and the "recursive teaching dimension".

Rahim Samei, Pavel Semukhin, Boting Yang and Sandra Zilles in *Sauer's Bound for a Notion of Teaching Complexity* show that Sauer's Lemma can be adjusted to the recursive teaching dimension of the concept. This paper establishes an upper bound on the size of a concept class with given recursive teaching dimension. The upper bound coincides with Sauer's well-known bound on classes with a fixed VC-dimension. They further introduce and study classes whose size meets the upper bound and other properties of this measure.

It is well known that the language accepted by an unknown deterministic finite automata can be efficiently PAC-learnable if membership queries are allowed. It is also well known that cryptographic lower bounds preclude the efficient PAC learnability of arbitrary DFAs when membership queries are not allowed and learning must be based only on random examples. It is natural to ask about whether specific restricted types of regular languages are PAC learnable. A shuffle ideal generated by a string u is simply the collection of all strings containing u as a (discontiguous) subsequence.

The paper *On the Learnability of Shuffle Ideals* by Dana Angluin, James Aspnes, and Aryeh Kontorovich shows that shuffle ideal languages are efficiently learnable from statistical queries under the uniform distribution, but not efficiently PAC-learnable, unless $RP = NP$.

Statistical Learning Theory and Classification. The simplest setting in which statistical learning theory takes place is the following. A training set (X_t, Y_t) of samples is given, where the outcomes Y_t depend on the instances X_t; the learner then has to construct some rule to predict new outcomes Y for new instances X. Often the training set is formed by independent and identically distributed samples and the new instance–outcome pairs are drawn independently from the same distribution. The simplest task is (binary) classification, which corresponds to the case where the outcomes Y are $\{0, 1\}$–valued. More abstract formulations of the learning task can be provided, based on a hypothesis space (gathering functions h that map instances to outcomes) and on a loss function (associating with each pair of predicted outcome $h(X)$ and observed outcome Y a measure of their divergence). We call generalization bounds the bounds on the expected loss of a prediction function $\hat{h}$ constructed on the training set and evaluated on a new independent random pair (X, Y). These bounds are often in terms of the hypothesis set (and in particular, of its so-called Vapnik-Chervonenkis dimension or of its Rademacher complexity).

Mehryar Mohri and Andres Muñoz Medina present a *New Analysis and Algorithm for Learning with Drifting Distributions*, that is, they consider the case where the distribution of the instance–outcomes pairs evolves with t. Their analysis relies on the notion of discrepancy, which is a loss-based measure of divergence. They prove performance bounds based on the Rademacher complexity of the hypothesis set and the discrepancy of distributions; these bounds improve upon previous ones based on the $\mathbb{L}_1$–distances between distributions.

Another twist on the simplest problem described above is formed by domain adaptation, which corresponds to the case where the test and training data generating distributions differ. Shai Ben-David and Ruth Urner's contribution is *On the Hardness of Covariate Shift Learning (and the Utility of Unlabeled Target Samples)*; the covariate shift setting refers to the assumption that outcomes Y are solely determined by instances X, and that the function linking the two elements is the same in both domains. Algorithms had been proposed in this setting but often with very few generalization guarantees. The authors show that, without strong prior knowledge about the training task, such guarantees are actually unachievable, unless the training set and the set of new instance–outcomes pairs are prohibitively large. However, the (necessarily large) set of new elements can be formed rather by mostly unlabeled instances, which are often much cheaper to obtain than instance–outcome pairs.

Hal Daumé III, Jeff M. Phillips, Avishek Saha, and Suresh Venkatasubramanian study *Efficient Protocols for Distributed Classification and Optimization.* Their contribution takes places within a general model for distributed learning that bounds the communication required for learning classifiers with ε error on linearly separable data adversarially distributed across nodes; this model was introduced by the authors in an earlier article and they elaborate on it here. Their main result is a two-party multiplicative-weight-update based protocol that uses $O\big(d^2 \log(1/\varepsilon)\big)$ words of communication to ε–optimally classify distributed data in arbitrary dimension d. This result extends to classification over k nodes with $O\big(kd^2 \log(1/\varepsilon)\big)$ words of communication. The proposed protocol is simple to implement and empirical results show its improved efficiency.

Relations between Models and Data. Data is the raw material for learning and is often handled through a model or a collection of models. But sometimes the available theoretical models can be partially wrong; or even worse, no such theoretical models exist and they need to be constructed from the data.

Standard Bayesian inference can behave suboptimally if the model is wrong. Peter Grünwald presents in his article *The Safe Bayesian: Learning the Learning Rate via the Mixability Gap* a modification of Bayesian inference which continues to achieve good rates with wrong models. The method adapts the Bayesian learning rate to the data, picking the rate minimizing the cumulative loss of sequential prediction by posterior randomization.

Clustering (the partition of data into meaningful categories) is one of the most widely used techniques in statistical data analysis. A recent trend of research in this field is concerned with so-called perturbation resilience assumptions. Lev Reyzin defines in *Data Stability in Clustering: A Closer Look* a new notion of stability that is implied by perturbation resilience and discusses the implications of assuming resilience or stability in the data; the strength of this resilience or stability is measured by a constant α. He shows that for even fairly small constants α, the data begins to have very strong structural properties, which makes the clustering task fairly trivial. When α approaches ≈ 5.7, the data begins to show what is called strict separation, where each point is closer to points in its own cluster than to points in other clusters.

Bandit Problems. Bandit problems form a model of repeated interaction between a learner and a stochastic environment. In its simplest formulation the learner is given a finite number of arms, each associated with an unknown probability distribution with bounded support. Whenever he pulls an arm he gets some reward, drawn independently at random according to its associated distribution; his objective is to maximize the obtained cumulative reward. To do so, a trade-off between testing sufficiently often all the arms (exploration) and pulling more often the seemingly better arms (exploitation) needs to be performed. A popular strategy, the UCB strategy, constructs confidence bounds for the expected reward of each arm and pulls at each round the arm with best upper confidence bound.

In their paper *Thompson Sampling: An Optimal Finite Time Analysis*, Emilie Kaufmann, Nathaniel Korda, and Rémi Munos study another, older, strategy, called Thompson sampling; it relies on a Bayesian estimation of the expected reward. The authors show that in the case of Bernoulli distributions of the rewards, this strategy is asymptotically optimal.

Ronald Ortner, Daniil Ryabko, Peter Auer, and Rémi Munos consider in their contribution *Regret Bounds for Restless Markov Bandits* a more difficult scenario in which the rewards produced by each arm are not independent and identically distributed anymore; they are governed by Markov chains, which take transitions independently of whether the learner pulls this arm or not. They derive $O(\sqrt{T})$ regret bounds, without formulating any assumption on the distributions of the Markov chains.

An application of these bandit problems is studied in *Minimax Number of Strata for Online Stratified Sampling given Noisy Samples* by Alexandra Carpentier and Rémi Munos: how to approximate the integral of a function f given a finite budget of n noisy evaluations of the function. This is done by resorting to an online stratified sampling performed by the algorithm Monte-Carlo UCB developed by the authors in an earlier article. In their contribution to this volume they show that this algorithm is minimax optimal both in terms of the number of samples n and in the number of strata K, up to logarithmic factors.

Online Prediction of Individual Sequences. Another setting of repeated interactions between a learner and an environment is formed by the setting of online prediction of individual sequences. However, here, the environment may also use a strategy to pick his actions. At each round, the learner suffers a loss (or a gain) that only depends on the pair of actions taken by the two players. The quality of the strategy of the learner is measured through its regret, that is, the difference between his cumulative loss and the cumulative loss that the best constant choice of an action would have obtained on the same sequence of actions of the environment. Simple and efficient strategies exist to control the regret when the range of the losses is known and the number of actions is not too large. The first two papers described below relax these requirements. The three other papers deal with refinements of the basic situation presented above: the third one studies how sharp regret bounds can be, the fourth one focuses on a refined notion of regret, and the fifth one considers the case where only a partial monitoring of the actions taken by the environment is available.

The article *Weighted Last-Step Min-Max Algorithm with Improved Sub-Logarithmic Regret* by Edward Moroshko and Koby Crammer takes place within the framework of online linear regression with the square loss. It proposes a development of Forster's last-step min-max algorithm for the case where the range of the choices of the environment is unknown.

Daiki Suehiro, Kohei Hatano, Shuji Kijima, Eiji Takimoto, and Kiyohito Nagano deal with a case where the set of actions of the learner is large but bears some structure. More precisely, their contribution *Online Prediction under Submodular Constraints* focuses on the case of an action set formed by the vertices of a polyhedron described by a submodular function. Examples of the general problem handled there include the cases of k–sets, (truncated) permutahedra, spanning trees, and k–forests.

Another line of research is to study how sharp the regret bounds can be. Eyal Gofer and Yishay Mansour focus in *Lower Bounds on Individual Sequence Regret* on lower bounds on the regret of algorithms only based on the cumulative losses of the actions, which include popular strategies. They characterize those with a nonnegative regret; they also show that any such algorithm obtaining in addition a refined $O(\sqrt{Q})$ upper bound in terms of quadratic variations of the losses must also suffer an $\Omega(\sqrt{Q})$ lower bound for any loss sequence with quadratic variation Q.

Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, and Vladimir Vovk take *A Closer Look at Adaptive Regret*, which is a refined notion a regret. It corresponds to measuring regret only on subintervals of time, that is, to assessing how well the algorithm approximates the best experts locally. They investigate two existing intuitive methods to derive algorithms with low adaptive regret, one based on specialist experts and the other based on restarts; they show that both methods lead to the same algorithm, namely Fixed Share, which was known for its tracking regret. They then perform a thorough analysis of the adaptive regret of Fixed Share and prove the optimality of this strategy in this context.

The setting of *Partial Monitoring with Side Information* by Gábor Bartók and Csaba Szepesvári is the following. At every round the learner only receives a partial feedback about the choice of the action taken by the environment. The interaction protocol relies on a fixed function f, unknown to the learner: The action taken by the environment is a vector x_t, which is revealed to the learner, and is then used to draw at random the final action J_t of the environment according to the distribution $f(x_t)$. Simultaneously and based on x_t, the learner chooses his action I_t. The only feedback he gets is drawn at random according to a distribution that depends only on I_t and J_t, but he does not get to see J_t. The authors define a notion of regret in this setting and show an algorithm to minimize it.

Other Models of Online Learning. This section gathers the contributions relative to online learning but that correspond neither to bandit problems nor to the prediction of individual sequences.

The goal of reinforcement learning is to construct algorithms that learn to act optimally, or nearly so, in unknown environments. Tor Lattimore and Marcus Hutter focus on finite-state discounted Markov decision processes (MDPs). More precisely, in *PAC Bounds for Discounted MDPs* they exhibit matching (up to logarithmic factors) upper and lower bounds on the sample-complexity of learning near-optimal behavior. These upper bounds are obtained for a modified version of algorithm UCRL.

Wouter M. Koolen and Vladimir Vovk present in *Buy Low, Sell High* a simplified setting of online trading where an investor trades in a single security. His objective is to get richer when the price of the security exhibits a large upcrossing without risking bankruptcy. They investigate payoff guarantees that are expressed in terms of the extremity of the upcrossings and obtain an exact and elegant characterization of the guarantees that can be achieved. Moreover, they derive a simple canonical strategy for each attainable guarantee.

Kernels methods consist of mapping the data points from their original space to a feature space, where the analysis and prediction are performed more efficiently; the obtained results are then mapped back into the original space. In their paper *Kernelization of Matrix Updates, When and How?* Manfred Warmuth, Wojciech Kotłowski, and Shuisheng Zhou define what it means for a learning algorithm to be kernelizable in the case where the instances are vectors, asymmetric matrices, and symmetric matrices, respectively. They characterize kernelizability in terms of an invariance of the algorithm to certain orthogonal transformations. They provide a number of examples in the online setting of how to apply their methods.

The paper *Predictive Complexity and Generalized Entropy Rate of Stationary Ergodic Processes* by Mrinalkanti Ghosh and Satyadev Nandakumar takes place in the framework of online prediction of binary outcomes. They use generalized entropy to study the loss rate of predictors when these outcomes are drawn according to stationary ergodic distributions. They use a game-theoretic viewpoint and first show that a notion of generalized entropy of a regular game is well-defined for stationary ergodic distributions. They then study predictive complexity, a generalization of Kolmogorov complexity. More precisely, they prove that when the predictive complexity of a restricted regular game exists, the average predictive complexity converges to the generalized entropy of the game almost everywhere with respect to the stationary ergodic distribution.

Declarative Modeling for Machine Learning and Data Mining

Luc De Raedt

Department of Computer Science, Katholieke Universiteit Leuven, Belgium

Abstract. Despite the popularity of machine learning and data mining today, it remains challenging to develop applications and software that incorporates machine learning or data mining techniques. This is because machine learning and data mining have focussed on developing high-performance algorithms for solving particular tasks rather than on developing general principles and techniques. I propose to alleviate these problems by applying the constraint programming methodology to machine learning and data mining and to specify machine learning and data mining problems as constraint satisfaction and optimization problems. What is essential is that the user be provided with a way to declaratively specify what the machine learning or data mining problem is rather than having to outline how that solution needs to be computed. This corresponds to a model + solver-based approach to machine learning and data mining, in which the user specifies the problem in a high level modeling language and the system automatically transforms such models into a format that can be used by a solver to efficiently generate a solution. This should be much easier for the user than having to implement or adapt an algorithm that computes a particular solution to a specific problem. Throughout the talk, I shall use illustrations from our work on constraint programming for itemset mining and probabilistic programming.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, p. 12, 2012.

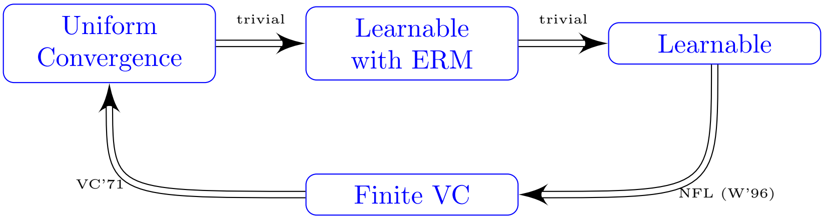
Learnability beyond Uniform Convergence

Shai Shalev-Shwartz

School of Computer Science and Engineering, The Hebrew University, Jerusalem

Abstract. The problem of characterizing learnability is the most basic question of statistical learning theory. A fundamental result is that learnability is equivalent to uniform convergence of the empirical risk to the population risk, and that if a problem is learnable, it is learnable via empirical risk minimization. The equivalence of uniform convergence and learnability was formally established only in the supervised classification and regression setting. We show that in (even slightly) more complex prediction problems learnability does not imply uniform convergence. We discuss several alternative attempts to characterize learnability. This extended abstract summarizes results published in [5, 3].

The fundamental theorem of learning theory states that for binary classification learning problems, learnability (in the PAC learning model of Valiant) is equivalent to uniform convergence of the empirical risk to the population risk. The components of this theorem are described below:



To be precise, we recall Vapnik's general setting of learning [6]: Let $\mathcal{Z}$ be a domain, $\mathcal{H}$ be a hypothesis class, and $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}$ be a loss function. Given a distribution $\mathcal{D}$ over $\mathcal{Z}$ we denote the risk of a hypothesis h by $L_{\mathcal{D}}(h) = \mathbb{E}_{z\sim\mathcal{D}}[\ell(h,z)]$. Given a sample $S = (z_1,\ldots,z_m) \sim \mathcal{D}^m$ we denote the empirical risk of h by $L_S(h) = \frac{1}{m}\sum_{i=1}^m \ell(h,z_i)$. The goal of the learner is to use S so as to find $h \in \mathcal{H}$ whose risk, $L_{\mathcal{D}}(h)$, is close to the minimum possible risk of a hypothesis in $\mathcal{H}$.

The special case of binary classification can be derived from the genreal setting by letting $\mathcal{Z} = \mathcal{X} \times \{0,1\}$, for some instance domain $\mathcal{X}$, letting $\mathcal{H}$ be a set of functions from $\mathcal{X}$ to $\{0,1\}$, and letting $\ell(h,(x,y)) = \mathbf{1}[h(x) \neq y]$ be the $0-1$ loss function.

Let us now recall the formal definitions of uniform convergence and learnability.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 13–16, 2012.

- **Uniform Convergence** with a sample complexity of $m_{\text{UC}}(\epsilon, \delta)$:
  For $m \geq m_{\text{UC}}(\epsilon, \delta)$,

$$\forall \mathcal{D}, \quad \mathop{\mathbb{P}}_{S \sim \mathcal{D}^m} \left[\forall h \in \mathcal{H},\ |L_S(h) - L_{\mathcal{D}}(h)| \leq \epsilon\right] \geq 1 - \delta$$

- **Learnable** with a sample complexity $m_{\text{PAC}}(\epsilon, \delta)$:
  $\exists \mathcal{A}$ s.t. for $m \geq m_{\text{PAC}}(\epsilon, \delta)$,

$$\forall \mathcal{D}, \quad \mathop{\mathbb{P}}_{S \sim \mathcal{D}^m} \left[L_{\mathcal{D}}(\mathcal{A}(S)) \leq \min_{h \in \mathcal{H}} L_{\mathcal{D}}(h) + \epsilon\right] \geq 1 - \delta$$

- **Empirical Risk Minimizer (ERM)**:
  An algorithm that returns $\mathcal{A}(S) \in \operatorname{argmin}_{h \in \mathcal{H}} L_S(h)$
- **Learnable by arbitrary ERM** with a rate $m_{\text{ERM}}(\epsilon, \delta)$:
  Like "Learnable" but $\mathcal{A}$ should be an ERM.

Getting back to the fundamental theorem of learning, it is trivial to see that uniform convergence with a rate of $m_{\text{UC}}(\epsilon, \delta)$ yields learnability by ERM with a rate of $m_{\text{ERM}}(\epsilon, \delta) \leq m_{\text{UC}}(\epsilon/2, \delta)$, which yields learnability with a rate of $m_{\text{PAC}}(\epsilon, \delta) \leq m_{\text{ERM}}(\epsilon, \delta)$.

For binary classification problems, the definition of the Vapnik-Chervonenkis (VC) dimension, together with the well known No-Free-Lunch theorem, yields that if a hypothesis class is learnable it must have a finite VC dimension. The seminal theorem of Vapnik and Chervonenkis shows that finite VC dimension yields learnability and by that we close our equivalence loop.

Since the 70's, following Vapnik and Chervonenkis's fundamental work on binary classification, it was widely believed that excluding trivialities, if a problem is at all learnable then uniform convergence holds and it is also learnable by every ERM rule. However, the equivalence between learnability and uniform convergence has been formally derived only for binary classification and for regression problems [4, 2, 1].

In his book, Vapnik attempted to show that uniform convergence is in fact necessary for learnability in all cases. However, Vapnik noted that this is simply not true as there are "trivial" problems which are learnable but for which uniform convergence does not hold. Consider for example the case of a "minorizing function": Let $\mathcal{H}'$ be a class of binary classifiers with infinite VC dimension, let $\mathcal{H} = \mathcal{H}' \cup \{h_0\}$, and let

$$\ell(h, (x, y)) = \begin{cases} 1 & \text{if } h \neq h_0 \wedge h(x) \neq y \\ 1/2 & \text{if } h \neq h_0 \wedge h(x) = y \\ 0 & \text{if } h = h_0 \end{cases}$$

Clearly, there is no uniform convergence here ($m_{\text{UC}} = \infty$). However, the problem is trivially learnable by ERM ($m_{\text{ERM}} = 1$). This phenomenon has been illustrated in Vapnik's book:

This example shows that there exist trivial cases of consistency that depend on whether a given set of functions contains a minorizing function.

Therefore, any theory of consistency that uses the classical definition needs

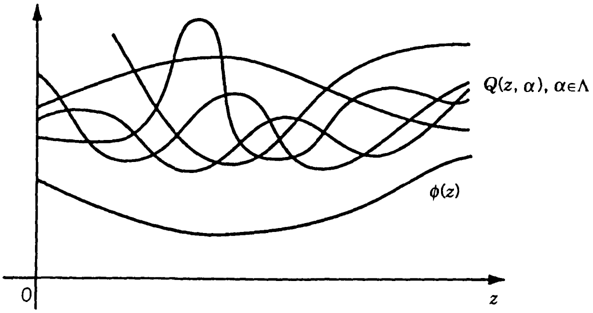


FIGURE 3.2. A case of trivial consistency. The ERM method is inconsistent on the set of functions $Q(z, \alpha)$, $\alpha \in \Lambda$, and is consistent on the set of functions $\phi(z) \bigcup Q(z, \alpha)$, $\alpha \in \Lambda$.

Vapnik referred to such cases as "trivial" learning problems. He defined a stronger notion of learnability, called *strict consistency*, and his "Key Theorem on Learning Theory" [6, Theorem 3.1] then states that *strict* consistency of empirical minimization is equivalent to one-sided[1] uniform convergence.

In [5] we have shown that learnability is not equivalent to uniform convergence even in non trivial learning problems, such as *stochastic convex optimization*, in which there is no dominating hypothesis that will always be selected. In fact some learning problems are learnable, but are not learnable by an ERM rule.

In [3] we have shown that this phenomenon reproduced even in *multiclass learning*, which is a supervised learning problem with the very same zero-one loss that is used in binary classification. We find this result surprising, as multiclass prediction is very close to binary classification.

These examples indicate that Vapnik's strict consistency might be too strict, which (re)rises the fundamental questions of "What problems are learnable?" and "How to learn?"

In [5] we make a first step and characterize learnability by the existence of an asymptotically ERM (AERM) algorithm, which is also stable. In [3] we try to determine the true sample complexity of multiclass learning and the optimal way to learn. Currently, we still do not have satisfactory answers to these questions. However, our analysis do give some hints regarding those questions, enabling us to prove that for the (important) case of *symmetric* hypothesis class, the sample complexity is characterized by a combinatorial measure called the Natarajan dimension. We conjecture that this result holds for non-symmetric classes as well.

[1] "One-sided" meaning requiring only $\sup_h (L_{\mathcal{D}}(h) - L_{\mathcal{S}}(h)) \longrightarrow 0$, rather then $\sup_h |L_{\mathcal{D}}(h) - F_{\mathcal{S}}(h)| \longrightarrow 0$.

References

[1] Alon, N., Ben-David, S., Cesa-Bianchi, N., Haussler, D.: Scale-sensitive dimensions, uniform convergence, and learnability. Journal of the ACM (JACM) 44(4), 615–631 (1997)

[2] Bartlett, P.L., Long, P.M., Williamson, R.C.: Fat-shattering and the learnability of real-valued functions. Journal of Computer and System Sciences 52(3), 434–452 (1996)

[3] Daniely, A., Sabato, S., Ben-David, S., Shalev-Shwartz, S.: Multiclass learnability and the erm principle. In: COLT (2011)

[4] Kearns, M.J., Schapire, R.E., Sellie, L.M.: Toward efficient agnostic learning. Machine Learning 17, 115–141 (1994)

[5] Shalev-Shwartz, S., Shamir, O., Srebro, N., Sridharan, K.: Learnability, stability and uniform convergence. The Journal of Machine Learning Research 9999, 2635–2670 (2010)

[6] Vapnik, V.N.: Statistical Learning Theory. Wiley (1998)

Some Rates of Convergence for the Selected Lasso Estimator

Pascal Massart and Caroline Meynet

Département de Mathématiques, Faculté des Sciences d'Orsay,
Université Paris-Sud XI, 91405 Orsay cedex, France

Abstract. We consider the estimation of a function in some ordered finite or infinite dictionary. We focus on the selected Lasso estimator introduced by Massart and Meynet (2011) as an adaptation of the Lasso suited to deal with infinite dictionaries. We use the oracle inequality established by Massart and Meynet (2011) to derive rates of convergence of this estimator on a wide range of function classes described by interpolation spaces such as in Barron et al. (2008). The results highlight that the selected Lasso estimator is adaptive to the smoothness of the function to be estimated, contrary to the classical Lasso or the greedy algorithm considered by Barron et al. (2008). Moreover, we prove that the rates of convergence of this estimator are optimal in the orthonormal case.

1 Introduction

We consider the problem of estimating a regression function f belonging to a Hilbert space $\mathbb{H}$ by some finite linear combination $\hat{f} = \hat{\theta}.\phi := \sum_j \hat{\theta}_j \, \phi_j$ of a given dictionary $\mathcal{D} = \{\phi_j\}_j$ in $\mathbb{H}$. Here by dictionary we mean any (non necessarily orthonormal) family of elements in $\mathbb{H}$. We consider a fairly general Gaussian learning framework which includes the fixed design regression or the white noise frameworks. The purpose is to construct estimators which enjoy both good statistical properties and computational performance even for large or infinite dictionaries.

For high-dimensional dictionaries, direct minimization of the empirical risk can lead to overfitting and one needs to add a penalty to avoid it. From a purely abstract view point an appropriate solution would be to use an ℓ_0-penalty by penalizing the number of non-zero coefficients $\hat{\theta}_j$ of $\hat{f}$ (see Birgé and Massart, 2001, for instance) so as to produce sparse estimators and interpretable models, but this minimization problem is non-convex and thus typically computationally unfeasible when the size of the dictionary becomes too large.

Some computationally efficient algorithms have been proposed during these last ten years that aim at mimicking ℓ_0-penalization. From a mathematical view point the main issue is to analyze the statistical performance of such procedures. The paper by Barron et al. (2008) perfectly illustrates the rather intensive research activity that has been performed recently in this direction. It is shown there that some commonly used greedy algorithms are achieving optimal rates

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 17–33, 2012.

of convergence (up to some logarithmic factors) on some properly defined Besov type spaces. Another candidate to be considered (at least for a finite dictionary) is the ℓ_1-penalization of least squares that leads to a tractable convex optimization problem. It is a natural candidate for at least two reasons: on the one hand the resulting procedure (the so-called Lasso) has been widely used in the recent years as surrogate for ℓ_0-penalization and on the other hand the main argument in Barron et al. (2008) consists of comparing the performance of greedy algorithms with a deterministic Lasso procedure.

In the spirit of Barron et al. (2008) we recently proved in Massart and Meynet (2011) that provided that the regularization parameter is properly chosen, the (noisy) Lasso performs almost as well as the deterministic Lasso. Note that this ℓ_1-result requires no assumption neither on the unknown target function nor on the variables ϕ_j of the dictionary (except simple normalization that we can always assume by considering $\phi_j / \|\phi_j\|$ instead of ϕ_j), contrary to the usual ℓ_0-oracle inequalities in the literature that are valid only under restrictive though unavoidable conditions (see Bickel et al., 2009, for instance). We derived this ℓ_1-oracle inequality from a fairly general model selection theorem for non linear models, interpreting ℓ_1-regularization as an ℓ_1-balls model selection criterion. Our approach allows to go one step further than the analysis of the Lasso for finite dictionaries and to deal with infinite dictionaries in various situations, leading to new procedures that we called selected Lasso because it simply consists of choosing from the data the size of a finite subdictionary on which the Lasso procedure is constructed. For an orthonormal dictionary, the resulting procedure is nothing else than a soft-thresholding with an adaptive threshold. Our purpose is here to analyze the rates of convergence of the selected Lasso on some properly defined Besov type spaces and show that it is fully adaptive (without extra logarithmic factors). This (small) gain as compared to the performance bounds for greedy algorithms obtained in Barron et al. (2008) comes from the adaptive choice of the size of the subdictionary. Let us now get in more details into the presentation of the framework and of the algorithms.

1.1 General Framework and Statistical Problem

Let $\mathbb{H}$ be a separable Hilbert space equipped with a scalar product $\langle ., . \rangle$ and its associated norm $\|.\|$. The statistical problem we consider is to estimate an unknown target function s in $\mathbb{H}$ when observing a process $(Y(t))_{t \in \mathbb{H}}$ defined by

$$Y(t) = \langle s, t \rangle + \varepsilon W(t), \quad t \in \mathbb{H}, \tag{1}$$

where $\varepsilon > 0$ is a fixed parameter and $(W(t))_{t \in \mathbb{H}}$ is an isonormal process, that is to say a centered Gaussian process with covariance given by $\mathbb{E}[W(u)W(t)] = \langle u, t \rangle$ for all $u, t \in \mathbb{H}$.

This framework is convenient to cover both finite-dimensional models and the infinite-dimensional white noise model as described in the following examples.

Example 1. [**Fixed design Gaussian regression model**] Let $\mathcal{X}$ be a measurable space. One observes n i.i.d. random couples $(x_1, Y_1), \dots, (x_n, Y_n)$ of $\mathcal{X} \times \mathbb{R}$ such that

$$Y_i = s(x_i) + \sigma\xi_i, \quad i = 1, \dots, n, \tag{2}$$

where the covariates $x_1, \dots, x_n$ are deterministic elements of $\mathcal{X}$, the errors ξ_i are i.i.d. $\mathcal{N}(0,1)$, $\sigma > 0$ and $s : \mathcal{X} \mapsto \mathbb{R}$ is the unknown regression function to be estimated. If one considers $\mathbb{H} = \mathbb{R}^n$ equipped with the scalar product $\langle u, v\rangle = \sum_{i=1}^n u_i\, v_i/n$, defines $y = (Y_1, \dots, Y_n)$, $\xi = (\xi_1, \dots, \xi_n)$ and denotes $t = (t(x_1), \dots, t(x_n))$ for every $t : \mathcal{X} \mapsto \mathbb{R}$, then $W(t) := \sqrt{n}\,\langle \xi, t\rangle$ defines an isonormal Gaussian process on $\mathbb{H}$ and $Y(t) := \langle y, t\rangle$ satisfies (1) with $\varepsilon = \sigma/\sqrt{n}$. In this case,

$$\|t\| = \sqrt{\frac{1}{n}\sum_{i=1}^{n} t^2(x_i)}\ . \tag{3}$$

Example 2. [**The white noise framework**] For $x \in [0,1]$, one observes $\zeta(x)$ given by the stochastic differential equation

$$d\zeta(x) = s(x)\,dx + \varepsilon\,dB(x) \text{ with } \zeta(0) = 0,$$

where B is a standard Brownian motion, s is a square-integrable function and $\varepsilon > 0$. Define $W(t) = \int_0^1 t(x)\,dB(x)$ for every $t \in \mathbb{L}_2([0,1])$. Then, W is an isonormal process on $\mathbb{H} = \mathbb{L}_2([0,1])$, and $Y(t) = \int_0^1 t(x)\,d\zeta(x)$ obeys to (1) if $\mathbb{H}$ is equipped with its usual scalar product $\langle s, t\rangle = \int_0^1 s(x)t(x)\,dx$. Typically, s is a signal and $d\zeta(x)$ represents the noisy signal received at time x. This framework easily extends to a d-dimensional setting if one considers some multivariate Brownian sheet B on $[0,1]^d$ and takes $\mathbb{H} = \mathbb{L}_2([0,1]^d)$.

To solve the general statistical problem (1), we introduce a dictionary $\mathcal{D}$, i.e. a given finite or infinite set of functions $\phi_j \in \mathbb{H}$ that arise as candidate basis functions for estimating the target function s, and consider estimators $\hat{s} = \hat{\alpha}.\phi := \sum_{j,\,\phi_j\in\mathcal{D}} \hat{\alpha}_j\,\phi_j$ in the linear span of $\mathcal{D}$. All the matter is to choose a "good" linear combination in the following meaning. It makes sense to aim at constructing an estimator as the best approximating point of s by minimizing $\|s - t\|$ or, equivalently, $-2\langle s, t\rangle + \|t\|^2$. However s is unknown, so one may instead minimize the empirical least squares criterion

$$\gamma(t) := -2Y(t) + \|t\|^2. \tag{4}$$

But, for high-dimensional data, direct minimization of the empirical least squares criterion can lead to overfitting. To avoid it, one can rather consider a penalized risk minimization problem and estimate s by

$$\hat{s} \in \arg\min_t \left\{\gamma(t) + \operatorname{pen}(t)\right\}, \tag{5}$$

where $\operatorname{pen}(t)$ is a positive penalty to be chosen according to the statistical goal.

Due to computer progress and development of state of the art technologies such as DNA microarrays, we are faced with high-dimensional data where the number of variables can be much larger than the sample size. To solve this problem, the sparsity scenario has been widely studied. It consists in assuming that there exists a sparse representation of the function s in the dictionary $\mathcal{D}$, that is to say that most coefficients $\hat{\alpha}_j$ can be taken to zero. Then, one could consider an ℓ_0-penalty in (5) in order to penalize the number of non-zero coefficients α_j and favor a sparse estimation of s. But there is no efficient algorithm to solve this non-convex minimization problem when the size of the dictionary becomes too large. So, alternative penalizations are to be considered to overcome this numerical problem. These last years, a great deal of attention has been focused on ℓ_1-penalization and its associated estimator the so-called Lasso (Tibshirani, 1996). This interest has been motivated by the geometric properties of the ℓ_1-norm: ℓ_1-penalization tends to produce sparse solutions and can thus be used as a convex surrogate for the non-convex ℓ_0-penalization.

1.2 The Lasso for Finite Dictionaries

For a finite dictionary $\mathcal{D}_p = \{\phi_1, \ldots, \phi_p\}$ of size p, the Lasso estimator of s is defined by

$$\hat{s}_p := \hat{s}(\lambda_p) = \operatorname*{arg\,min}_{t \in \mathcal{L}_1(\mathcal{D}_p)} \left\{ \gamma(t) + \lambda_p \|t\|_{\mathcal{L}_1(\mathcal{D}_p)} \right\}, \tag{6}$$

where $\gamma(t)$ is defined by (4), $\lambda_p > 0$ is a regularization parameter and

$$\|t\|_{\mathcal{L}_1(\mathcal{D}_p)} := \inf \left\{ \|\alpha\|_1 = \sum_{j=1}^{p} |\alpha_j| \ ; \ \alpha \in \mathbb{R}^p \text{ such that } t = \alpha.\phi \right\} \tag{7}$$

is the ℓ_1-norm of any function t in the linear span of $\mathcal{D}_p$. Lots of studies have been carried out on this estimator. In a sparsity viewpoint, ℓ_0-oracle inequalities have been proved to study the performance of this estimator as a variable selection procedure (Bickel et al., 2009; van de Geer, 2008; Koltchinskii, 2009). In parallel, a few results on the performance of the Lasso for its ℓ_1-regularization properties have been established (Bartlett et al., 2012; Huang et al., 2008; Massart and Meynet, 2011; Rigollet and Tsybakov, 2011). All these results are valid only for a regularization parameter of order

$$\lambda_p \gtrsim \sqrt{\frac{\ln p}{n}}. \tag{8}$$

1.3 The Selected Lasso Estimator

The results established for the Lasso in finite dictionaries are usually impossible to extend to infinite dictionaries because there is no longer size p for infinite dictionaries so that one can no longer calibrate the regularization parameter as it is done in (8). Therefore, it is difficult to evaluate the theoretical performance of the Lasso for infinite dictionaries. To solve this problem and deal with infinite dictionaries, Massart and Meynet (2011) proposed an estimator – the selected Lasso estimator– which is an adaptation of the Lasso suited to infinite countable ordered dictionaries. Their idea is the following. Given an infinite countable ordered[1] dictionary $\mathcal{D} = \{\phi_j\}_{j\in\mathbb{N}^\star} = \{\phi_1, \phi_2, \dots\}$, they consider the dyadic sequence of truncated dictionaries $\mathcal{D}_1 \subset \cdots \subset \mathcal{D}_p \subset \cdots \subset \mathcal{D}$ with $\mathcal{D}_p = \{\phi_1, \dots, \phi_p\}$ for $p \in \Lambda := \{2^J, J \in \mathbb{N}\}$. Given this sequence $(\mathcal{D}_p)_p$, they introduce the associated sequence of Lasso estimators $(\hat{s}_p)_p$ defined by (6), and choose

$$\hat{p} = \operatorname*{arg\,min}_{p\in\Lambda} \left\{\gamma(\hat{s}_p) + \lambda_p \|\hat{s}_p\|_{\mathcal{L}_1(\mathcal{D}_p)} + \mathrm{pen}(p)\right\} \tag{9}$$

$$= \operatorname*{arg\,min}_{p\in\Lambda} \left\{ \operatorname*{arg\,min}_{t\in\mathcal{L}_1(\mathcal{D}_p)} \left\{\gamma(t) + \lambda_p\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\} + \mathrm{pen}(p)\right\}, \tag{10}$$

where $\mathrm{pen}(p)$ is a penalty to be chosen to penalize the size p of the truncated dictionary $\mathcal{D}_p$ for all $p \in \Lambda$. Then, they take $\hat{s}_{\hat{p}}$ as final estimator. This selected Lasso estimator $\hat{s}_{\hat{p}}$ is based on an algorithm choosing automatically the level of truncation of the dictionary $\mathcal{D}$ making the best tradeoff between approximation, ℓ_1-regularization and sparsity.

Although introduced for infinite dictionaries, this estimator remains well-defined for finite dictionaries and it may be profitable to use it rather than the classical Lasso for such dictionaries. In particular, the definition of $\hat{s}_{\hat{p}}$ guarantees that $\hat{s}_{\hat{p}}$ makes a better tradeoff between approximation, ℓ_1-regularization and sparsity than the Lasso and that it is always sparser than the Lasso.

From a theoretical point of view, Massart and Meynet (2011) established an oracle inequality satisfied by this selected Lasso estimator:

Theorem 1. *Assume that* $\sup_{j\in\mathbb{N}^\star} \|\phi_j\| \leq 1$. *Set for all* $p \in \Lambda$,

$$\lambda_p = c_1\varepsilon\left(\sqrt{\ln p} + 1\right), \qquad \mathrm{pen}(p) = c_2\varepsilon^2 \ln p, \tag{11}$$

where $c_1 \geq 4$ *and* $c_2 > c_1/\sqrt{\ln 2}$. *Let* $\hat{s}_{\hat{p}}$ *be the selected Lasso estimator defined by* (10).
Then, there exists an absolute constant $C > 0$ *such that*

$$\mathbb{E}\left[\|s - \hat{s}_{\hat{p}}\|^2 + \lambda_{\hat{p}}\|\hat{s}_{\hat{p}}\|_{\mathcal{L}_1(\mathcal{D}_{\hat{p}})} + \mathrm{pen}(\hat{p})\right]$$
$$\leq C\left[\inf_{p\in\Lambda}\left\{\inf_{t\in\mathcal{L}_1(\mathcal{D}_p)}\left\{\|s-t\|^2 + \lambda_p\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\} + \mathrm{pen}(p)\right\} + \varepsilon^2\right]. \tag{12}$$

1.4 Our Contribution

In this article, we use Theorem 1 to derive rates of convergence of the selected Lasso estimator. First, we restrict to orthonormal dictionaries for a target function s in the intersection between a weak $\mathcal{L}_q$ space and a Besov space. In this

[1] Ordering the variables can be more or less difficult according to the problem under consideration. For some applications, such as decomposition in wavelet dictionaries, the variables may be naturally ordered.

case, we establish both an upper bound of the risk of the selected Lasso estimator and a lower bound of the minimax risk to check that the rates of convergence achieved by this estimator are optimal. Then, we extend our upper bound to the non-orthonormal case.

The article is organized as follows. We present the rates of convergence established for the selected Lasso estimator in Section 2. The proofs are detailed in Section 3.

2 Some Rates of Convergence for the Selected Lasso Estimator

We establish rates of convergence for the selected Lasso estimator for a wide range of function classes described by interpolation spaces. They are derived from the oracle inequality (12). We consider the framework and notations introduced in Section 1. In particular, we consider a Hilbert space $\mathbb{H}$ and an infinite countable dictionary $\mathcal{D} = \{\phi_j\}_{j \in \mathbb{N}^\star}$ which is a basis of $\mathbb{H}$.

2.1 Orthonormal Dictionaries

Here, we assume that $\mathcal{D}$ is an orthonormal basis of $\mathbb{H}$.

Definition of the Spaces. We say that a function u belongs to $w\mathcal{L}_q(R)$ for some $1 < q < 2$ and $R > 0$ if $u = \sum_{j=1}^{\infty} \alpha_j \, \phi_j$ with coefficients α_j in the weak ℓ_q-balls of radius R:

$$\sup_{\eta > 0} \left(\eta^q \sum_{j=1}^{\infty} \mathbb{1}_{\{|\alpha_j| > \eta\}} \right) \leq R^q. \tag{13}$$

We say that u belongs to the Besov space $\mathcal{B}^r_{2,\infty}(R)$ with radius R if we have the following control of the high-level components of u in the orthonormal basis $\mathcal{D}$:

$$\sup_{J \in \mathbb{N}^\star} \left(J^{2r} \sum_{j=J}^{\infty} \alpha_j^2 \right) \leq R^2. \tag{14}$$

Upper Bound of the Quadratic Risk

Proposition 1. *Assume that the dictionary $\mathcal{D}$ is an orthonormal basis of the Hilbert space $\mathbb{H}$. Let $1 < q < 2$, $r > 0$, $R > 0$ such that $R\varepsilon^{-1} \geq \mathrm{e}$, and assume that $s \in w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$. Consider the selected Lasso estimator $\hat{s}_{\hat{p}}$ defined by* (10) *with parameters λ_p and* pen(p) *given by* (11).
Then, there exists $C_{q,r} > 0$ depending only on q and r such that the quadratic risk of $\hat{s}_{\hat{p}}$ satisfies

$$\mathbb{E}\left[\|s - \hat{s}_{\hat{p}}\|^2\right] \leq C_{q,r} \, R^q \left(\varepsilon \sqrt{\ln\left(R\varepsilon^{-1}\right)}\right)^{2-q}. \tag{15}$$

Proof. Page 26.

Remark 1. The assumption $R\varepsilon^{-1} \geq \mathrm{e}$ of Proposition 1 is not restrictive since it only means that we consider non-degenerate situations where the signal to noise ratio is large enough, which is the only interesting case to use the selected Lasso estimator. Indeed, if $R\varepsilon^{-1}$ is too small, then the estimator equal to zero will always be better than any other non-zero estimators, in particular the selected Lasso estimator.

The lower bound (15) is to be compared with the rates of convergence achieved by the Lasso estimator $\hat{s}_p$ defined by (6) for any fixed value p. For orthonormal dictionaries, the Lasso estimators are soft-thresholding estimators with a fixed threshold determined by the level of truncation of the dictionary, while the selected Lasso estimator is a soft-thresholding estimator with an adaptive threshold automatically chosen by the algorithm constructing this estimator. So, the bound (15) is to be compared with the rates of convergence achieved by the soft-thresholding estimators with a fixed threshold when the target function belongs to $w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$. From Rivoirard (2006, Theorem 1), the rates achieved by the soft-thresholding estimators with a fixed threshold strongly depend on the parameter of smoothness r and are valid only for values of r large enough compared to the level of truncation of the dictionary. On the contrary, Proposition 1 shows that the rates achieved by the selected Lasso estimator are valid whatever the value of $r > 0$ and that this smoothness parameter has little effect on the rates since it only appears through the multiplicative factor $C_{q,r}$. Thus, Proposition 1 highlights the major advantage of the selected Lasso estimator over the classical Lasso estimators which is its adaptability to the unknown parameters of smoothness q and r of the target function. This adaptability comes from the fact that the selected Lasso estimator is constructed from an algorithm choosing an adaptive level of truncation of the dictionary.

Lower Bound of the Minimax Risk. We now establish a lower bound of the minimax risk over the balls $w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$ to prove that the rates of convergence (15) are optimal. We even establish a stronger result by providing the lower bound of the minimax risk over the smaller balls $\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R) \subset w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$, where we denote by $\mathcal{L}_q(R)$ the set of functions whose coefficients in the orthonormal basis $\mathcal{D} = \{\phi_j\}_{j\in\mathbb{N}^\star}$ are in the ℓ_q-ball of radius R, that is to say functions $\sum_{j=1}^{\infty} \alpha_j\, \phi_j$ such that $\sum_{j=1}^{\infty} |\alpha_j|^q \leq R^q$.

Proposition 2. *Assume that the dictionary $\mathcal{D}$ is an orthonormal basis of $\mathbb{H}$. Let $1 < q < 2$, $0 < r < 1/q - 1/2$ and $R > 0$ such that $R\varepsilon^{-1} \geq \max(\mathrm{e}^2, \varsigma^2)$ where*

$$\varsigma := \frac{1}{r} - q\left(1 + \frac{1}{2r}\right) > 0. \tag{16}$$

Then, there exists an absolute constant $\kappa > 0$ such that the minimax risk over $\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$ satisfies

$$\inf_{\tilde{s}} \sup_{s \in \mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)} \mathbb{E}\left[\|s - \tilde{s}\|^2\right] \geq \kappa \, \varsigma^{1-\frac{q}{2}} \, R^q \left(\varepsilon \sqrt{\ln\left(R\varepsilon^{-1}\right)}\right)^{2-q}, \tag{17}$$

where the infimum is taken over all possible estimators $\tilde{s}$.

Proof. Page 29.

Remark 2. The constraint $r < 1/q - 1/2$ of Proposition 2 is necessary to work on the intersection between an $\mathcal{L}_q$-ball and a Besov ball. Indeed, assume that $r > 1/q - 1/2$. For all $R > 0$, put $R' = (1 - 2^{r\varsigma})^{1/q} R$ where ς is defined by (16). Then, it is easy to check that $\mathcal{B}^r_{2,\infty}(R') \subset \mathcal{L}_q(R)$. Thus, $\mathcal{B}^r_{2,\infty}(R') = \mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R')$. Moreover, $R' < R$, so $\mathcal{B}^r_{2,\infty}(R') \subset \mathcal{B}^r_{2,\infty}(R)$ and $\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R') \subset \mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$. Consequently, $\mathcal{B}^r_{2,\infty}(R') \subset \mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R) \subset \mathcal{B}^r_{2,\infty}(R)$: the intersection $\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$ is no longer a real intersection between an $\mathcal{L}_q$-ball and a Besov ball but rather a Besov ball itself.

The upper bound (15) and the lower bound (17) match up to a constant. This proves that the selected Lasso estimator is simultaneously approximately minimax over $w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$ for suitable signal to noise ratio $R\varepsilon^{-1}$ in the orthonormal case.

2.2 Non-orthonormal Dictionaries

Here, we no longer assume that $\mathcal{D}$ is orthonormal. We extend the upper bound (15) of the quadratic risk of this estimator when assuming that the target function belongs to some real interpolation spaces that are extensions of the spaces $w\mathcal{L}_q \cap \mathcal{B}^r_{2,\infty}$ considered in the orthonormal case.

Definition of the Interpolation Spaces. We introduce a whole range of interpolation spaces $\mathcal{B}_{q,r}$ that are intermediate spaces between subsets of $\mathcal{L}_1(\mathcal{D})$ and the Hilbert space $\mathbb{H}$.

Definition 1. [Spaces $\mathcal{L}_{1,r}$ and $\mathcal{B}_{q,r}$] *Let $R > 0$, $r > 0$, $1 < q < 2$ and $\nu = 1/q - 1/2$.*
We say that $u \in \mathbb{H}$ belongs to $\mathcal{L}_{1,r}$ if there exists $C > 0$ such that for all $p \in \mathbb{N}^\star$, there exists $u_p \in \mathcal{L}_1(\mathcal{D}_p)$ such that

$$\|u_p\|_{\mathcal{L}_1(\mathcal{D}_p)} \leq C$$

and

$$\|u - u_p\| \leq C p^{-r}. \tag{18}$$

The smallest C such that this holds defines a norm $\|u\|_{\mathcal{L}_{1,r}}$ on the space $\mathcal{L}_{1,r}$.
We say that u belongs to $\mathcal{B}_{q,r}(R)$ if, for all $\delta > 0$,

$$\inf_{t \in \mathcal{L}_{1,r}} \left\{ \|u - t\| + \delta \|t\|_{\mathcal{L}_{1,r}} \right\} \leq R\,\delta^{2\nu}. \tag{19}$$

We say that $u \in \mathcal{B}_{q,r}$ if there exists $R > 0$ such that $u \in \mathcal{B}_{q,r}(R)$. In this case, the smallest R such that $u \in \mathcal{B}_{q,r}(R)$ defines a norm on the space $\mathcal{B}_{q,r}$ and is denoted by $\|u\|_{\mathcal{B}_{q,r}}$.

Remark 3. The abstract interpolation spaces $\mathcal{B}_{q,r}$ are in fact natural extensions of the spaces $w\mathcal{L}_q \cap \mathcal{B}^r_{2,\infty}$ for non-orthonormal dictionaries. Indeed, if $\mathcal{D}$ is an orthonormal basis of $\mathbb{H}$, then, for all $1 < q < 2$ and $r > 0$, there exists $C_{q,r} > 0$ depending only on q and r such that, for all $R > 0$,

$$w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R) \,\subset\, \mathcal{B}_{q,r}(C_{q,r}\,R). \tag{20}$$

Proof. Page 31.

Upper Bound of the Quadratic Risk

Proposition 3. *Assume that* $\sup_{j \in \mathbb{N}^\star} \|\phi_j\| \leq 1$. *Let $1 < q < 2$, $r > 0$, $R > 0$ such that $R\varepsilon^{-1} \geq \mathrm{e}$ and assume that $s \in \mathcal{B}_{q,r}(R)$. Consider the selected Lasso estimator $\hat{s}_{\hat{p}}$ defined by (10) with parameters λ_p and* $\mathrm{pen}(p)$ *given by (11).*
Then, there exists $C_{q,r} > 0$ depending only on q and r such that the quadratic risk of $\hat{s}_{\hat{p}}$ satisfies

$$\mathbb{E}\left[\|s - \hat{s}_{\hat{p}}\|^2\right] \leq C_{q,r}\,R^q \left(\varepsilon\sqrt{\ln\left(R\varepsilon^{-1}\right)}\right)^{2-q}. \tag{21}$$

Proof. Page 31.

Proposition 3 is to be compared with Proposition 1 established in the orthonormal case. Taking into account the inclusion (20) and noting that the upper bounds of the quadratric risk (15) and (21) are exactly of the same order and valid under the same assumption on the signal to noise ratio, we can conclude that Proposition 3 extends the result established in Proposition 1. Yet, we shall provide an independent proof of Proposition 1 in Appendix 3.1 to see how things work in the simpler orthonormal case.

Proposition 3 highlights the high performance of the selected Lasso estimator compared with other existing estimators in the theory of approximation and learning. In particular, (21) proves that the selected Lasso estimator performs as well as the greedy algorithms for which Barron et al. (2008) have provided similar rates of convergence. Besides, since the construction of the selected Lasso estimator is based on an adaptive truncation of the dictionary, this estimator has the great advantage of being adaptive to the unknown parameters of smoothness q and r of the target function, whereas the greedy algorithms achieve their rates of convergence only for restricted values of the parameter r depending on the level of truncation of the dictionary (Barron et al., 2008, Corollary 3.7).

3 Proofs

3.1 Orthonormal Dictionaries

Proof of the Upper Bound: Proposition 1. We know from Theorem 1 that the quadratic risk of the selected Lasso estimator $\hat{s}_{\hat{p}}$ is bounded by

$$\mathbb{E}\left[\|s-\hat{s}_{\hat{p}}\|^2\right] \leq C\left[\inf_{p\in\Lambda}\left\{\inf_{t\in\mathcal{L}_1(\mathcal{D}_p)}\left\{\|s-t\|^2+\lambda_p\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\}+\mathrm{pen}(p)\right\}+\varepsilon^2\right], \tag{22}$$

where C is an absolute positive constant. Now, thanks to the following lemma, we will bound $\inf_{t\in\mathcal{L}_1(\mathcal{D}_p)}\left\{\|s-t\|^2+\lambda_p\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\}$ for all $p\in\Lambda$.

Lemma 1. *Assume that the dictionary $\mathcal{D}$ is an orthonormal basis of the Hilbert space $\mathbb{H}$ and that there exist $1<q<2$, $r>0$ and $R>0$ such that $s\in w\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)$. For all $p\in\mathbb{N}^\star$ and $\lambda>0$, define*

$$s_{p,\lambda} := \underset{t\in\mathcal{L}_1(\mathcal{D}_p)}{\arg\min}\left\{\|s-t\|^2+\lambda\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\}.$$

Then, there exist $C_q>0$ depending only on q and $C_r>0$ depending only on r such that for all $p\in\mathbb{N}^\star$ and $\lambda>0$,

$$\|s_{p,\lambda}\|_{\mathcal{L}_1(\mathcal{D}_p)} \leq C_q R^q \lambda^{1-q}$$

and

$$\|s-s_{p,\lambda}\|^2 \leq C_r R^2 p^{-2r} + C_q R^q \lambda^{2-q}.$$

The proof of Lemma 1 uses the two following easy calculations.

Lemma 2. *For all $a=(a_1,\ldots,a_p)\in\mathbb{R}^p$ and $\delta>0$,*

$$\sum_{j=1}^p a_j^2\,\mathbb{1}_{\{|a_j|\leq\delta\}} \leq 2\sum_{j=1}^p\int_0^\delta t\,\mathbb{1}_{\{|a_j|>t\}}\,dt.$$

Proof.

$$\begin{aligned}
&2\sum_{j=1}^p\int_0^\delta t\,\mathbb{1}_{\{|a_j|>t\}}\,dt\\
&=2\sum_{j=1}^p\left[\left(\int_0^\delta t\,\mathbb{1}_{\{|a_j|>t\}}\,dt\right)\mathbb{1}_{\{|a_j|>\delta\}}+\left(\int_0^\delta t\,\mathbb{1}_{\{|a_j|>t\}}\,dt\right)\mathbb{1}_{\{|a_j|\leq\delta\}}\right]\\
&=2\sum_{j=1}^p\left[\left(\int_0^\delta t\,dt\right)\mathbb{1}_{\{|a_j|>\delta\}}+\left(\int_0^{|a_j|} t\,dt\right)\mathbb{1}_{\{|a_j|\leq\delta\}}\right]\\
&=\sum_{j=1}^p\left(\delta^2\,\mathbb{1}_{\{|a_j|>\delta\}}+a_j^2\,\mathbb{1}_{\{|a_j|\leq\delta\}}\right)\\
&\geq\sum_{j=1}^p a_j^2\,\mathbb{1}_{\{|a_j|\leq\delta\}}.
\end{aligned}$$

Lemma 3. *For all $a = (a_1, \dots, a_p) \in \mathbb{R}^p$ and $\delta > 0$,*

$$\sum_{j=1}^{p} |a_j| \mathbb{1}_{\{|a_j|>\delta\}} = \delta \sum_{j=1}^{p} \mathbb{1}_{\{|a_j|>\delta\}} + \sum_{j=1}^{p} \int_{\delta}^{+\infty} \mathbb{1}_{\{|a_j|>t\}}\, dt.$$

Proof.

$$\sum_{j=1}^{p} \int_{\delta}^{+\infty} \mathbb{1}_{\{|a_j|>t\}}\, dt = \sum_{j=1}^{p} \left(\int_{\delta}^{|a_j|} dt \right) \mathbb{1}_{\{|a_j|>\delta\}} = \sum_{j=1}^{p} (|a_j| - \delta)\, \mathbb{1}_{\{|a_j|>\delta\}}.$$

Proof of Lemma 1.
Let us denote by $\{\alpha_j^*\}_{j\in\mathbb{N}^\star}$ the coefficients of the target function s in the basis $\mathcal{D} = \{\phi_j\}_{j\in\mathbb{N}^\star}$, so that $s = \alpha^*.\phi = \sum_{j\in\mathbb{N}^\star} \alpha_j^*\, \phi_j$.
For all $p \in \mathbb{N}^\star$, set $A_p := \{\alpha = (\alpha_j)_{j\in\mathbb{N}^\star};\ \alpha_j \in \mathbb{R},\ \alpha_j = 0 \text{ for } j \geq p+1\}$.
Let $\lambda > 0$. Since $s_{p,\lambda} \in \mathcal{L}_1(\mathcal{D}_p)$, there exists $\alpha^{p,\lambda} \in A_p$ such that $s_{p,\lambda} = \alpha^{p,\lambda}.\phi$. Moreover, from (7) and the orthonormality of the basis $\mathcal{D}$,

$$\alpha^{p,\lambda} = \underset{\alpha\in A_p}{\arg\min} \left\{ \|\alpha^*.\phi - \alpha.\phi\|^2 + \lambda\|\alpha\|_1 \right\} = \underset{\alpha\in A_p}{\arg\min} \left\{ \|\alpha^* - \alpha\|^2 + \lambda\|\alpha\|_1 \right\}. \tag{23}$$

By calculating the subdifferential of the function $\alpha \in \mathbb{R}^p \mapsto \|\alpha^* - \alpha\|^2 + \lambda\|\alpha\|_1$, we get that the solution of the convex minimization problem (23) is $\alpha^{p,\lambda} = (\alpha_1^{p,\lambda}, \dots, \alpha_p^{p,\lambda}, 0, \dots, 0, \dots)$ where for all $j \in \{1, \dots, p\}$,

$$\alpha_j^{p,\lambda} = \begin{cases} \alpha_j^* - \lambda/2 & \text{if } \alpha_j^* > \lambda/2, \\ \alpha_j^* + \lambda/2 & \text{if } \alpha_j^* < -\lambda/2, \\ 0 & \text{otherwise.} \end{cases}$$

Then, we have

$$\begin{aligned} \|s - s_{p,\lambda}\|^2 &= \|\alpha^* - \alpha^{p,\lambda}\|^2 \\ &= \sum_{j=1}^{\infty} \left(\alpha_j^* - \alpha_j^{p,\lambda} \right)^2 \\ &= \sum_{j=p+1}^{\infty} {\alpha_j^*}^2 + \sum_{j=1}^{p} {\alpha_j^*}^2 \mathbb{1}_{\{|\alpha_j^*|\leq\lambda/2\}} + \sum_{j=1}^{p} \frac{\lambda^2}{4} \mathbb{1}_{\{|\alpha_j^*|>\lambda/2\}} \\ &\leq \underbrace{\sum_{j=p+1}^{\infty} {\alpha_j^*}^2}_{(i)} + \underbrace{\sum_{j=1}^{p} {\alpha_j^*}^2 \mathbb{1}_{\{|\alpha_j^*|\leq\lambda/2\}}}_{(ii)} + \frac{\lambda}{2} \underbrace{\sum_{j=1}^{p} |\alpha_j^*| \mathbb{1}_{\{|\alpha_j^*|>\lambda/2\}}}_{(iii)} \quad , \end{aligned} \tag{24}$$

while

$$\|s_{p,\lambda}\|_{\mathcal{L}_1(\mathcal{D}_p)} = \sum_{j=1}^{\infty} |\alpha_j^{p,\lambda}| = \sum_{j=1}^{p} \left(|\alpha_j^*| - \frac{\lambda}{2} \right) \mathbb{1}_{\{|\alpha_j^*|>\lambda/2\}} \leq \underbrace{\sum_{j=1}^{p} |\alpha_j^*| \mathbb{1}_{\{|\alpha_j^*|>\lambda/2\}}}_{(iii)} \ . \tag{25}$$

Now, s is assumed to belong to $\mathcal{B}^r_{2,\infty}(R)$, so (14) implies that (i) is bounded by

$$\sum_{j=p+1}^{\infty} \alpha_j^{*2} \leq R^2(p+1)^{-2r} \leq 2^{-2r}R^2p^{-2r}. \tag{26}$$

Let us now bound (ii) and (iii) thanks to the assumption $s \in w\mathcal{L}_q(R)$. By applying Lemma 2 and Lemma 3 with $a_j = \alpha_j^*$ for all $j \in \{1,\ldots,p\}$ and $\delta = \lambda/2$, and by using the fact that $\sum_{j=1}^p \mathbb{1}_{\{|\alpha_j^*|>t\}} \leq \sum_{j=1}^\infty \mathbb{1}_{\{|\alpha_j^*|>t\}} \leq R^q t^{-q}$ for all $t > 0$ if $s \in w\mathcal{L}_q(R)$, we get that (ii) is bounded by

$$\sum_{j=1}^{p} \alpha_j^{*2}\mathbb{1}_{\{|\alpha_j^*|\leq\lambda/2\}} \leq \frac{2^{q-1}}{2-q}\, R^q\lambda^{2-q}, \tag{27}$$

while (iii) is bounded by

$$\sum_{j=1}^{p} |\alpha_j^*|\mathbb{1}_{\{|\alpha_j^*|>\lambda/2\}} \leq \frac{q\,2^{q-1}}{q-1}\, R^q\lambda^{1-q}. \tag{28}$$

Gathering (25) and (28) on the one hand and (24), (26), (27) and (28) on the other hand, we get that there exists $C_q > 0$ depending only on q and $C_r > 0$ depending only on r such that $\|s_{p,\lambda}\|_{\mathcal{L}_1(\mathcal{D}_p)} \leq C_q R^q \lambda^{1-q}$ and $\|s - s_{p,\lambda}\|^2 \leq C_r R^2 p^{-2r} + C_q R^q \lambda^{2-q}$. □

Proof of Proposition 1.
We deduce from Theorem 1 and Lemma 1 that there exists some constant $C_{q,r} > 0$ depending only on q and r such that the quadratic risk of $\hat{s}_{\hat{p}}$ is bounded by

$$\begin{aligned}\mathbb{E}\left[\|s-\hat{s}_{\hat{p}}\|^2\right] &\leq C_{q,r}\left[\inf_{p\in\Lambda}\left\{R^2p^{-2r} + R^q\left(\varepsilon\left(\sqrt{\ln p}+1\right)\right)^{2-q} + \varepsilon^2\ln p\right\} + \varepsilon^2\right] \\ &\leq C_{q,r}\inf_{p\in\Lambda\setminus\{1\}}\left\{R^2p^{-2r} + R^q(\varepsilon\sqrt{\ln p})^{2-q} + \varepsilon^2\ln p\right\},\end{aligned} \tag{29}$$

where we use the fact that, for all $p \geq 2$, $\sqrt{\ln p}+1 \leq (1+1/\sqrt{\ln 2})\sqrt{\ln p}$ and $\varepsilon^2 \leq \varepsilon^2(\ln p)/\ln 2$. Now, we choose p such that the terms inside the infimum are of the same order. Denote by $\lceil x \rceil$ the smallest integer greater than x. Define $J_{q,r} = \lceil (2-q)(2r)^{-1}\log_2(R\varepsilon^{-1})\rceil$ and $p_{q,r} = 2^{J_{q,r}}$. Since we have assumed $R\varepsilon^{-1} \geq \mathrm{e}$, then $p_{q,r} \in \Lambda\setminus\{1\}$ and we deduce from (29) that

$$\mathbb{E}\left[\|s-\hat{s}_{\hat{p}}\|^2\right] \leq C_{q,r}\left(R^2p_{q,r}^{-2r} + R^q(\varepsilon\sqrt{\ln p_{q,r}})^{2-q} + \varepsilon^2\ln p_{q,r}\right). \tag{30}$$

Now, let us give an upper bound of each term of the right-hand side of (30). From the fact that $2 \leq \mathrm{e} \leq R\varepsilon^{-1}$ and by definition of $p_{q,r}$, on the one hand we have $p_{q,r} \geq (R\varepsilon^{-1})^{(2-q)/(2r)}$, while on the other hand we have

$$\ln p_{q,r} \leq \ln 2 + \frac{2-q}{2r}\ln\left(R\varepsilon^{-1}\right) \leq \left(1+\frac{2-q}{2r}\right)\ln\left(R\varepsilon^{-1}\right) := A_{q,r}\ln\left(R\varepsilon^{-1}\right)$$

where $A_{q,r} > 0$ depends only on q and r. Thus, we get that

$$R^2 p_{q,r}^{-2r} \leq R^2 \left(R\varepsilon^{-1}\right)^{q-2} = R^q \varepsilon^{2-q} \tag{31}$$

while

$$R^q \left(\varepsilon\sqrt{\ln p_{q,r}}\right)^{2-q} \leq {A_{q,r}}^{1-\frac{q}{2}} R^q \left(\varepsilon\sqrt{\ln\left(R\varepsilon^{-1}\right)}\right)^{2-q} \tag{32}$$

and

$$\varepsilon^2 \ln p_{q,r} \leq A_{q,r}\, \varepsilon^2 \ln\left(R\varepsilon^{-1}\right). \tag{33}$$

Now, these three bounds are upper bounded by $C_{q,r} R^q(\varepsilon\sqrt{\ln(R\varepsilon^{-1})})^{2-q}$ where $C_{q,r} > 0$ depends only on q and r. Indeed, $R\varepsilon^{-1} \geq \mathrm{e}$ and $2-q > 0$, so (31) is bounded by $R^q(\varepsilon\sqrt{\ln(R\varepsilon^{-1})})^{2-q}$. Moreover, the right-hand side of (33) can be written $A_{q,r}\left(g((R\varepsilon^{-1})^2)\right)^{q/2} R^q(\varepsilon\sqrt{\ln(R\varepsilon^{-1})})^{2-q}$ with $g :]0,+\infty[\mapsto \mathbb{R},\ x \mapsto \ln(x)/(2x)$. Using the fact that $g(x^2) \leq 1/(2x)$ for all $x > 0$ and that $R\varepsilon^{-1} \geq \mathrm{e}$, we get that (33) is bounded by $A_{q,r}(2\mathrm{e})^{-q/2} R^q(\varepsilon\sqrt{\ln(R\varepsilon^{-1})})^{2-q}$.

Then, we deduce from (30) that there exists $C_{q,r} > 0$ depending only on q and r such that

$$\mathbb{E}\left[\|s - \hat{s}_{\hat{p}}\|^2\right] \leq C_{q,r}\, R^q \left(\varepsilon\sqrt{\ln\left(R\varepsilon^{-1}\right)}\right)^{2-q}.$$

□

Proof of the Lower Bound: Proposition 2. Define

$$M = \varepsilon\sqrt{\varsigma\ln\left(R\varepsilon^{-1}\right)}, \quad J = \left\lfloor \frac{2-q}{2r}\ \log_2\left(RM^{-1}\right)\right\rfloor, \quad K = \left\lfloor q\log_2\left(RM^{-1}\right)\right\rfloor.$$

Set $p = 2^J$ and $d = 2^K$. Let us first check that M is well-defined and that $d \leq p$ under the assumptions of Proposition 2. Under the assumption $r < 1/q - 1/2$, we have $u > 0$, and since $R\varepsilon^{-1} \geq \mathrm{e}^2 \geq \mathrm{e}$, M is well-defined. Moreover, since $r < 1/q - 1/2$, we have $(2-q)/(2r) > q$, so it only remains to check that $RM^{-1} \geq \mathrm{e}$ to prove that $d \leq p$. We shall in fact prove the following stronger result:

Claim. If $R\varepsilon^{-1} \geq \max(\mathrm{e}^2, \varsigma^2)$, then $R\varepsilon^{-1}/\left(\ln(R\varepsilon^{-1})\right) \geq \varsigma$.

This result implies that, under the assumption $R\varepsilon^{-1} \geq \max(\mathrm{e}^2, \varsigma^2)$,

$$RM^{-1} = \frac{R\varepsilon^{-1}}{\sqrt{\varsigma\ln\left(R\varepsilon^{-1}\right)}} = \sqrt{R\varepsilon^{-1}}\sqrt{\frac{R\varepsilon^{-1}}{\varsigma\ln\left(R\varepsilon^{-1}\right)}} \geq \mathrm{e}\times 1 \geq \mathrm{e}.$$

Let us prove Claim 3.1. Introduce the function $g :]0,+\infty[\mapsto \mathbb{R},\ x \mapsto x/\ln(x)$. It is easy to check that $g(x^2) \geq x$ for all $x > 0$ and that g is non-decreasing on $[\mathrm{e},+\infty[$. Now, assume that $R\varepsilon^{-1} \geq \max(\mathrm{e}^2, \varsigma^2)$. Using the properties of g, we deduce that, if $\varsigma \geq \mathrm{e}$ then $R\varepsilon^{-1} \geq \varsigma^2 \geq \mathrm{e}^2 \geq \mathrm{e}$ and $R\varepsilon^{-1}/(\ln(R\varepsilon^{-1})) = g(R\varepsilon^{-1}) \geq g(\varsigma^2) \geq \varsigma$, while if $\varsigma < \mathrm{e}$ then $R\varepsilon^{-1} \geq \mathrm{e}^2 \geq \mathrm{e}$ and $R\varepsilon^{-1}/(\ln(R\varepsilon^{-1})) = g(R\varepsilon^{-1}) \geq g(\mathrm{e}^2) \geq \mathrm{e} > \varsigma$, hence Claim 3.1.

Now, consider the hypercube $A(p,d,M)$ defined by

$$\left\{\sum_{j=1}^{\infty}\alpha_j\,\phi_j;\ (\alpha_1,\ldots,\alpha_p)\in[0,M]^p,\ \alpha_j=0 \text{ for } j\geq p+1,\ \sum_{j=1}^{p}\mathbb{1}_{\{\alpha_j\neq 0\}}=d\right\}$$
$$=\left\{M\sum_{j=1}^{\infty}\beta_j\,\phi_j;\ (\beta_1,\ldots,\beta_p)\in[0,1]^p,\ \beta_j=0 \text{ for } j\geq p+1,\ \sum_{j=1}^{p}\mathbb{1}_{\{\beta_j\neq 0\}}=d\right\}.$$

The essence of the proof is just to check that $A(p,d,M)\subset\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)$. This shall enable us to bound from below the minimax risk over $\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)$ by the lower bound of the minimax risk over $A(p,d,M)$ provided by Birgé and Massart (2001).

Let $u\in A(p,d,M)$. Write $u=\sum_{j=1}^{\infty}\alpha_j\phi_j=M\sum_{j=1}^{\infty}\beta_j\phi_j$.

$$\sum_{j=1}^{\infty}|\alpha_j|^q=M^q\sum_{j=1}^{p}\beta_j^q\,\mathbb{1}_{\{\beta_j\neq 0\}}\leq M^q\sum_{j=1}^{p}\mathbb{1}_{\{\beta_j\neq 0\}}\leq M^q d\leq M^q\left(RM^{-1}\right)^q\leq R^q.$$

Thus, $u\in\mathcal{L}_q(R)$.

Let $J_0\in\mathbb{N}^\star$. If $J_0>p$, then

$$J_0^{2r}\sum_{j=J_0}^{\infty}\alpha_j^2\leq J_0^{2r}\sum_{j=p+1}^{\infty}\alpha_j^2=0\leq R^2.$$

If $J_0\leq p$, then

$$J_0^{2r}\sum_{j=J_0}^{\infty}\alpha_j^2=J_0^{2r}M^2\sum_{j=J_0}^{p}\beta_j^2\mathbb{1}_{\{\beta_j\neq 0\}}\leq J_0^{2r}M^2\sum_{j=J_0}^{p}\mathbb{1}_{\{\beta_j\neq 0\}}\leq p^{2r}M^2d\leq R^2.$$

Thus, $u\in\mathcal{B}^r_{2,\infty}(R)$.

Therefore, $A(p,d,M)\subset\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)$ and

$$\inf_{\tilde{s}}\sup_{s\in\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)}\mathbb{E}\left[\|s-\tilde{s}\|^2\right]\geq\inf_{\tilde{s}}\sup_{s\in A(p,d,M)}\mathbb{E}\left[\|s-\tilde{s}\|^2\right].\qquad(34)$$

Now, from Birgé and Massart (2001, Theorem 5), we know that the minimax risk over $A\left(p,d,M\right)$ satisfies

$$\inf_{\tilde{s}}\sup_{s\in A(p,d,M)}\mathbb{E}\left[\|s-\tilde{s}\|^2\right]\geq\kappa d\min\left\{M^2,\varepsilon^2\left(1+\ln\left(\frac{p}{d}\right)\right)\right\}$$
$$\geq\kappa\frac{\left(RM^{-1}\right)^q}{2}\min\left\{M^2,\varepsilon^2\left(1+\ln\left(\frac{p}{d}\right)\right)\right\}\qquad(35)$$

where $\kappa > 0$ denotes some absolute constant.
Moreover, we have

$$\begin{aligned}\varepsilon^2\left(1+\ln\left(\frac{p}{d}\right)\right) &\geq \varepsilon^2\left(1+\ln\left[\frac{\left(RM^{-1}\right)^{\frac{2-q}{2r}}}{2\left(RM^{-1}\right)^q}\right]\right)\\ &\geq \varepsilon^2\ln\left[\left(RM^{-1}\right)^{\varsigma}\right]\\ &= \varepsilon^2\ln\left[\left(R\varepsilon^{-1}\right)^{\varsigma}\left(\varepsilon M^{-1}\right)^{\varsigma}\right]\\ &= M^2+\varepsilon^2\ln\left[\left(\varepsilon M^{-1}\right)^{\varsigma}\right]\\ &= M^2-\frac{\varsigma}{2}\,\varepsilon^2\ln\left[\varsigma\ln\left(R\varepsilon^{-1}\right)\right]. \end{aligned} \tag{36}$$

But the assumption $R\varepsilon^{-1} \geq \max(\mathrm{e}^2, \varsigma^2)$ implies that (36) is greater than $M^2/2$. Indeed, first note that

$$M^2-\frac{\varsigma}{2}\,\varepsilon^2\ln\left[\varsigma\ln\left(R\varepsilon^{-1}\right)\right] \geq M^2/2 \;\Leftrightarrow\; \frac{R\varepsilon^{-1}}{\ln(R\varepsilon^{-1})} \geq \varsigma, \tag{37}$$

and then apply Claim 3.1. Thus, we deduce from (34), (35), (36) and (37) that there exists some $\kappa > 0$ such that

$$\inf_{\tilde{s}}\sup_{s\in\mathcal{L}_q(R)\cap\mathcal{B}^r_{2,\infty}(R)}\mathbb{E}\left[\|s-\tilde{s}\|^2\right] \geq \kappa R^q M^{2-q} = \kappa\varsigma^{1-\frac{q}{2}}R^q\left(\varepsilon\sqrt{\ln(R\varepsilon^{-1})}\right)^{2-q}.$$

□

3.2 Non-orthonormal Dictionaries

Sketch of the proof of Proposition 3.
Proposition 3 is deduced from the oracle inequality (12) in Theorem 1. First, the proof consists in bounding $\inf_{t\in\mathcal{L}_1(\mathcal{D}_p)}\left\{\|s-t\|^2+\lambda_p\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\}$ for all $p \in \mathbb{N}^\star$, just as it is done in Lemma 1 in the orthonormal case. This first step is very similar to Corollary 3.7 in Barron et al. (2008). Then, an additional step is needed to adapt the truncation of the dictionary according to the unknown parameters of smoothness q and r of the target function. This second step is similar to the proof of Proposition 1 in the orthonormal case. We refer the interested reader to Massart and Meynet (2010, proof of Proposition 5.7) for a detailed proof of Proposition 3. □

3.3 Interpolation Spaces

Proof of Remark 3.
Assume that the dictionary $\mathcal{D}$ is an orthonormal basis of the Hilbert space $\mathbb{H}$ and that there exist $1 < q < 2$, $r > 0$ and $R > 0$ such that $s \in w\mathcal{L}_q(R) \cap \mathcal{B}^r_{2,\infty}(R)$. For all $p \in \mathbb{N}^\star$ and $\lambda > 0$, define

$$s_{p,\lambda} := \underset{t\in\mathcal{L}_1(\mathcal{D}_p)}{\arg\min}\left\{\|s-t\|^2+\lambda\|t\|_{\mathcal{L}_1(\mathcal{D}_p)}\right\}.$$

The proof is divided in two main parts. First, we choose λ such that $s_{p,\lambda} \in \mathcal{L}_{1,r}$. Secondly, we choose p such that $\|s - s_{p,\lambda}\| + \delta \|s_{p,\lambda}\|_{\mathcal{L}_{1,r}} \leq C_{q,r} R\, \delta^{2\nu}$ for all $\delta > 0$, some $C_{q,r} > 0$ and $\nu = 1/q - 1/2$, which means that $s \in \mathcal{B}_{q,r}(C_{q,r} R)$.

Let us first choose λ such that $s_{p,\lambda} \in \mathcal{L}_{1,r}$. From Lemma 1, we have

$$\|s - s_{p,\lambda}\| \leq \sqrt{C_r R^2 p^{-2r} + C_q R^q \lambda^{2-q}} \leq \sqrt{C_r}\, R p^{-r} + \sqrt{C_q}\, R^{q/2} \lambda^{1-q/2}.$$

Now choose λ such that $\sqrt{C_r}\, R p^{-r} = \sqrt{C_q}\, R^{q/2} \lambda^{1-q/2}$, that is to say

$$\lambda_p := R \left(\sqrt{C_q C_r^{-1}}\, p^r \right)^{-\frac{2}{2-q}}. \tag{38}$$

Then, we have

$$\|s - s_{p,\lambda_p}\| \leq 2\sqrt{C_r}\, R p^{-r}. \tag{39}$$

Let us now check that $s_{p,\lambda_p} \in \mathcal{L}_{1,r}$. Define

$$C_p := \max \left\{ 4\sqrt{C_r}\, R, \max_{p' \in \mathbb{N}^\star,\ p' \leq p} \|s_{p',\lambda_{p'}}\|_{\mathcal{L}_1(\mathcal{D}_{p'})} \right\}. \tag{40}$$

Let $p' \in \mathbb{N}^\star$. By definition of $s_{p',\lambda_{p'}}$, we have $s_{p',\lambda_{p'}} \in \mathcal{L}_1(\mathcal{D}_{p'})$. If $p' \leq p$, then we deduce from (39) and (40) that

$$\|s_{p,\lambda_p} - s_{p',\lambda_{p'}}\| \leq \|s_{p,\lambda_p} - s\| + \|s - s_{p',\lambda_{p'}}\| \leq 2\sqrt{C_r}\, R \left(p^{-r} + p'^{-r} \right) \leq C_p p'^{-r},$$

and we have $\|s_{p',\lambda_{p'}}\|_{\mathcal{L}_1(\mathcal{D}_{p'})} \leq C_p$ by definition of C_p. If $p' > p$, then $\mathcal{L}_1(\mathcal{D}_p) \subset \mathcal{L}_1(\mathcal{D}_{p'})$ and $s_{p,\lambda_p} \in \mathcal{L}_1(\mathcal{D}_{p'})$ with $\|s_{p,\lambda_p}\|_{\mathcal{L}_1(\mathcal{D}_{p'})} \leq \|s_{p,\lambda_p}\|_{\mathcal{L}_1(\mathcal{D}_p)} \leq C_p$ and $\|s_{p,\lambda_p} - s_{p,\lambda_p}\| = 0 \leq C_p p'^{-r}$. So, $s_{p,\lambda_p} \in \mathcal{L}_{1,r}$.

Now, it only remains to choose a convenient $p \in \mathbb{N}^\star$ to prove that $s \in \mathcal{B}_{q,r}(C_{q,r} R)$ for some $C_{q,r}$.

Let us first give an upper bound of $\|s_{p,\lambda_p}\|_{\mathcal{L}_{1,r}}$ for all $p \in \mathbb{N}^\star$. By definition of $\|s_{p,\lambda_p}\|_{\mathcal{L}_{1,r}}$ and the above upper bounds, we have $\|s_{p,\lambda_p}\|_{\mathcal{L}_{1,r}} \leq C_p$. So, we just have to bound C_p. Let $p' \in \mathbb{N}^\star, p' \leq p$. From Lemma 1, there exists $C_q > 0$ depending only on q such that $\|s_{p',\lambda_{p'}}\|_{\mathcal{L}_1(\mathcal{D}_{p'})} \leq C_q R^q \lambda_{p'}^{1-q}$. So, we get from (38) that

$$\begin{aligned} \|s_{p',\lambda_{p'}}\|_{\mathcal{L}_1(\mathcal{D}_{p'})} &\leq C_q R \left(\sqrt{C_q C_r^{-1}}\, p'^r \right)^{\frac{2(q-1)}{2-q}} \\ &\leq C_q R \left(\sqrt{C_q C_r^{-1}} p^r \right)^{\frac{2(q-1)}{2-q}} \\ &= C_q^{\frac{1}{2-q}} C_r^{-\frac{(q-1)}{2-q}} R p^{\frac{2(q-1)r}{2-q}}, \end{aligned}$$

and we deduce from (40) that

$$C_p \leq \max\left\{4\sqrt{C_r}\,R, C_q^{\frac{1}{2-q}} C_r^{-\frac{(q-1)}{2-q}} Rp^{\frac{2(q-1)r}{2-q}}\right\} \leq C_{q,r} Rp^{\frac{2(q-1)r}{2-q}}$$

where $C_{q,r} > 0$ depends only on q and r. Thus, we have

$$\|s_{p,\lambda_p}\|_{\mathcal{L}_{1,r}} \leq C_{q,r} Rp^{\frac{2(q-1)r}{2-q}}. \tag{41}$$

Then, we deduce from (39) and (41) that for all $p \in \mathbb{N}^\star$ and $\delta > 0$,

$$\begin{aligned} \inf_{t \in \mathcal{L}_{1,r}} \left\{\|s-t\| + \delta\|t\|_{\mathcal{L}_{1,r}}\right\} &\leq \|s - s_{p,\lambda_p}\| + \delta\|s_{p,\lambda_p}\|_{\mathcal{L}_{1,r}} \\ &\leq 2\sqrt{C_r}\,Rp^{-r} + \delta C_{q,r} Rp^{\frac{2(q-1)r}{2-q}}. \end{aligned} \tag{42}$$

Now, we choose p such that p^{-r} and $\delta\, p^{\frac{2(q-1)r}{2-q}}$ are of the same order. More precisely, set $p = 2^J$ where $J = \left\lceil (2-q)(qr)^{-1}\log_2(\delta^{-1})\right\rceil$. With this value of p, we get that there exists $C'_{q,r} > 0$ depending only on q and r such that (42) is upper bounded by $C'_{q,r} R\,\delta^{(2-q)/q} = C'_{q,r} R\,\delta^{2\nu}$. This means that $s \in \mathcal{B}_{q,r}(C'_{q,r}R)$, hence (20). □

References

Barron, A., Cohen, A., Dahmen, W., DeVore, R.: Approximation and learning by greedy algorithms. Annals of Statistics 36(1), 64–94 (2008)

Bartlett, P., Mendelson, S., Neeman, J.: ℓ_1-regularized linear regression: persistence and oracle inequalities. Probability Theory and Related Fields (2012)

Bickel, P., Ritov, Y., Tsybakov, A.: Simultaneous analysis of Lasso and Dantzig selector. Annals of Statistics 37(4), 1705–1732 (2009)

Birgé, L., Massart, P.: Gaussian model selection. Journal of the European Mathematical Society 3(3), 203–268 (2001)

Huang, C., Cheang, G., Barron, A.: Risk of penalized least squares, greedy selection and ℓ_1-penalization for flexible function libraries. Submitted to the Annals of Statistics (2008)

Koltchinskii, V.: Sparsity in penalized empirical risk minimization. The Annals of Statistics 45(1), 7–57 (2009)

Massart, P., Meynet, C.: An ℓ_1-oracle inequality for the Lasso. ArXiv 1007.4791 (2010)

Massart, P., Meynet, C.: The Lasso as an ℓ_1-ball model selection procedure. Electronic Journal of Statistics 5, 669–687 (2011)

Rigollet, P., Tsybakov, A.: Exponential Screening and optimal rates of sparse estimation. The Annals of Statistics 39(2), 731–771 (2011)

Rivoirard, V.: Nonlinear estimation over weak Besov spaces and minimax Bayes method. Bernoulli 12(4), 609–632 (2006)

Tibshirani, R.: Regression shrinkage and selection via the Lasso. Journal of the Royal Statistical Society. Series B 58, 267–288 (1996)

van de Geer, S.: High dimensional generalized linear models and the Lasso. The Annals of Statistics 36(2), 614–645 (2008)

Recent Developments in Pattern Mining

Toon Calders

TU Eindhoven, The Netherlands
t.calders@tue.nl

Abstract. Pattern Mining is one of the most researched topics in the data mining community. Literally hundreds of algorithms for efficiently enumerating all frequent itemsets have been proposed. These exhaustive algorithms, however, all suffer from the pattern explosion problem. Depending on the minimal support threshold, even for moderately sized databases, millions of patterns may be generated. Although this problem is by now well recognized in te pattern mining community, it has not yet been solved satisfactorily. In my talk I will give an overview of the different approaches that have been proposed to alleviate this problem. As a first step, constraint-based mining and condensed representations such as the closed itemsets and the non-derivable itemsets were introduced. These methods, however, still produce too many and redundant results. More recently, promising methods based upon the minimal description length principle, information theory, and statistical models have been introduced. We show the respective advantages and disadvantages of these approaches and their connections, and illustrate their usefulness on real life data. After this overview we move from itemsets to more complex patterns, such as sequences and graphs. Even though these extensions seem trivial at first, they turn out to be quite challenging. I will end my talk with an overview of what I consider to be important open questions in this fascinating research area.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, p. 34, 2012.

Exploring Sequential Data[*]

Gilbert Ritschard

NCCR LIVES and Institute for Demographic and Life Course Studies,
University of Geneva, CH-1211 Geneva 4, Switzerland
gilbert.ritschard@unige.ch

Abstract. The tutorial is devoted to categorical sequence data describing for instance the successive buys of customers, working states of devices, visited web pages, or professional careers. Addressed topics include the rendering of state and event sequences, longitudinal characteristics of sequences, measuring pairwise dissimilarities and dissimilarity-based analysis of sequence data such as clustering, representative sequences, and regression trees. The tutorial also provides a short introduction to the practice of sequence analysis with the TraMineR R-package.

[*] The full version of this paper is published in the Proceedings of the 15th International Conference on Discovery Science, Lecture Notes in Artificial Intelligence Vol. 7569.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, p. 35, 2012.

Enlarging Learnable Classes

Sanjay Jain[1,⋆], Timo Kötzing[2,⋆⋆], and Frank Stephan[3,⋆⋆⋆]

[1] Department of Computer Science, National University of Singapore, Singapore 117417, Republic of Singapore
sanjay@comp.nus.edu.sg

[2] Max-Planck-Institut für Informatik, Campus E 1 4, 66123 Saarbrücken, Germany
koetzing@mpi-inf.mpg.de

[3] Department of Mathematics, National University of Singapore, Singapore 119076, Republic of Singapore
fstephan@comp.nus.edu.sg

Abstract. An early result in inductive inference shows that the class of **Ex**-learnable sets is *not* closed under unions. In this paper we are interested in the following question: For what classes of functions is the union with an arbitrary **Ex**-learnable class again **Ex**-learnable, either effectively (in an index for a learner of an **Ex**-learnable class) or non-effectively? We show that the effective case and the non-effective case separate, and we give a sufficient criterion for the effective case. Furthermore, we extend our notions to considering unions with classes of single functions, as well as to other learning criteria, such as finite learning and behaviorally correct learning.

Furthermore, we consider the possibility of (effectively) extending learners to learn (infinitely) more functions. It is known that all **Ex**-learners learning a *dense* set of functions can be effectively extended to learn infinitely more. It was open whether the learners learning a *non-dense* set of functions can be similarly extended. We show that this is *not* possible, but we give an alternative split of all possible learners into two sets such that, for each of the sets, all learners from that set can be effectively extended. We analyze similar concepts also for other learning criteria.

1 Introduction

One branch of inductive inference investigates the learnability of functions; the basic scenario given in the seminal paper by Gold [7] is as follows. Let $\mathcal{S}$ be a class of recursive functions; we say that $\mathcal{S}$ is *explanatorily learnable* iff there is a learner M which issues conjectures $e_0, e_1, \ldots$ with e_n being based on the data $f(0)f(1)\ldots f(n-1)$ such that, for all $f \in \mathcal{S}$, almost all of these conjectures are the same index e explaining f, that is, satisfying $\varphi_e = f$ with respect to an

⋆ Supported by NUS grants C252-000-087-001 and R252-000-420-112.

⋆⋆ Major parts of this paper were written when Timo Kötzing was visiting the Department of Computer Science at the National University of Singapore.

⋆⋆⋆ Supported in part by NUS grant R252-000-420-112.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 36–50, 2012.

underlying numbering $\varphi_0, \varphi_1, \ldots$ of all partial recursive functions. In this paper, we consider learnability by partial recursive learners; with M_e we refer to the learner derived from the e-th partial recursive function.

During the course of time, several variants of this basic notion of explanatory learning (**Ex**) have been considered; most notably, *behaviorally correct learning* (**BC**) [1], in which the learner has to almost always output a correct index for the input function (these indices though are not constrained to be the same).

Another variant considered is *finite learning* (**Fin**) [7] where the learner outputs a special symbol (?) until it makes one conjecture e which is never abandoned; this conjecture must of course be correct for a function to be learnt. Osherson, Stob and Weinstein [10] introduced a generalization of this notion, namely *confident learning* (**Conf**), where the learner can revise the hypothesis finitely often; it must, however, on each function f, even if it is not in the class to be learnt, eventually stabilize on one conjecture e. In inductive inference, one often only needs the weak version of this property where the convergence criterion only applies to recursive functions while the convergence behavior on non-recursive ones is not constrained (**WConf**, [14]).

Minicozzi [9] called a learner *reliable* iff the learner, on every function, either converges to a correct index or signals infinitely often that it does not find the index (by doing a mind change or outputting a question mark). One can combine the notion of reliability and confidence: A learner is *weakly reliable and confident* (**WConfRel**) iff the learner, for every recursive function f, either converges to an index e with $\varphi_e = f$ or almost always outputs ? (in order to signal non-convergence).

The above crtieria and the relations between them have been extensively studied, giving the following inclusion relations [2, 5, 6, 7, 9, 10, 14]:

- **Fin** $\subset$ **Conf** $\subset$ **WConf** $\subset$ **Ex** $\subset$ **BC**;
- **ConfRel** $\subset$ **WConfRel** $\subset$ **Rel** $\subset$ **Ex** $\subset$ **BC**;
- **Fin** $\not\subseteq$ **Rel** and **Rel** $\not\subseteq$ **WConf**.

Besides inclusion (learnability with respect to which criterion implies learnability with respect to another criterion), structural questions have also been studied: Is the union of two learnable classes learnable? Can one extend each learnable class?

Blum and Blum's *Non-Union Theorem* [2] (see also [1]) gave a quite strong answer to the first question: There are two classes $\mathcal{S}$ and $\mathcal{S}'$ of recursive functions such that each of them is learnable under the criterion **Ex** but their union is not learnable even under the more general criterion **BC**. Indeed, one can even learn the class $\mathcal{S}$ confidently and the class $\mathcal{S}'$ reliably. Thus, the Non-Union Theorem gives an interesting contrast to the fact that both confident learning and reliable learning are effectively closed under union.

Furthermore, it is interesting to ask how effective the union is. That is, if the union of two classes is learnable, can one effectively construct a learner for the union, given programs for the learners of the two given classes? The answer is "No" in general as can be seen directly by the proof of the Non-Union Theorem.

The confidently learnable class $\mathcal{S}$ above consists of all the functions f such that $f(0)$ is an index for f, and the class $\mathcal{S}'$ consists of all the functions f which are almost everywhere 0 (Blum and Blum [2] used slightly different classes $\mathcal{S}$ and $\mathcal{S}'$ which were $\{0,1\}$-valued; our $\mathcal{S}$ and $\mathcal{S}'$ makes the presentation simpler). Now consider the union of $\mathcal{S}'$ with a class $\mathcal{S}_e$, where $\mathcal{S}_e$ contains φ_e in the case that φ_e is total and $\varphi_e(0) = e$; otherwise $\mathcal{S}_e$ is empty. It is easy to show that, for each e, the class $\mathcal{S}_e \cup \mathcal{S}'$ is explanatory (**Ex**) learnable. If this union would be effective, giving rise to a learner $M_{h(e)}$ for the class $\mathcal{S}_e \cup \mathcal{S}'$, then one could make a learner N for $\mathcal{S} \cup \mathcal{S}'$ as follows: for non-empty sequences σ, $N(\sigma) = M_{h(\sigma(0))}(\sigma)$; a contradiction to the non-union theorem.

This example suggests to study four notions of when the unions of a given class $\mathcal{S}$ with another class is **Ex**-learnable:

1. $\mathcal{S}$ is (non-constructively) **Ex**-*unionable* iff for every **Ex**-learnable class $\mathcal{S}'$, the class $\mathcal{S} \cup \mathcal{S}'$ is **Ex**-learnable;
2. $\mathcal{S}$ is *constructively* **Ex**-*unionable* iff one can effectively convert every **Ex**-learner for a class $\mathcal{S}'$ into an **Ex**-learner for the class $\mathcal{S} \cup \mathcal{S}'$;
3. $\mathcal{S}$ is *singleton*-**Ex**-*unionable* iff for every total computable g, $\mathcal{S} \cup \{g\}$ is **Ex**-learnable.
4. $\mathcal{S}$ is *constructively singleton*-**Ex**-*unionable* iff there is a recursive function which assigns, to every index e, an **Ex**-learner for the class $\mathcal{S} \cup \{\varphi_e\}$ if φ_e is total and for the class $\mathcal{S}$ if φ_e is partial.

The same notions can also be defined for other learning criteria like finite, confident and behaviorally correct learning. We get the following results:

1. If a class $\mathcal{S}$ has a weakly confident learner then it is constructively singleton-**Ex**-unionable.
2. If a class $\mathcal{S}$ has a weakly confident and reliable learner then it is constructively **Ex**-unionable.
3. There is a class which is **Ex**-unionable and **BC**-unionable but does not satisfy any of the constructive unionability properties.
4. For finite learning, we show that unionability with classes and constructive union with singletons fails for all non-empty classes; only non-constructive unions with singletons is possible in the case that every pointwise limit of functions in the class is again in the class.

All our results for the cases of purely **Ex**-learning are summarized in Figure 1.

Forming the union with another class or adding a function are specific methods to enlarge a class. Thus, it is natural to ask when a learnable class of functions can be extended at all, without prescribing how to do this. Case and Fulk [4] addressed this question and showed, for the principal learning criteria **Ex** and **BC**, that one can extend learners to learn infinitely more functions whenever the learner satisfies a certain quality, say learns a dense class of functions. This enlargement can be done constructively (under this precondition). Furthermore, one can non-constructively extend any learnable class for many usual learning criteria like **Fin**, **Conf**, **Rel**, **ConfRel**, **WConf**, **WConfRel**, **Ex** and **BC**. Case and Fulk [4] left open two particular questions:

1. Is there a method to extend constructively every learner M_e which does not **Ex**-learn a dense class of functions?

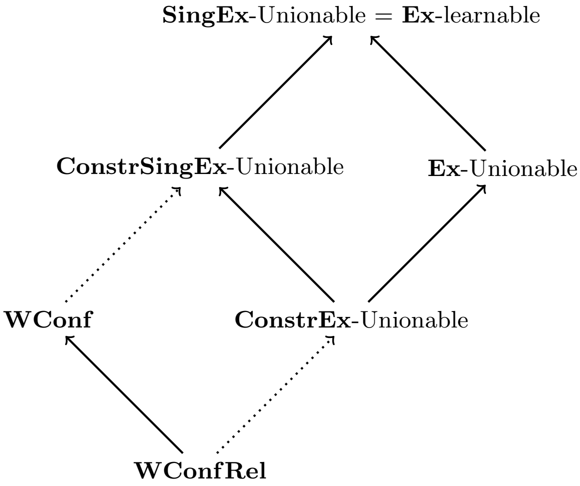

SingEx-Unionable = **Ex**-learnable
ConstrSingEx-Unionable
Ex-Unionable
WConf
ConstrEx-Unionable
WConfRel

Fig. 1. The inclusion relations for the various unionability notions. It is unknown whether the dotted arrows might also go in the converse direction. All inclusions are given by arrows (and possibly reversed dotted arrows) and the concatenations of these.

2. How much nonconstructive information is needed in order to extend every learner M_e to learn infinitely many more functions? I.e., in how many classes does one have to split the learners so as to have constructive extension for each of the classes?

Theorem 25 answers the first question negatively – such a method does not exist.

On the other hand, the answer to the second question is that only a split into two classes is necessary. This result is not based on the information about whether the class is dense or not; instead it is based on the information about whether there exists a σ such that for no extension τ of σ: $M(\tau)\downarrow \neq M(\sigma)\downarrow$. In Theorem 27 we show that there is a recursive function h such that $\mathbf{Ex}(M_{h(e,b)})$ is a proper superclass of $\mathbf{Ex}(M_e)$ whenever either $b = 1$ and such a σ exists or $b = 0$ and such a σ does not exist.

2 Preliminaries

Let $\mathbb{N}$ denote the set of natural numbers. The symbols $\subseteq, \subset, \supseteq, \supset$ respectively denote subset, proper subset, superset and proper superset. For strings α and β, we let $\alpha \preceq \beta$ denote that α is a prefix of β. We let $\langle \cdot, \cdot \rangle$ denote a fixed computable pairing function from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$, which is increasing in both its arguments. We assume that $\langle 0, 0 \rangle = 0$.

Let φ denote a fixed acceptable programming system [12] for the class of all partial recursive functions. Let φ_i denote the i-th program in this programming system. Then, i is called the index or program for the partial recursive function φ_i. Let $\mathcal{R}$ denote the set of all total recursive functions and $\mathcal{P}$ denote the set of all partial recursive functions. Let $\mathcal{R}_{0,1}$ denote the set of all total recursive functions f with range$(f) \subseteq \{0,1\}$. Let K denote the diagonal halting set $\{x : \varphi_x(x)\downarrow\}$. For a function η, let $\eta(x)\downarrow$ denote that $\eta(x)$ is defined, and $\eta(x)\uparrow$ denote that $\eta(x)$ is not defined. We let pad be a 1–1 recursive function such that, for all i, j, $\varphi_{\text{pad}(i,j)} = \varphi_i$. Please find unexplained recursion theoretic notions in Rogers' book [12]. We let $\mathcal{S}$ range over sets of recursive functions.

Let σ, τ range over finite sequences. We often identify a total function with its sequence of values, $f(0)f(1)f(2)\ldots$; similarly for finite sequences. Let $f[n] = f(0)f(1)\ldots f(n-1)$. We use the notation $\sigma \preceq \tau$ to denote that σ is a prefix of τ (an initial subfunction of τ). Let $\varLambda$ denote the empty sequence. Let $|\sigma|$ denote the length of σ. Let $\mathbb{S}$eq denote the set of all finite sequences.

Let $\sigma \cdot \tau$ denote concatenation of sequences, where σ is finite. When it is clear from context, we often drop $\cdot$ and just use $\sigma\tau$ for concatenation. For a finite sequence $\sigma \neq \varLambda$, let σ^- be σ with the last element dropped, that is, $\sigma^- \cdot \sigma(|\sigma|) = \sigma$. Let $[\mathcal{S}] = \{f[n] \mid f \in \mathcal{S}\}$. Thus, $[\mathcal{R}] = \mathbb{S}$eq. For notation simplification, $[f] = [\{f\}]$. A class $\mathcal{S}$ is said to be *dense* if $[\mathcal{S}] = [\mathcal{R}]$. A class $\mathcal{S}$ is *everywhere sparse* iff for all $\tau \in \mathbb{S}$eq, there exists a $\tau' \succeq \tau$ such that $\tau' \notin [\mathcal{S}]$. A total function f is an *accumulation point* of $\mathcal{S}$ iff there exist pairwise distinct functions $g_0, g_1, \ldots$ in $\mathcal{S}$ such that, for all $n \in \mathbb{N}$, $f[n] \preceq g_n$.

A *learner* is a partial-recursive mapping from finite sequences to $\mathbb{N} \cup \{?\}$. We let M, N and P range over learners and let $\mathcal{C}$ range over classes of learners. Let $M_0, M_1, \ldots$ denote an acceptable numbering of all the learners.

We say that M converges on function f to i (written: $M(f)\downarrow = i$) iff for all but finitely many n, $M(f[n]) = i$. If $M(f)\downarrow = i$ for some $i \in \mathbb{N}$, then we say that M converges on f (written: $M(f)\downarrow$). We say that $M(f)$ diverges (written: $M(f)\uparrow$) if $M(f)$ does not converge to any $i \in \mathbb{N}$. We now describe some of the learning criteria.

Definition 1. Suppose M is a learner and f is a total function.

(a) [7] We say that M **Ex**-*learns* f (written: $f \in \mathbf{Ex}(M)$) iff (i) for all s, $M(f[s])$ is defined, and (ii) there exists an i such that $\varphi_i = f$ and, for all but finitely many n, $M(f[n]) = i$.
(b) [1, 6] We say that M **BC**-*learns* f (written: $f \in \mathbf{BC}(M)$) iff, (i) for all s, $M(f[s])$ is defined, and (ii) for all but finitely many n, $\varphi_{M(f[n])} = f$.
(c) [1, 6] We say that M **Fin**-*learns* f (written: $f \in \mathbf{Fin}(M)$) iff (i) for all s, $M(f[s])$ is defined, and (ii) there exist n and i such that $\varphi_i = f$, for all $m < n$, $M(f[n]) =?$, and for all $m \geq n$, $M(f[n]) = i$.
(d) [6] We say that M $\mathbf{Ex}_n$-*learns* f (written: $f \in \mathbf{Ex}_n(M)$) iff (i) M **Ex**-learns f and (ii) card$(\{m \mid ? \neq M(f[m]) \neq M(f[m+1])\}) \leq n$.

We say that M makes a *mind change* at $f[m+1]$ if $? \neq M(f[m]) \neq M(f[m+1])$.

Definition 2. Let I be a learning criterion (defined above or later in this paper):

(a) We say that M I-learns $\mathcal{S}$ (written: $\mathcal{S} \subseteq I(M)$) iff M I-learns each $f \in \mathcal{S}$.
(b) We say that $\mathcal{S}$ is I-learnable iff there exists a learner M which I-learns $\mathcal{S}$.
(c) $I = \{\mathcal{S} \mid \exists M\, [\mathcal{S} \subseteq I(M)]\}$.

Definition 3. (a) [10] We say that M is *confident* iff (i) M is total and (ii) for all total f, $M(f)\downarrow$ or for all but finitely many n, $M(f[n]) =?$.

(b) We say that M is *weakly confident* iff (i) M is total and (ii) for all $f \in \mathcal{R}$, $M(f)\downarrow$ or for all but finitely many n, $M(f[n]) =?$.

(c) [2, 9] We say that M is *reliable* iff (i) M is total and (ii) for all total f, $M(f)\downarrow$ implies M **Ex**-learns f.

(d) We say that M is *weakly reliable* iff (i) M is total and (ii) for all $f \in \mathcal{R}$, $M(f)\downarrow$ implies M **Ex**-learns f.

(e) We say that M is *confident and reliable* iff M is total and, either M **Ex**-learns f or $M(f[n]) =?$ for all but finitely many n.

(f) We say that M is *weakly confident and reliable* iff M is total and, for all $f \in \mathcal{R}$, either M **Ex**-learns f or $M(f[n]) =?$ for all but finitely many n.

Definition 4. We say that M **Conf**-learns $\mathcal{S}$ if M **Ex**-learns $\mathcal{S}$ and M is confident. Similarly, we define **Rel**, **WConf**, **WRel**, **ConfRel** and **WConfRel** learning criteria where we require the learners to be reliable, weakly confident, weakly reliable, confident and reliable, and weakly confident and reliable respectively.

For all the learning criteria considered in this paper, one can assume without loss of generality that the learners are total. In particular, from any learner M, one can effectively construct a total learner M' such that, for all the learning criteria I considered in this paper, $I(M) \subseteq I(M')$ (this can be shown essentially using the same proof as for $I = \mathbf{Ex}$ used by [10]). We often implicitly assume such conversion of learners into total learners. The following proposition shows that learners for unions of confidently learnable classes can be effectively found; similarly for learners of unions of reliably learnable classes.

Proposition 5 (Blum and Blum [2], Minicozzi [9], Osherson, Stob and Weinstein [10]). Each criterion I from **Conf**, **WConf**, **Rel**, **WRel**, **ConfRel**, **WConfRel** is closed effectively under union: there exists a recursive function h_I such that, if M_i I-learns $\mathcal{S}$ and M_j I-learns $\mathcal{S}'$ then $M_{h_I(i,j)}$ I-learns $\mathcal{S} \cup \mathcal{S}'$.

Definition 6. [13] A set $\mathcal{S} \subseteq \mathcal{R}$ is *two-sided classifiable* iff there is a machine M such that, for all $f \in \mathcal{R}$,

(i) if $f \in \mathcal{S}$, then $\forall^\infty x\,[M(f[x]) = 1]$;
(ii) if $f \notin \mathcal{S}$, then $\forall^\infty x\,[M(f[x]) = 0]$.

The next theorem characterizes **WConfRel** in terms of classification.

Theorem 7. Let $\mathcal{S} \subseteq \mathcal{R}$. The following are equivalent:

(a) $\mathcal{S}$ is **WConfRel**-learnable;
(b) A superset of $\mathcal{S}$ is **Ex**-learnable and two-sided classifiable.

3 Initial Results on Unionability

We start with giving the general definition of unionability.

Definition 8. Let I be a learning criterion and $\mathcal{S} \subset \mathcal{R}$.

(a) $\mathcal{S}$ is *I-unionable* iff, for all I-learnable classes $\mathcal{S}'$, $\mathcal{S} \cup \mathcal{S}'$ is I-learnable.
(b) $\mathcal{S}$ is *constructively I-unionable* iff there is an $h \in \mathcal{R}$ such that, for all e, $\mathcal{S} \cup I(M_e) \subseteq I(M_{h(e)})$.
(c) $\mathcal{S}$ is *singleton-I-unionable* iff, for all $f \in \mathcal{R}$, $\mathcal{S} \cup \{f\}$ is I-learnable.
(d) $\mathcal{S}$ is *constructively singleton-I-unionable* iff there is $h \in \mathcal{R}$ such that, for all e, $M_{h(e)}$ I-learns $\mathcal{S} \cup \{\varphi_e\} \cap \mathcal{R}$.

For the various versions of unionability, in the following sections we will consider in detail which classes are I-unionable for I being **Fin**, **Ex** or **BC**, starting with **Fin**-unionability in this section.

Theorem 9 (Blum and Blum [2]). There are classes $\mathcal{S}$ and $\mathcal{S}'$ such that

(a) $\mathcal{S}$ is **Fin**-learnable (and thus $\mathcal{S} \in$ **Conf** and $\mathcal{S} \in$ **WConf**);
(b) $\mathcal{S}'$ is **Rel**-learnable;
(c) $\mathcal{S} \cup \mathcal{S}' \notin$ **BC**.

Thus, both classes $\mathcal{S}$ and $\mathcal{S}'$ are neither **Ex**-unionable nor **BC**-unionable. In the following, we want to characterise **Fin**-unionability.

Theorem 10. (a) $\mathcal{S}$ is **Fin**-unionable iff $\mathcal{S} = \emptyset$.
(b) $\mathcal{S}$ is constructively **Fin**-unionable iff $\mathcal{S} = \emptyset$.
(c) $\mathcal{S}$ is constructively singleton-**Fin**-unionable iff $\mathcal{S} = \emptyset$.
(d) $\mathcal{S}$ is singleton-**Fin**-unionable iff $\mathcal{S}$ is **Fin**-learnable and $\mathcal{S}$ has no recursive accumulation point.

Proof. (a) and (b) Let $\mathcal{S} \neq \emptyset$ be a set of total computable functions and let $f \in \mathcal{S}$. For all i, let f_i be such that $f_i(i) = f(i)+1$ and, for all $x \neq i$, $f_i(x) = f(x)$. Then the class $\mathcal{S}' = \{f_i \mid i \in \mathbb{N}\}$ is **Fin**-learnable, but $\mathcal{S} \cup \mathcal{S}'$ is not **Fin**-learnable.

(c) We keep $\mathcal{S}$ and f and f_i as in part (a and b) above. Furthermore, we consider a recursive function g such that $\varphi_{g(e)} = f_s$, if e is enumerated into K in exactly s steps; $\varphi_{g(e)} = f$, if e is not enumerated into K. Furthermore, let h be a recursive function such that $M_{h(e)}$ **Fin**-learns $\mathcal{S} \cup \{\varphi_e\} \cap \mathcal{R}$. Let $k(e)$ be the first number found, in some algorithmic search, such that $M_{h(e)}(f[k(e)])\downarrow \neq ?$. The function k is total recursive, as, for all e, $M_{h(e)}$ **Fin**-learns f. If e is enumerated into K in exactly s steps, then $k(g(e)) \geq s$, as otherwise, $\varphi_{g(e)}[k(g(e))] = f_s[k(g(e))] = f[k(g(e))]$, and thus $M_{h(g(e))}$ cannot **Fin**-learn both f and $\varphi_{g(e)}$. Hence e is in K iff e is enumerated within $k(g(e))$ steps into K, a contradiction to K being undecidable.

(d) Clearly $\mathcal{S}$ must be in **Fin** to be singleton-**Fin**-unionable.

We first show that **Fin**-learnable classes with a recursive accumulation point are not singleton-**Fin**-unionable. Let $\mathcal{S}$ be such that there is a recursive accumulation point f of $\mathcal{S}$. Suppose $\mathcal{S} \cup \{f\}$ is **Fin**-learnable, as witnessed by M.

Let x be such that $M(f[x])\downarrow\neq ?$. Furthermore, let $f' \in \mathcal{S}$, $f \neq f'$ be such that $f[x] \preceq f'$. Such an f' exists as f is an accumulation point of $\mathcal{S}$. Now M cannot **Fin**-learn both f and f', as $f[x] \preceq f$ and $f[x] \preceq f'$. This is a contradiction to M **Fin**-learning $\mathcal{S} \cup \{f\}$.

Now suppose $\mathcal{S}$ is **Fin**-learnable as witnessed by M and $\mathcal{S}$ has no recursive accumulation point. Let $f \in \mathcal{R}$. We show that $\mathcal{S}_0 \cup \{f\}$ is **Fin**-learnable. If $f \in \mathcal{S}$, nothing is left to be shown. Suppose $f \notin \mathcal{S}$; thus, there exists an x such that $f[x] \notin [\mathcal{S}]$. Let e be an index for f; we define N such that, for all σ,

$$N(\sigma) = \begin{cases} ?, & \text{if } \sigma \prec f[x]; \\ e, & \text{if } f[x] \preceq \sigma; \\ M(\sigma), & \text{otherwise.} \end{cases}$$

It is easy to verify that N **Fin**-learns $\mathcal{S} \cup \{f\}$. □

It is clear that every constructively I-unionable class is I-unionable and every constructively singleton-I-unionable class is singleton-I-unionable. The next proposition gives the third straight-forward inclusion.

Proposition 11. Let $I \in \{\mathbf{Fin}, \mathbf{Conf}, \mathbf{WConf}, \mathbf{Ex}, \mathbf{BC}\}$. If $\mathcal{S}$ is constructively I-unionable then $\mathcal{S}$ is constructively singleton-I-unionable.

Proof. Given e, consider the I-learner $M_{h(e)}$ which always outputs e; if φ_e is total, then $I(M_{h(e)}) = \{\varphi_e\}$, else $I(M_{h(e)}) = \emptyset$. Now, due to the constructive I-unionability of $\mathcal{S}$, the class is also constructively singleton-I-unionable by forming constructively the union with the class I-learnt by $M_{h(e)}$. □

For the criteria **Rel**, **WRel**, **ConfRel** and **WConfRel**, one cannot translate an index e into a learner for φ_e of the given type, as one is not able to test in the limit whether φ_e is partial or total. This obstacle on the way to prove a hypothetical implication like "constructively **Rel**-unionable ⇒ constructively singleton-**Rel**-unionable" is real and the conjectured implication does not hold: On the one hand, every **Rel**-learnable class is constructively **Rel**-unionable [9]; on the other hand, Theorem 17 as well as Blum and Blum's Non-Union-Theorem exhibit a **Rel**-learnable class which is not constructively singleton-**Rel**-unionable.

4 Ex- and BC-Unionable Classes

Case and Fulk [4] investigated **Ex**- and **BC**-unionability and obtained the following basic result that one can always add a function to a given class; so in contrast to finite learning, every **Ex**-learnable class is non-constructively singleton-**Ex**-unionable; the same applies to **BC**-learning.

Proposition 12 (Case and Fulk [4]). If I is either **Ex** or **BC**, $f \in \mathcal{R}$ and $\mathcal{S}$ is I-learnable, then $\mathcal{S} \cup \{f\}$ is I-learnable.

Theorem 13. Suppose I is either **Ex** or **BC**. Suppose $\mathcal{S} \in \mathbf{WConfRel}$. Then $\mathcal{S}$ is constructively I-unionable.

Proof. Suppose $\mathcal{S} \in \mathbf{WConfRel}$ as witnessed by $M \in \mathcal{R}$. Let h be a recursive function such that $M_{h(i)}$ behaves as follows.

Let M_i' be obtained effectively from i such that M_i' is total and $I(M_i') = I(M_i)$. If $M(\sigma) = ?$, then $M_{h(i)}(\sigma) = M_i'(\sigma)$. Otherwise, $M_{h(i)}(\sigma) = M(\sigma)$. It is easy to verify that $M_{h(i)}$ I-learns $\mathcal{S} \cup I(M_i)$. □

Theorem 14. Suppose I is either $\mathbf{Ex}$ or $\mathbf{BC}$. Suppose $\mathcal{S} \in \mathbf{WConf}$. Then $\mathcal{S}$ is constructively singleton-I-unionable.

Proof. Let f be a recursive function such that $M_{f(e)}$ always outputs e on any input. Then, $M_{f(e)}$ $\mathbf{WConf}$-learns $\{\varphi_e\}$. Let M_i be a $\mathbf{WConf}$-learner for $\mathcal{S}$. Let $h_{\mathbf{WConf}}$ be as from Proposition 5. Then, $h_{\mathbf{WConf}}(f(e), i)$ witnesses the theorem. □

Corollary 15. Suppose I is either $\mathbf{Ex}$ or $\mathbf{BC}$. Let $\mathcal{S} = \{f \in \mathcal{R} : \varphi_{f(0)} = f\}$. Then, $\mathcal{S}$ is constructively singleton-I-unionable, but not I-unionable.

Theorem 16. There are classes $\mathcal{S}, \mathcal{S}' \subseteq \mathcal{R}$ such that

(a) $\mathcal{S}$ and $\mathcal{S}'$ are both $\mathbf{Ex}$-learnable;
(b) $\mathcal{S}$ and $\mathcal{S}'$ are both constructively $\mathbf{BC}$-unionable;
(c) $\mathcal{S} \cup \mathcal{S}'$ is not $\mathbf{Ex}$-learnable;
(d) $\mathcal{S}$ is not constructively singleton-$\mathbf{Ex}$-unionable;
(e) $\mathcal{S}'$ is constructively singleton-$\mathbf{Ex}$-unionable.

Proof. Kummer and Stephan [8, Theorem 8.1] constructed a uniformly partial-recursive family $\varphi_{g(0)}, \varphi_{g(1)}, \ldots$ of functions such that each $\varphi_{g(n)}$ is undefined at most at one place and $1^n0 \preceq \varphi_{g(n)}$ for all n. Let $\mathcal{S}$ be the set of all total extensions of functions $\varphi_{g(n)}$ which are not total. Let $\mathcal{S}'$ be set of all total $\varphi_{g(n)}$. It is easy to verify that $\mathcal{S}$ and $\mathcal{S}'$ are both in $\mathbf{Ex}$.

Kummer and Stephan [8] showed that $\mathcal{S} \cup \mathcal{S}'$ is $\mathbf{BC}$-learnable. Actually $\mathcal{S} \cup \mathcal{S}'$ and every subclass of it is constructively $\mathbf{BC}$-unionable. To see this, let *patch* be a recursive function such that $\varphi_{patch(i,\sigma)}(x) = \sigma(x)$ if $x < |\sigma|$; $\varphi_{patch(i,\sigma)}(x) = \varphi_i(x)$ if $x \geq |\sigma|$.

Now, let any total $\mathbf{BC}$-learner M for some class be given. Now, a new $\mathbf{BC}$-learner N, obtained effectively from M, learning $\mathbf{BC}(M) \cup \mathcal{S} \cup \mathcal{S}'$ is defined as follows:

```
If there is an $n$ such that $1^n0 \preceq \sigma$ and no $x < |\sigma|$ satisfies that $\varphi_{g(n)}(x)$
    converges within $|\sigma|$ steps to a value different from $\sigma(x)$,
Then $N(\sigma) = patch(g(n), \sigma)$,
Else $N(\sigma) = M(\sigma)$.
```

Furthermore, Kummer and Stephan [8] showed that $\mathcal{S} \cup \mathcal{S}'$ is not $\mathbf{Ex}$-learnable, hence $\mathcal{S}$ and $\mathcal{S}'$ are not $\mathbf{Ex}$-unionable. As $\mathcal{S}'$ is $\mathbf{Fin}$-learnable, by Theorem 14, $\mathcal{S}'$ is also constructively singleton-$\mathbf{Ex}$-unionable.

Furthermore, $\mathcal{S}$ is not constructively singleton-$\mathbf{Ex}$-unionable. Suppose by way of contradiction that h witnesses that $\mathcal{S}$ is constructively singleton-$\mathbf{Ex}$-unionable.

Then, the following learner N witnesses that $\mathcal{S} \cup \mathcal{S}' \in \mathbf{Ex}$: If $1^n0 \preceq \sigma$ for some n, then $N(\sigma) = M_{h(g(n))}(\sigma)$, else $N(\sigma) = 0$. However, by Kummer and Stephan [8], such a learner does not exist. □

Theorem 17. There is a class $\mathcal{S}$ which is **Ex**-unionable, **BC**-unionable, but is not constructively singleton-**BC**-unionable.

Proof. For each n, we will define function f_n below. The class $\mathcal{S}$ will consist of all functions of the form $f_n(0)f_n(1)\ldots f_n(x)y^\infty$ which start with values of some f_n until a point x and are constant from then onwards.

Without loss of generality assume that learner M_0 **Ex**-learns all eventually constant functions. The functions f_n satisfy the following properties:

(I) $f_n(0) = n$;
(II) Each f_n is recursive;
(III) The mapping $n, x \mapsto f_n(x)$ is limit-recursive;
(IV) For each $m \leq n$,
either for infinitely many s, $(\exists x)\,[\varphi_{M_m(f_n[s])}(x){\downarrow} \neq f_n(x)]$,
or there is a $\sigma \preceq f_n$ such that $(\forall \tau)\,[\varphi_{M_m(\sigma\tau)}$ is a subfunction of $\sigma\tau]$.

Note that the above properties imply that M_m does not **BC**-learn f_n, for any $n \geq m$. Thus, in particular, f_n is not an eventually constant function.

The construction of f_n is done by inductively defining longer and longer initial segments $f_n[\ell_{n,t}]$ of f_n together with the length $\ell_{n,t}$. Let $\ell_{n,0} = 0$. In stage t, $\ell_{n,t+1}$ and $f_n[\ell_{n,t+1}]$ are defined as follows: Let m be the remainder of t divided by $n+1$. Search for τ, η, a hypothesis e and an $x < \ell_{n,t} + |\tau\eta|$ such that $\varphi_{M_m(f_n[\ell_{n,t}]\cdot\tau)}(x){\downarrow} \neq (f_n[\ell_{n,t}] \cdot \tau\eta)(x)$. If such τ, η, e, x are found then $\ell_{n,t+1} = \ell_{n,t} + |\tau\eta| + 1$ and $f_n[\ell_{n,t+1}] = f_n[\ell_{n,t}] \cdot \tau\eta \cdot 0$ else $\ell_{n,t+1} = \ell_{n,t} + 1$ and $f_n[\ell_{n,t+1}] = f_n[\ell_{n,t}] \cdot 0$.

Note that if the search does not succeed in stage t then it does not succeed in stage $t+n+1$ either, as that stage also deals with the same m and $f_n[\ell_{n,t+n+1}]$ is an extension of $f_n[\ell_{n,t}]$. Therefore each f_n is recursive. Furthermore, the f_n are uniformly limit-recursive as one can use the oracle for K to decide whether the extension exists in each specific case. It is clear that property (IV) of f_n mentioned above is also met by the way each f_n is constructed.

Now suppose that a total learner M_e **Ex**-learns or **BC**-learns a class $\mathcal{S}'$. Thus the functions $f_e, f_{e+1}, f_{e+2}, \ldots$ are not learnt by M_e and thus not members of $\mathcal{S}'$. Now consider the following new learner N for $\mathcal{S} \cup \mathcal{S}'$. Let $f_{n,t}$ be the t-th approximation (as a recursive function) to f_n; the $f_{n,t}$ converge pointwise to f_n. N, on input σ of length $t > 0$, is defined as follows:

```
If $\sigma \preceq f_d$ for some $d \in \{0, 1, \ldots, e\}$,
Then $N(\sigma)$ is an index for $f_d$ for the least such $d$,
Else if $\sigma = f_{n,t}(0)f_{n,t}(1)\ldots f_{n,t}(x)y^{t-x-1}$ for some $n, y$ and $x < t-1$,
Then $N(\sigma)$ outputs a canonical index for $f_{n,t}(0)f_{n,t}(1)\ldots f_{n,t}(x)y^\infty$,
Else $N(\sigma) = M_e(\sigma)$.
```

One can easily verify that N **Ex**-learns $f_0, f_1, \ldots, f_e$ and also **Ex**-learns every member of $\mathcal{S}$. Furthermore, for each $f \in \mathcal{S}' - \mathcal{S} - \{f_0, f_1, \ldots, f_e\}$, there are $n = f(0)$, a least x with $f(x+1) \neq f_n(x+1)$ and a least $x' > x$ with $f(x'+1) \neq f(x')$. If $\sigma \preceq f$ is long enough, then $f_{n,|\sigma|}$ equals f_n for inputs below $x+2$ and $|\sigma| > x'+1$ and thus the learner N outputs $M_e(\sigma)$. Hence if M_e is an **Ex**-learner for $\mathcal{S}'$ then N is an **Ex**-learner for $\mathcal{S} \cup \mathcal{S}'$ and if M_e is a **BC**-learner for $\mathcal{S}'$ then N is a **BC**-learner for $\mathcal{S} \cup \mathcal{S}'$.

Now assume by way of contradiction that $\mathcal{S}$ is constructively singleton-**BC**-unionable as witnessed by a recursive function h. We will define a learner N below. For ease of notation, we define N as running in stages and think of learners as getting the graph of the whole function as input, and outputting a sequence of conjectures, all but finitely many of which are programs for the input function (for **BC**-learning); for **Ex**-learning, this sequence of programs also converges syntactically.

Let f denote the function to be learnt and let $n = f(0)$. Now define a trigger-event m to be activated iff there is a $t > m$ such that $f[m] \preceq f_{n,t}$ (as defined above). If $f = f_n$ then infinitely many trigger events are eventually activated; otherwise only finitely many trigger events are eventually activated. On any input function f, the learner N starts in stage 0.

```
Stage $\langle i, j\rangle$:
    In this stage $N$ copies the output of $M_{h(i)}$ until
      (i) the $(\langle i, j\rangle + 1)$-th trigger event has been activated and
      (ii) there are $x, z$ such that $x > j$ and $\varphi_{M_{h(i)}(f[x])}(x) \downarrow \neq f(z)$.
    When both events have occurred, the learner $N$ leaves stage $\langle i, j\rangle$
    and goes to the next stage $\langle i, j\rangle + 1$.
End stage $\langle i, j\rangle$
```

Note that whenever the input function f is from $\mathcal{S}$, then only finitely many trigger-events are activated and therefore the construction leaves only finitely many stages. Hence, the learner N eventually follows the learner $M_{h(i)}$, for some i, and thus **BC**-learns f.

Let n be such that $M_n = N$. Consider the behaviour of N on f_n. As, for each prefix σ of f_n, N **BC**-learns $\sigma 0^\infty$, it follows from property (IV) of f_n that there exist infinitely many x such that, for some z, $\varphi_{N(f_n[x])}(z){\downarrow} \neq f_n(z)$. Furthermore, infinitely many trigger events are activated on input function being f_n. Thus, inductively, for each stage $\langle i, j\rangle$, $\varphi_{M_{h(i)}(f_n[x])}(z){\downarrow} \neq f_n(z)$, for some $x > j$. Therefore, for all i, $\varphi_{M_{h(i)}(f_n[x])} \neq f_n$, for infinitely many x. Thus, for each i, $M_{h(i)}$ does not **BC**-learn f_n. However, as there exists an i such that $f_n = \varphi_i$, the learner $M_{h(i)}$ must **BC**-learn f_n. A contradiction. Thus, $\mathcal{S}$ is not constructively singleton-**BC**-unionable. □

Corollary 18. Due to the implications among the criteria of unionability, the class $\mathcal{S}$ from Theorem 17 also fails to be constructively singleton-**Ex**-unionable, constructively **BC**-unionable or constructively **Ex**-unionable. Furthermore, $\mathcal{S}$ is not **WConf**-learnable.

The next proposition shows that **Ex** and **BC**-unionable classes are everywhere sparse.

Proposition 19. Suppose I is **Ex** or **BC**. Suppose $\mathcal{S}$ is not everywhere sparse. Then $\mathcal{S}$ is neither I-unionable nor constructively singleton-I-unionable.

The following theorem generalises the Non-Union-Theorem.

Theorem 20. Let $\mathcal{S} \subseteq \mathcal{R}$ be **Ex**-learnable. Then there are $\mathcal{S}_0 \subseteq \mathcal{R}$ and $\mathcal{S}_1 \subseteq \mathcal{R}$ such that $\mathcal{S} \cup \mathcal{S}_0$ and $\mathcal{S} \cup \mathcal{S}_1$ are **Ex**-learnable but $\mathcal{S}_0 \cup \mathcal{S}_1$ is not **BC**-learnable.

5 Extendability

In the previous sections, the question was whether a class $\mathcal{S}$ can be extended by either adding a full class $\mathcal{S}'$ or just a function φ_e without losing learnability; in this section we ask whether a class can be extended effectively without prescribing how this should be done. So on one hand, the task becomes easier as it is not prescribed what to add, on the other hand the task might also become more difficult as one has to find functions not yet learnt in order to add them (while previously, they were given by a learner or an index). Before discussing this in detail, the next definition should make the notion of extending more precise.

Definition 21. Let $\mathcal{C}$ be a set of learners and I a learning criterion.

(a) We say that we can *infinitely I-improve learners from $\mathcal{C}$* iff, for all $M \in \mathcal{C}$, there is a learner $N \in \mathcal{P}$ such that $I(M) \subseteq I(N)$ and $I(N) \setminus I(M)$ is infinite.

(b) We say that we can *uniformly infinitely I-improve learners from $\mathcal{C}$* iff there is a recursive function h such that, for all e with $M_e \in \mathcal{C}$, $I(M_e) \subseteq I(M_{h(e)})$ and $I(M_{h(e)}) \setminus I(M_e)$ is infinite.

Proposition 22. Let $\mathcal{C}$ be a set of learners and I be **Ex** or **BC**. Suppose there is a function $g \in \mathcal{R}$ such that, for all e with $M_e \in \mathcal{C}$, $\{\varphi_{g(e,x)} \mid x \in \mathbb{N}\}$ is an infinite I-unionable set disjoint from $I(M_e)$. Furthermore, assume that one can determine with a two-sided classifier effectively obtainable from e, for each recursive function f, whether $f \in \{\varphi_{g(e,x)} \mid x \in \mathbb{N}\}$. Then we can uniformly infinitely I-improve learners from $\mathcal{C}$.

Lemma 23. Suppose $\mathcal{C}$ is a set of learners and $\sigma_0 \in \mathbb{S}\mathrm{eq}$. Suppose for all e, σ one can effectively find a sequence $\tau_{e,\sigma}$ such that if $M_e \in \mathcal{C}$ and $\sigma_0 \preceq \sigma$, then $\sigma \preceq \tau_{e,\sigma}$ and $M_e(\sigma) \neq M_e(\tau_{e,\sigma})$. Then we can uniformly infinitely **Ex**-improve all learners from $\mathcal{C}$.

Proof. By implicit use of the parametric recursion theorem [12], let g be a recursive function such that, for all e, x,

$$\varphi_{g(e,x)} = \bigcup_s \varphi^s_{g(e,x)} \text{ where } \varphi^0_{g(e,x)} = \sigma_0 \cdot e \cdot x \text{ and } \varphi^{s+1}_{g(e,x)} = \tau_{e,\varphi^s_{g(e,x)}}.$$

Now, each $M_e \in \mathcal{C}$ fails to **Ex**-learn every $\varphi_{g(e,x)}$, $x \in \mathbb{N}$. Furthermore, there is a two-sided classifier for each of the classes $\{\varphi_{g(e,x)} \mid x \in \mathbb{N}\}$. The theorem now follows from Proposition 22. □

Case and Fulk [4] showed that every **Ex**-learner can be infinitely extended. Furthermore, for the subclass of learners learning a dense set of functions, an effective procedure is implicitly given for turning any such learner into an infinitely more successful one.

Theorem 24 (Case and Fulk [4]). We can infinitely **Ex**-improve every learner. Furthermore, we can *uniformly* infinitely **Ex**-improve all learners M where $\mathbf{Ex}(M)$ is dense.

As an open question, Case and Fulk [4] asked whether there is another effective procedure for the complement, that is, for learners that are not dense.

The next theorem answers this question in the negative by showing that there is no computable function turning any given (index for an) **Ex**-learner which is not successful on a dense set into an (index for a) strictly more successful learner – not even by a single additional function.

Theorem 25. For every recursive function h there is a learner M_e such that $[\mathbf{Ex}(M_e)] \neq [\mathcal{R}]$ and $\mathbf{Ex}(M_{h(e)})$ is not a strict superset of $\mathbf{Ex}(M_e)$.

Proof. It suffices to show that for every recursive h, there is an index e with $[\mathbf{Ex}(M_e)] \neq [\mathcal{R}]$ and either $\mathbf{Ex}(M_{h(e)}) \not\supseteq \mathbf{Ex}(M_e)$ or $\mathbf{Ex}(M_{h(e)}) \setminus \mathbf{Ex}(M_e)$ contains at most one function. (As if, for some recursive function h', for every e, $M_{h'(e)}$ is such that $\mathbf{Ex}(M_{h'(e)})$ exceeds $\mathbf{Ex}(M_e)$ by at least one function, then $\mathbf{Ex}(M_{h'(h'(e))})$ would exceed $\mathbf{Ex}(M_e)$ by at least two functions).

Suppose, by way of contradiction, that there is a recursive function h such that, for all e with $[\mathbf{Ex}(M_e)] \neq [\mathcal{R}]$, $\mathbf{Ex}(M_{h(e)})$ contains $\mathbf{Ex}(M_e)$ and exceeds it by at least two functions.

We define a recursive function g implicitly by inductively defining, for any $e \in \mathbb{N}$, a (possibly finite) $\preceq$-increasing sequence of sequences $(\sigma_i^e)_{i \in \mathbb{N}}$ and a recursive function g by

$$\begin{array}{rcl} \sigma_0^e & = & \Lambda; \\ \forall i\, [\sigma_{i+1}^e \text{ is the first} & \sigma \succ \sigma_i^e & \text{found such that } M_{h(e)}(\sigma)\downarrow \neq M_{h(e)}(\sigma_i^e)\downarrow]; \\ \varphi_{g(e)} & = & \displaystyle\bigcup_{i \in \mathbb{N}} \sigma_i^e. \end{array}$$

We let k be a recursive function such that, for all e, τ, $k(e,\tau)$ is the maximum i such that σ_i^e is defined within $|\tau|$ steps. By Kleene's recursion theorem, there is a program e such that, for all τ,

$$M_e(\tau) = \begin{cases} g(e), & \text{if } \exists i\, [\tau \preceq \sigma_i^e]; \\ \mathrm{pad}(M_{h(e)}(\tau), k(e,\tau)), & \text{if } \exists i\, [\sigma_i^e \cup \tau \text{ is not single-valued}]; \\ \uparrow, & \text{otherwise.} \end{cases}$$

Now if M_e does not learn a dense set of functions, then $\mathbf{Ex}(M_{h(e)})$ must exceed $\mathbf{Ex}(M_e)$ by at least two more functions.

Case 1: $\varphi_{g(e)}$ is total.
Then M_e **Ex**-learns *only* $\varphi_{g(e)}$; thus, $M_{h(e)}$ **Ex**-learns $\varphi_{g(e)}$ by supposition. However, by construction of σ_i^e and $g(e)$, $M_{h(e)}$ on $\varphi_{g(e)}$ makes infinitely many mind changes, a contradiction.

Case 2: σ_i^e is defined only for finitely many i.
Let i be the maximum such that σ_i^e is defined. Thus, M_e is undefined on any extension of σ_i^e, and, hence, does not learn a dense set. Suppose $f \in \mathcal{R}$ does not extend σ_i^e. For all j large enough, we now have $M(f[j]) = \mathrm{pad}(M_{h(e)}(f[j]), i)$. Thus, for large enough j, $M(f[j])$ is semantically equivalent to $M_{h(e)}(f[j])$. Thus, any function that is not an extension of σ_i^e, is **Ex**-learned by $M_{h(e)}$ iff it is **Ex**-learned by M_e. Thus, as $M_{h(e)}$ never changes its mind beyond σ_i^e, on any extension of σ_i^e, it can **Ex**-learn at most *one* more function than M_e, a contradiction. □

As an immediate corollary, we get that we cannot constructively find initial segments where a given learner does not learn any extension.

Corollary 26. There is no function $g \in \mathcal{P}$ such that, for all e with $\mathbf{Ex}(M_e)$ not dense, we have that $g(e)$ is a finite sequence with $g(e) \notin [\mathbf{Ex}(M_e)]$.

Case and Fulk [4] ask whether there is any partitioning of all learners into two (or at least finitely many) sets such that, for each of the sets, all learners from that set can be uniformly extended. From Theorem 25 we know that this partitioning cannot be according to whether the set of learned functions is dense. The following theorem answers the open problem by giving a different split of all possible learners into two different classes.

Theorem 27. Let $\mathcal{C}$ be the set of all total learners M such that M changes its mind on a dense set of sequences. Then we can uniformly infinitely **Ex**-improve all learners from $\mathcal{C}$ and from $\mathcal{R} \setminus \mathcal{C}$.

Proof. It follows from Lemma 23, that we can uniformly infinitely **Ex**-improve learners from $\mathcal{C}$. We now consider the case of extending learners from $\mathcal{R} \setminus \mathcal{C}$. For any given e and t, let $\tau_{e,t}$ denote the length-lexicographically first sequence found such that M_e does not change its mind on the first t extensions of $\tau_{e,t}$. For any sequence σ and any b we let $g(\sigma, b)$ denote an index for σb^∞. Let $h \in \mathcal{R}$ be such that, for all e and σ,

$$M_{h(e)}(\sigma) = \begin{cases} g(\tau_{e,|\sigma|}, b), & \text{if there is } b \text{ with } \sigma \preceq \tau_{e,|\sigma|} b^\infty; \\ M_e(\sigma), & \text{otherwise.} \end{cases}$$

For all e with $M_e \in \mathcal{R} \setminus \mathcal{C}$, we have that the sequence $\tau_{e,0}, \tau_{e,1}, \ldots$ converges to a τ_e such that M_e does not make any mind changes on any extension of τ_e. Now, $M_{h(e)}$ learns $\mathbf{Ex}(M_e) \cup \{\tau_e \cdot b^\infty \mid b \in \mathbb{N}\}$. Note that M_e can **Ex**-learn at most one function extending τ_e. The theorem follows. □

As one can effectively convert any partial learner to a total learner with the same (or more) learning capacity, the above result also applies for partial learners.

For **Fin**-learning, extending learners is much easier: any learner that learns anything at all can be infinitely extended.

Theorem 28. Let I be one of $\mathbf{Ex}_m$ or **Conf**. There is a function h such that, for all e with $I(M_e) \neq \emptyset$, $I(M_{h(e)})$ infinitely extends $I(M_e)$. Here $M_{h(e)}$ is confident, if M_e is confident.

Similarly for reliable learning, one can always extend a learner infinitely.

Theorem 29. There is a recursive function h such that, for e with M_e reliable, $M_{h(e)}$ is reliable and $\mathbf{Ex}(M_{h(e)})$ infinitely extends $\mathbf{Ex}(M_e)$.

References

[1] Bārzdiņš, J.A.: Two theorems on the limiting synthesis of functions. Theory of Algorithms and Programs, Latvian State University, Riga, USSR 210, 82–88 (1974)

[2] Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. Information and Control 28, 125–155 (1975)

[3] Case, J.: Periodicity in generations of automata. Mathematical Systems Theory 8, 15–32 (1974)

[4] Case, J., Fulk, M.: Maximal machine learnable classes. Journal of Computer and System Sciences 58, 211–214 (1999)

[5] Case, J., Jain, S., Manguelle, S.N.: Refinements of inductive inference by Popperian and reliable machines. Kybernetika 30, 23–52 (1994)

[6] Case, J., Smith, C.: Comparison of identification criteria for machine inductive inference. Theoretical Computer Science 25, 193–220 (1983)

[7] Gold, M.: Language identification in the limit. Information and Control 10, 447–474 (1967)

[8] Kummer, M., Stephan, F.: On the structure of degrees of inferability. Journal of Computer and System Sciences 52, 214–238 (1996)

[9] Minicozzi, E.: Some natural properties of strong-identification in inductive inference. Theoretical Computer Science 2, 345–360 (1976)

[10] Osherson, D., Stob, M., Weinstein, S.: Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists. MIT Press, Cambridge (1986)

[11] Pitt, L.: Inductive Inference, DFAs, and Computational Complexity. In: Jantke, K.P. (ed.) AII 1989. LNCS (LNAI), vol. 397, pp. 18–44. Springer, Heidelberg (1989)

[12] Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw Hill, New York (1967); (reprinted in 1987)

[13] Stephan, F.: On one-sided versus two-sided classification. Archive for Mathematical Logic 40, 489–513 (2001)

[14] Sharma, A., Stephan, F., Ventsov, Y.: Generalized notions of mind change complexity. Information and Computation 189, 235–262 (2004)

Confident and Consistent Partial Learning of Recursive Functions

Ziyuan Gao[1] and Frank Stephan[2,⋆]

[1] Department of Mathematics,
National University of Singapore, Singapore 119076, Republic of Singapore
ziyuan84@yahoo.com

[2] Department of Mathematics and Department of Computer Science,
National University of Singapore, Singapore 119076, Republic of Singapore
fstephan@comp.nus.edu.sg

Abstract. Partial learning is a criterion where the learner infinitely often outputs one correct conjecture while every other hypothesis is issued only finitely often. This paper addresses two variants of partial learning in the setting of inductive inference of functions: first, confident partial learning requires that the learner also on those functions which it does not learn, singles out exactly one hypothesis which is output infinitely often; second, essentially class consistent partial learning is partial learning with the additional constraint that on the functions to be learnt, almost all hypotheses issued are consistent with all the data seen so far. The results of the present work are that confident partial learning is more general than explanatory learning, incomparable with behaviourally correct learning and closed under union; essentially class consistent partial learning is more general than behaviourally correct learning and incomparable with confident partial learning. Furthermore, it is investigated which oracles permit to learn all recursive functions under these criteria: for confident partial learning, some non-high oracles are omniscient; for essentially class consistent partial learning, all PA-complete and all oracles of hyperimmune Turing degree are omniscient.

1 Introduction

Gold [6] initiated the study of inductive inference, which investigates various forms of learning recursive functions and r.e. sets in the limit. Gold originally considered learners which syntactically converge to the correct conjecture; Osherson, Stob and Weinstein [14] generalised learning to *partial learning*, where a recursive learner which receives piecewise information about the graph of an unknown recursive function, presented in the natural ordering of the input values. At each stage, the learner is required to output a conjecture based on a pre-assigned hypothesis space - usually taken to be a fixed acceptable numbering of all partial-recursive functions - and is judged to have successfully learnt

[⋆] Work is supported in part by NUS grant R252-000-420-112.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 51–65, 2012.

a target function if it outputs exactly one correct identification of the function infinitely often and outputs any other conjecture only finitely often.

On one hand, many natural examples of classes of recursive functions fail to be identifiable in the limit by any recursive learner, even in the broadest sense of semantic convergence [1]; this deficiency has spurred various alternative approaches to learnability in the inductive inference literature. Feldman [4], for example, showed that a decidable rewriting system (drs) is always learnable from positive information sequences in a certain restricted sense. On the other hand, Osherson, Stob and Weinstein [14] discovered that the whole class *REC* of recursive functions is partially learnable and partial learnability is much more general than even behaviourally correct learnability. Subsequently, researchers thought that this is too general and studied what happens when partial learning is combined with more restrictive constraints, most notably consistency which was introduced by Bārzdiņš [1]. Indeed, consistent partial learners can easily be shown to fail the class of all recursive functions. Wiehagen and Zeugmann [17] and later Grieser [7] and Jain and Stephan [10] studied consistent learning and partial consistent learning. Other constraints of partial learning were neglected, mostly as the corresponding notions coincided with partial learning itself.

The present work wants to fill this gap and starts with carrying over the notion of partial confident learning which was introduced in a language learning setting by Gao, Stephan, Wu and Yamamoto [5]. Confidence in partial learning enforces that the learner must issue exactly one hypothesis infinitely often on the text for any total function. In the case of language learning, the notion turned out to be restrictive [5]: even the class of all cofinite sets is not confidently partially learnable.

On the other hand, confident partial learning has some regularity properties. In the here investigated case of function learning, one can show that the union of confidently partially learnable classes is confidently partially learnable (this is parallel to the corresponding result for confidently explanatory learning of classes of functions); furthermore, this notion is more general than Gold's original notion of explanatory learning [3, 6] and incomparable to the more general notion of behaviourally correct learning [1].

Consistency, whilst a fairly stringent learning criterion, may be quite a desirable quality of learners, especially when the inductive inference paradigm is viewed as a model for scientific discovery. It is conceivable that a scientific theory with any epitemic value must be developed in accordance with empirical data, and, while allowing for a certain margin of error due to experimental inaccuracies, should possess a set of potential falsifiers that determine the consistency or non-consistency of its fundamental assumptions under the conditions of a controlled experiment [12]. Briefly, the falsificationist methodological rule expounded by Popper [15] states that a scientific theory is to be rejected if it is inconsistent with some basic statement unanimously accepted by the scientific community. In view of this benchmark by which science progresses, one may argue that consistency with empirical data is an essential characteristic of the hypotheses issued by scientists modelled as recursive learners.

Jain and Stephan [10] showed that the class REC of all recursive functions can be consistently partially learnt relative to an oracle A if and only if A has hyperimmune degree. In the present paper, we show that by weakening this learning constraint to *essential consistency*, under which a recursive learner is only required to be consistent on cofinitely many segments of a text input, REC can be partially inferred relative to any PA-complete oracle. Thus, by the result of Jockusch and Soare [11] that there are hyperimmune-free PA-complete sets, one can conclude that there is a strictly larger family of oracles relative to which REC is essentially class consistently partially learnable. The main result for this notion is that it is still more general than behaviourally correct learning; this is a surpising result as usually the generalisations of behaviourally correct learning are either obtained by varying the concept of semantic convergence (for example, by augmenting it with errors) or by taking a notion which is already learning the full class REC. Further results on essentially class consistent learning in the present work are that this notion is neither closed under union nor comparable to confident partial learning.

2 Notation

The notation and terminology from recursion theory adopted in this paper follows the book of Rogers [16] in the main. Background on inductive inference can be found in [9]. $\mathbb{N}$ denotes the set of natural numbers. Let $\varphi_0, \varphi_1, \varphi_2, \ldots$ denote a fixed acceptable numbering of all partial-recursive functions. Given a set S, $\overline{S}$ denotes the complement of S, and S^* denotes the set of all finite sequences in S. Let $W_0, W_1, W_2, \ldots$ be a universal numbering of all r.e. sets, where W_e is the domain of φ_e. $\langle x, y\rangle$ denotes Cantor's pairing function, given by $\langle x, y\rangle = \frac{1}{2}(x+y)(x+y+1)+y$. $W_{e,s}$ is an approximation to W_e; without loss of generality, $W_{e,s} \subseteq \{0, 1, \ldots, s\} \cap W_{e,s+1}$ and the set $\{\langle e, x, s\rangle : x \in W_{e,s}\}$ is primitive recursive. $\varphi_e(x)\uparrow$ means that $\varphi_e(x)$ remains undefined; $\varphi_{e,s}(x)\downarrow$ means that $\varphi_e(x)$ is defined, and that the computation of $\varphi_e(x)$ halts within s steps. Turing reducibility is denoted by $\leq_T$; $A \leq_T B$ holds if A can be computed via a machine which knows B, that is, for any given x, it gives information on whether or not x belongs to B. $A \equiv_T B$ means that $A \leq_T B$ and $B \leq_T A$ both hold, and $\{A : A \equiv_T B\}$ is called the Turing degree of B. The class of all recursive functions is denoted by REC; the class of all $\{0, 1\}$-valued recursive functions is denoted by $REC_{0,1}$. For any two partial-recursive functions f and g, $f =^* g$ denotes that for cofinitely many x, $f(x)\downarrow = g(x)\downarrow$. The symbol $\mathbb{K}$ denotes the diagonal halting problem. The jump of a set A is denoted by A' and denotes the relativised halting problem $A' = \{e : \varphi_e^A(e)\downarrow\}$. For any two sets A and B, $A \oplus B = \{2x : x \in A\} \cup \{2y+1 : y \in B\}$. Analogously, $A \oplus B \oplus C = \{3x : x \in A\} \cup \{3y+1 : y \in B\} \cup \{3z+2 : z \in C\}$.

For any $\sigma, \tau \in (\mathbb{N} \cup \{\#\})^*$, $\sigma \preceq \tau$ if and only if $\sigma = \tau$ or τ is an extension of σ, $\sigma \prec \tau$ if and only if σ is a proper prefix of τ, and $\sigma(n)$ denotes the element in the nth position of σ, starting from $n = 0$. Given a number a and some fixed $n \geq 1$, denote by a^n the finite sequence $a \ldots a$, where a occurs n times. a^0 denotes the empty string. $|\sigma|$ is the length of σ. The concatenation of two strings σ and τ shall be denoted by $\sigma\tau$ and occasionally by $\sigma \circ \tau$.

Jockusch and Soare [11], as well as Hanf [8], established that the class of degrees of members of a given $\prod_1^0$ class coincides with the class of degrees of complete extensions of some finitely axiomatizable first-order theory; a set which falls within the latter class is called *PA-complete*. In this paper, we adopt the following equivalent definition of PA-completeness.

Definition 1. A set A is *PA-complete* if and only if, given any partial-recursive and $\{0,1\}$-valued function ψ, one can compute relative to A a total extension Ψ of ψ. A set A is *low* if and only if its jump is Turing equivalent to the halting problem: $A' \equiv_T \mathbb{K}$. For any m, a set A is *m-generic* if for every Σ_m^0 set $W \subseteq \{0,1\}^*$ there is an n such that either $A(0) \circ A(1) \circ \ldots \circ A(n) \in W$ or no extension of $A(0) \circ A(1) \circ \ldots \circ A(n)$ belongs to W. g^A denotes that g is computed using A.

3 Learnability

Let $\mathcal{C}$ be a class of recursive functions. Throughout this paper, the mode of data presentation is that of a *canonical text*, by which is meant an infinite sequence whose ith term is $\langle i, f(i)\rangle$, where f is some total function. Formally, the *canonical text* T_f for some f in $\mathcal{C}$ is the map $T_f : \mathbb{N} \to \mathbb{N}$ such that $T_f(n) = f(n)$ for all n. $T_f[n]$ denotes the string $T_f(0) \circ T_f(1) \circ \ldots \circ T_f(n)$. The main learning criteria studied in this paper are *partial learning*, *explanatory learning* and *behaviourally correct learning*. The learning success criteria are defined below with respect to learning from canonical texts and M is supposed to be recursive.

i. Osherson, Stob and Weinstein [14] defined that M *partially* (*Part*) learns $\mathcal{C}$ if, for each f in $\mathcal{C}$, there is exactly one index e such that $M(T_f[k]) = e$ for infinitely many k; this index e also satisfies $f = \varphi_e$.
ii. Gold [6] defined that M *explanatorily* (*Ex*) learns $\mathcal{C}$ if, for each f in $\mathcal{C}$, there is a number n for which $f = \varphi_{M(T_f[n])}$ and, for any $j \geq n$, $M(T_f[j]) = M(T_f[n])$.
iii. Bārzdiņš [1] defined that M *behaviourally correctly* (*BC*) learns $\mathcal{C}$ if, for each f in $\mathcal{C}$, there is a number n for which $f = \varphi_{M(T_f[j])}$ whenever $j \geq n$.

The next two definitions impose additional constraints on the learner.

Definition 2. i. [5] A recursive learner M is said to *confidently partially* learn $\mathcal{C}$ if it partially learns $\mathcal{C}$ from canonical text and outputs on every infinite sequence exactly one index infinitely often.
ii. A recursive learner M is said to *essentially class consistently partially* learn $\mathcal{C}$ if it partially learns $\mathcal{C}$ from canonical text and, for each f in $\mathcal{C}$, $\varphi_{M(T_f[n])}(m)\downarrow = f(m)$ holds whenever $m \leq n$ for cofinitely many n.

Blum and Blum [3] introduced the notion of a *locking sequence* for explanatory learning, whose existence is a necessary criterion for a learner to successfully identify the recursive function generating the text seen. With a slight modification, one can adapt this concept to the partial learning model.

Definition 3. Let M be a recursive learner and f be a recursive function partially learnt by M. Then there is a finite sequence $\sigma = f(0) \circ f(1) \circ \ldots \circ f(n)$ such that

- $\varphi_{M(\sigma)} = f$;
- For all $k > n$, there is a $l > k$ such that $M(f(0) \circ f(1) \circ \ldots \circ f(k) \circ \ldots \circ f(l)) = M(\sigma)$.

This σ shall be called a *locking sequence* for f.

4 Confident Partial Learning

The first learning constraint proposed here as a means of sharpening partial learnability is that of *confidence*. Osherson, Stob and Weinstein [14] introduced confidence for explanatory and other learning notions by defining that a confident learner provides on each function a hypothesis with respect to the given learning criterion and this hypothesis has to be correct on all functions in the class to be learnt. It is known that confidence is a real restriction for finite, explanatory and behaviourally correct learning compared to the non-confident versions of the respective learning criteria. The following result shows that confidence is also restrictive for partial learning; there is, in fact, a class which is behaviourally correctly learnable but not confidently partially learnable.

Theorem 4. *There is a behaviourally correctly learnable class of recursive functions which is not confidently partially learnable.*

Proof. Consider the class $\mathcal{C} = \{f : f$ is recursive and $\{0,1\}$-valued $\wedge\ \exists e[|\overline{W}_e| < \infty \wedge f(e+1) = 1 \wedge \forall x \leq e[f(x) = 0] \wedge f =^* \varphi_e]\}$. A behaviourally correct learner M outputs a default index 0 until it witnesses the first number e such that $f(x) = 0$ for all $x \leq e$ and $f(e+1) = 1$; subsequently, on the input $\sigma = 0^e \circ 1 \circ f(e+2) \circ \ldots \circ f(e+k)$, it conjectures an index i with $\varphi_i(x) = \sigma(x)$ if $x < |\sigma|$, and $\varphi_i(x) = \varphi_e(x)$ otherwise.

Assume by way of contradiction that one may define a recursive confident partial learner N of the class $\mathcal{C}$. It shall be shown that this implies the existence of a $\mathbb{K}'$-recursive procedure for deciding whether $d \in \{e : W_e$ is cofinite$\}$ for any given d, contradicting the known fact that the latter set is Σ_3^0-complete. First, let g be a recursive function for which $\varphi_{g(d)}$ is defined in stages as follows:

- Set $\varphi_{g(d),0}(x) \uparrow$ for all x. Initialise the markers $a_0, a_1, a_2, \ldots$ by setting $a_{i,0} = \langle i, 0\rangle + d + 1$ for $i \in \mathbb{N}$.
- At stage $t+1$, consider the markers $a_{0,t}, a_{1,t}, a_{2,t}, \ldots, a_{t,t}$ with $a_{i,t} = \langle i, r\rangle + d + 1$, and perform the following: if neither $\varphi_{g(d),t}$ nor $\varphi_{i,t}$ is defined on the input $\langle i, j\rangle + d + 1$ for $j \in \{0, 1, \ldots, t+1\} - \{r\}$, set $\varphi_{g(d)}(\langle i, j\rangle + d + 1) = 0$; if $\varphi_{i,t}(\langle i, r\rangle + d + 1)$ is defined but $\varphi_{g(d)}(\langle i, r\rangle + d + 1)$ is not defined, then set $\varphi_{g(d)}(\langle i, r\rangle + d + 1) = 1 - \varphi_{i,t}(\langle i, r\rangle + d + 1)$.
  Furthermore, update $a_{i,t+1} = \langle i, t+1\rangle + d + 1$ if and only if $r \leq t$ and $|\{0, 1, \ldots, r\} - W_{d,t}| < i$.
  Let $\varphi_{g(d),t+1}(x) = \varphi_{g(d),t}(x)$ for all x with $\varphi_{g(d),t}(x) \downarrow$.

It shall be shown that the partial-recursive function $\varphi_{g(d)}$ as defined above possesses the following properties:

1. If W_d is cofinite, then there is an i_0 for which the markers $a_{i,t}$ move infinitely often if and only if $i \geq i_0$, so that $W_{g(d)}$ is also cofinite.
2. If W_d is coinfinite, then the markers $a_{i,t}$ move only finitely often, and there is no total recursive function extending $\varphi_{g(d)}$.

Item 1. follows because if W_d is cofinite, and $|\overline{W}_d| = k$, then for all $i > k$ and each r, there is a t large enough so that $|\{0,1,\ldots,r\} - W_{d,t}| < i$. This means that for all $i > k$, the markers $a_{i,t}$ move infinitely often. Moreover, this implies that $W_{g(d)}$ is cofinite, for each stage ensures that $\varphi_{g(d)}$ is defined on all inputs $\langle i,j\rangle + d + 1$ for which $j < r$, and since $a_{i,t}$ is shifted to $\langle i,r\rangle + d + 1$ for arbitrarily large values of r for all $i > k$, $\varphi_{g(d)}$ eventually becomes defined on all inputs $\langle i,j\rangle + d + 1$ for $i > k$ and $j \in \mathbb{N}$. For $i \leq k$, suppose that the markers $a_0, a_1, \ldots, a_k$ settle down permanently on the values $\langle 0,r_0\rangle + d + 1, \langle 1,r_1\rangle + d + 1, \ldots, \langle k,r_k\rangle + d + 1$ respectively; by the algorithm, while $\varphi_{g(d)}$ might remain undefined on some of these inputs, $\varphi_{g(d)}$ is, however, defined for all $\langle i,j\rangle + d + 1$ with $i \leq k$ and $j > r_i$. Thus $W_{g(d)}$ is indeed cofinite.

On the other hand, if W_d is coinfinite, then for each fixed i there are r, t sufficiently large so that $|\{0,1,\ldots,r\} - W_{d,t}| \geq i$. At stage $t+1$, each marker $a_i = \langle i,r\rangle + d + 1$ is updated to a new value $\langle i,t+1\rangle + d + 1$ with $t + 1 > r$ if $|\{0,1,\ldots,r\} - W_{d,t}| < i$; for this reason, there will eventually be a stage s at which $|\{0,1,\ldots,u\} - W_{d,s}| \geq i$, when $a_{i,s} = \langle i,u\rangle + d + 1$, and the inequality would continue to hold at all subsequent stages, in turn implying that the value of a_i will be permanently fixed as this value. Furthermore, if φ_i is a total function, then there will be a stage s' at which $\varphi_{i,s'}(\langle i,u\rangle + d + 1)$ is defined, and the algorithm would secure that $\varphi_{g(d)}(\langle i,u\rangle + d + 1)$ differs from the value of $\varphi_{i,s'}(\langle i,u\rangle + d + 1)$. Therefore there cannot be a total recursive function extending $\varphi_{g(d)}$.

Now let A be a PA-complete set which is low, that is, every partial-recursive $\{0,1\}$-valued function may be extended to an A-recursive function, and, in addition, $A'' \equiv_T \mathbb{K}'$. Jockusch and Soare [11] give a construction of such a set A. Furthermore, let $\varphi^A_{f(d)}$ be a uniformly A-recursive extension of the partial-recursive function $\varphi_{g(d)}$ such that $\varphi^A_{f(d)}$ is $\{0,1\}$-valued. There is a further recursive function h for which $W^A_{h(d,e)} = \{n :\ N$ outputs e at least n times on the text $0^{g(d)} \circ 1 \circ \varphi^A_{f(d)}(g(d)+2) \circ \varphi^A_{f(d)}(g(d)+3) \circ \ldots\}$. Owing to the confidence of N, one can determine by means of the oracle A'' the unique e such that $W^A_{h(d,e)}$ is infinite.

If W_d is cofinite, then, as was shown above, $\varphi_{g(d)}$ is also cofinite, and so $\varphi^A_{f(d)}$ is a total recursive extension of $\varphi_{g(d)}$, that is, $\varphi_{g(d)} =^* \varphi^A_{f(d)}$. Therefore N learns the recursive function generating the text $0^{g(d)} \circ 1 \circ \varphi^A_{f(d)}(g(d)+2) \circ \varphi^A_{f(d)}(g(d)+3) \circ \ldots$, and consequently $\varphi_e(x) = \varphi^A_{f(d)}(x)$ for all $x \geq g(d)+2$.

However, if W_d is coinfinite, it follows from the construction of $\varphi_{g(d)}$ that there is no total recursive function extending $\varphi_{g(d)}$, giving that $\varphi_e \neq \varphi^A_{f(d)}$, or more specifically, there is an $x \geq g(d) + 2$ such that either $\varphi_e(x) \uparrow$ or $\varphi_e(x) \downarrow \neq \varphi^A_{f(d)}(x) \downarrow$.

Hence W_d is cofinite if and only if for all $x \geq g(d)+2$, $\varphi_e(x) \downarrow = \varphi^A_{f(d)}(x) \downarrow$. As this condition may be checked using the oracle A'', and A'' is Turing equivalent to $\mathbb{K}'$, it may be concluded that $\{d : W_d \text{ is cofinite}\} \equiv_T \mathbb{K}'$, which is the desired contradiction. Therefore the class $\mathcal{C}$ cannot be confidently partially learnt. ▮

The following theorem formulates a criterion that may appear at first sight to be less stringent than confident partial learnability, but is in fact equivalent to it. The proof illustrates a padding technique, dependent on the underlying hypothesis space of the learner, that is often applied throughout this work to construct confident partial learners.

Theorem 5. *A class $\mathcal{C}$ of recursive functions is confidently partially learnable if and only if there is a recursive learner M such that*

- *M outputs on each text exactly one index infinitely often;*
- *if T is the canonical text for a recursive function f in $\mathcal{C}$ and d is the index output by M infinitely often on T, then there is an index e of f with $e \leq d$.*

Proof. Suppose that there is a recursive learner M of $\mathcal{C}$ which satisfies the learning criteria laid out in the statement of the theorem. Let $pad(e, d)$ be a two-place recursive function such that $\varphi_{pad(e,d)} = \varphi_e$ and $pad(e, d) \neq pad(e', d')$ if $(e, d) \neq (e', d')$ for all numbers e, d, e', d'. One may define a learner N which confidently partially learns $\mathcal{C}$ as follows: on the input text $T = f(0) \circ f(1) \circ \ldots \circ f(n) \circ \ldots$, N outputs $pad(e, d)$ at least n times if N has output d many indices of the form $pad(e', d')$ with $e' < e$ among its first n hypotheses and either $\varphi_e(x) \downarrow = f(x)$ for all $x < n$ or M has output e at least n times.

For the verification, assume that e is the least index such that either M outputs e infinitely often or $f = \varphi_e$. Consider $e' < e$ and the least $d_{e'}$ such that $\varphi_{e'}(x)$ differs from $f(x)$ for some $x < d_{e'}$ and M does not output e' $d_{e'}$ times. Then N will also at most $d_{e'}$ times output an index of the form $pad(e', d')$. Furthermore, let d be the number of times an index of the form $pad(e', d')$ with $e' < e \wedge d' \in \mathbb{N}$ is output by N. Then N will output $pad(e, d)$ infinitely often and that is the only index output infinitely often by N when processing f.

For the converse direction, any given confident partial learner of $\mathcal{C}$ clearly satisfies the conditions on M given in the statement of this theorem. ▮

Definition 6. A class is Ex^1-learnable iff there is a learner M which converges on the text of a function f in this class to an index e such that, for all but at most one x, $\varphi_e(x){\downarrow} = f(x)$.

Theorem 7. *Every Ex^1-learnable class is confidently partially learnable.*

Proof. Assume that M is an Ex^1-learner for a class S, where, without loss of generality, $M(\sigma \circ \tau) \geq M(\sigma)$ for all σ, τ. Furthermore let *patch* be a recursive function with $\varphi_{patch(e,x,y)}(x) = y$ and $\varphi_{patch(e,x,y)}(z) = \varphi_e(z)$ for all $z \neq x$; without loss of generality, *patch* is a one-one function. Now one constructs a new confident partial learner N as follows:

- N outputs $\mathit{patch}(e, x, f(x))$ at least n times if M has at least n times output the index e and $\varphi_e(x)$ does not output $f(x)$ within n steps while $\varphi_e(z)$ is defined and equal to $f(z)$ for all $z < x$;
- N outputs $\mathit{patch}(e, 0, f(0))$ at least n times if M has at least n times output the index e and $\varphi_e(z)$ is defined and equal to $f(z)$ for all $z < n$;
- N outputs $\mathit{patch}(0, 0, 0)$ at least n times if M makes on f at least n mind changes.

One can see the following: If M diverges on f then N outputs the hypothesis $\mathit{patch}(0, 0, 0)$ infinitely often and all other hypotheses only finitely often. If M converges to a correct index e on f then N outputs $\mathit{patch}(e, 0, f(0))$ infinitely often and all other indices only finitely often. If M converges on f to an index e which differs on at least one value from f by either being undefined or being wrong then N outputs $\mathit{patch}(e, x, f(x))$ infinitely often where x is the least number where φ_e is either undefined or different from f. This shows that N is a confident partial learner for the given class. ∎

Remark 8. The preceding result is a generalisation of the statement that every explanatorily learnable class is confidently learnable. Indeed, note that the class $\mathcal{C} = \{f : f(0)$ is an index for f which is correct at all but at most one inputs$\}$ is Ex^1-learnable and behaviourally correctly learnable but not explanatorily learnable. One could easily generalise the result such that one shows that every Ex^a-learnable class where the learner converges to an index with at most a errors is confidently partially learnable where $a \in \{0, 1, 2, \ldots\}$ is a fixed constant. The more general criterion of Ex^*-learnable classes is not covered by confident partial learning as the class from Theorem 4 shows.

It is quite a curious feature of confident learning under various success criteria that it is closed under finite unions. In particular, it is known that the union of finitely many confidently vacillatorily learnable classes is also confidently vacillatorily learnable; the analogous result for confident behaviourally correct learning also holds true. The next theorem states that this property of confident learning even extends to partial learnability. That is to say, if $\mathcal{C}_1$ and $\mathcal{C}_2$ are confidently partially learnable classes of recursive functions, then $\mathcal{C}_1 \cup \mathcal{C}_2$ is also confidently partially learnable.

Theorem 9. *Confident partial learning is closed under finite unions; that is, if $\mathcal{C}_1$ and $\mathcal{C}_2$ are confidently partially learnable classes, then $\mathcal{C}_1 \cup \mathcal{C}_2$ is confidently partially learnable.*

Proof. Let M and N be confident partial learners of the classes $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively. Now using Theorem 5, one can consturct a new learner R which outputs $\langle i, j\rangle$ at least n times iff M outputs i and N outputs j at least n times. It is directly obvious that on every text of a function, the learner R outputs exactly one index $\langle i, j\rangle$ infinitely often; this index is an upper bound of an index e of the function to be learnt whenever $i \geq e \vee j \geq e$. Hence R is a confident partial learner (in the sense of Theorem 5) of $\mathcal{C}_1 \cup \mathcal{C}_2$. ∎

Corollary 10. *There is a confidently partially learnable class which is not behaviourally correctly learnble.*

Proof. Blum and Blum's Non-Union Theorem [3] provides classes $\mathcal{C}_1$ and $\mathcal{C}_2$ which are explanatory learnable while their union is not behaviourally correctly learnable. By Theorem 7 the two classes are confidently partially learnable and by Theorem 9 their union $\mathcal{C}_1 \cup \mathcal{C}_2$ is confidently partially learnable as well. ∎

Theorem 4 demonstrates that the class of all total recursive functions is not confidently partially learnable. Nonetheless, there is a less restrictive notion of confident partial learning, somewhat analogous to a blend of behaviourally correct learning and partial learning, that permits the class of all recursive functions to be learnt. This notion of learning is spelt out in the following theorem.

Theorem 11. *There is a recursive learner M such that on every function f there is exactly one partial-recursive function Ψ for which M outputs an index infinitely often, and $f = \Psi$ whenever f is recursive.*

Proof. The learner M works in stages n which are executed in parallel (as some simulations might provide additional indices which have to be taken into account on a stage): M first searches for the first e_n found such that for all $m < n$ it holds that $e_n \geq e_m$ and $\varphi_{e_n}(x) \downarrow = f(x)$ for all $x < n$. From then onwards, the learner searches all $d \leq e_n$ with $\forall x < n\,[\varphi_d(x) \downarrow = f(x)]$; for each such d it outputs the index d itself and a further index $h(d, n, f(n))$ where $\varphi_{h(d,n,f(n))} = \varphi_c$ for the first $c \leq d$ found such that $\forall x \leq n\,[\varphi_c(x) \downarrow = f(x)]$; if such a c does not exist then $\varphi_{h(d,n,f(n))}$ is everywhere undefined.

In the case that e is the least index of f, it follows that M outputs only finitely often an index of the type d or $h(d, n, f(n))$ with $d < e$. M will infnitely often output e. Furthermore, for almost all n, each index of the form $h(d, n, f(n))$ output by M satisfies that $d \geq e$ and that therefore $\varphi_{h(d,n,f(n))} = \varphi_c$ for some φ_c extending $f(0) \circ f(1) \circ \ldots \circ f(n)$. Also, the indices of the form d with $d \neq e$ issued at stage n satisfy that φ_d coincides with f strictly below n. Therefore, the learner issues for each partial function different from f only finitely often an index.

In the case that f is not recursive, then the sequence $e_0, e_1, \ldots$ is increasing and unbounded. For each e_m there is a maximal $n > m$ such that M outputs an index $h(d, n, f(n))$ with $d \leq e_m$. Then $\varphi_{h(d,n,f(n))}$ is the everywhere undefined function, as there is no φ_c with $c \leq n$ such that φ_c extends $f(0) \circ f(1) \circ \ldots \circ f(n)$. Hence M outputs infinitely often an index of the everywhere undefined function. Furthermore, there is no other partial function for which M infinitely often outputs an index: whenever M outputs an index for it at stage n then the corresponding partial function is defined and equal to $f(x)$ at every input $x < n$; as no partial-recursive function coincides with f, M only finitely often outputs an index of that partial function. This completes the proof. ∎

The remainder of the present section is devoted to the study of confident partial learning relative to oracles. As a first step towards characterising the Turing degrees of oracles relative to which all recursive functions can be confidently partially learnt, one may observe that the proof of Theorem 4 produces the following corollary.

Theorem 12. *There is a behaviourally correctly learnable class $\mathcal{C} \subseteq REC_{0,1}$ such that $\mathcal{C}$ is confidently partially learnable relative to B only if $B'' \geq_T \mathbb{K}''$.*

The next lemma, in whose proof the padding property of the default hypothesis space $\{\varphi_0, \varphi_1, \varphi_2, \ldots\}$ is pivotal, will be applied in the subsequent theorem.

Lemma 13. *For every A''-recursive function $F^{A''}$, there is an A-recursive function f^A such that for all numbers d, if $F^{A''}(d) = e$, then there is a unique number e' for which there are infinitely many t with $f^A(d,t) = e'$ and $\varphi_e = \varphi_{e'}$.*

Proof. Given that $F^{A''} \leq_T A''$, there exists a sequence of A-recursive approximations $\{f_{i,j}\}_{i,j \in \mathbb{N}}$ such that for all numbers e, $\exists i \forall i' \geq i \exists j \forall j' \geq j [f_{i,j}(e) = F^{A''}(e)]$ holds. One may define an A-recursive function G which satisfies $G(e,t) = pad(e', \langle i, s \rangle)$ for for some i, s and infinitely many t iff $F^{A''}(e) = e'$.

First, let $a_{e,0}, a_{e,1}, a_{e,2}, \ldots$ be an A-recursive sequence in which $pad(d,i)$ occurs at least n times if and only if for all $i' \in \{i, i+1, \ldots, i+n\}$, there are n numbers j' such that $f_{i',j'}(e) = d$. This condition ensures that $pad(d,i)$ occurs in $a_{e,0}, a_{e,1}, a_{e,2}, \ldots$ infinitely often for some i if and only if $d = F^{A''}(e)$; however, the i is not unique and there might be $i' > i$ such that also $pad(d,i')$ also occurs infinitely often in the sequence.

Second, let $a'_{e,0}, a'_{e,1}, a'_{e,2}, \ldots$ be a further A'-recursive sequence in which $pad(d, \langle i, s \rangle)$ occurs n times if and only if there is a stage $t \geq s$ such that there are n numbers $u \leq s$ with $a_{e,u} = pad(d,i)$ and s is the least number with $pad(d,i') \notin \{a_{e,s}, a_{e,s+1}, \ldots, a_{e,t}\}$ for all $i' < i$.

Subsequently, one may produce the two-valued A-recursive function G by setting $G(e,t) = a'_{e,t}$ for all such sequences $a'_{e,0}, a'_{e,1}, a'_{e,2}, \ldots$ constructed for each e. By the above construction, the A-recursive function G satisfies the condition that for all e, there is exactly one index e' with $G(e,t) = e'$ for infinitely many t and this e' is of the form $pad(F^{A''}(e), \langle i, s \rangle)$ for some i, s. This establishes the lemma. ∎

Having established a necessary condition on the computational power of confident learners that can learn REC, one may hope for an analogous sufficient condition. By means of the above lemma, the theorem below proposes several oracle conditions that, when taken together, enable REC to be confidently partially learnt.

Theorem 14. *If B is low, PA-complete and $A \geq_T B$, $A'' \geq_T \mathbb{K}''$, then there is an A-recursive confident partial learner for REC.*

Proof. The class of all recursive $\{0,1\}$-valued functions, $REC_{0,1}$, is explanatorily learnable by a learner M which outputs B-recursive indices. First, one may construct a numbering $\{\varphi^B_{h(0)}, \varphi^B_{h(1)}, \ldots\}$ of $\{0,1\}$-valued B-recursive functions such that $REC_{0,1} \subset \{\varphi^B_{h(0)}, \varphi^B_{h(1)}, \ldots\}$, and for all e and each input x,

$$\varphi^B_{h(e)}(x) = \begin{cases} 0 \text{ if } \varphi_e(x) \downarrow = 0; \\ 1 \text{ if } \varphi_e(x) \downarrow > 0; \end{cases}$$

as B is PA-complete, there is a B-recursive function g such that each partial B-recursive function $\varphi^B_{h(e)}$ may be extended to a total $\{0,1\}$-valued function $\varphi^B_{g(e)}$. Without loss of generality, assume that $g(d_k) \geq d_k$. There is an explanatory learner which conjectures on the input $f(0) \circ f(1) \circ \ldots \circ f(n)$ the index $g(e)$ for the least e with $\varphi^B_{g(e)}(x) = f(x)$ for all $x \leq n$; let M be an equivalent B-recursive confident partial learner and let $g(d_0), g(d_1), g(d_2), \ldots$ be the hypotheses issued by M when it is learning some $f \in REC_{0,1}$. Define the B'''-recursive function $F^{B'''}$ by

$$F^{B'''}(g(d_k)) = \begin{cases} e \text{ if } e \text{ is the minimal index with } \varphi_e = \varphi^B_{g(d_k)}; \\ 0 \text{ if there is no index } e \text{ with } \varphi_e = \varphi^B_{g(d_k)}. \end{cases}$$

Furthermore, since $B''' \leq_T A''$ by assumption, it follows that $F^{B'''} = F^{A''}$. One can now define a confident partial A-recursive learner N: by Lemma 13, there is an A-recursive function $f^A(d,t)$ such $f^A(d,t)$ outputs a unique index e' with $\varphi_{e'} = \varphi_{F^{A''}(d)}$ for infinitely many t. N is then set to output $pad(f^A(g(d_k),t), g(d_k))$ if and only if M outputs $g(d_k)$ for the t-th time.

One can further generalise this result to construct a learner P that confidently partially learns REC relative to A: This learner P would translate a text of f into a text of the graph of f and then simulate the learner for $REC_{0,1}$ and retranslate every hypothesis for a graph into a hypothesis for the function itself. ∎

The condition that the double jump of the oracle be Turing above $\mathbb{K}''$ is not, however, sufficient for confidently partially learning REC, as the following theorem demonstrates.

Theorem 15. *There is a set A with $A'' \geq_T \mathbb{K}''$ such that A is 2-generic and $REC_{0,1}$ is not confidently partially learnable relative to A.*

5 Essentially Consistent Partial Learning

The present section considers a weakened form of consistency in partial learning, namely, *essential class consistency*. Under this learning requirement, the learner is permitted to be inconsistent on finitely many segments of the canonical text for some recursive function in the class to be learnt. Before developing this notion, we shall first review the more restrictive type of *consistent* learning, and attempt to compare it with confident partial learning.

Definition 16 (Bārzdiņš [2]). A recursive learner M is said to be *consistent* on a text T if and only if for all $n \in \mathbb{N}$, $M(T[n]) \downarrow$ and $\varphi_{M(T[n])}(x) \downarrow = T(x)$ whenever $x \leq n$. A learner is said to be consistent on a function f just if it is consistent on the canonical text for f. A learner is said to *class consistently partially* learn $\mathcal{C}$ if and only if it partially learns $\mathcal{C}$ and is consistent on each f in $\mathcal{C}$.

Whilst class consistency may appear to be a fairly restrictive learning constraint, the following theorem implies that it cannot in general guarantee that a class of recursive functions is confidently partially learnable.

Theorem 17. *There is a class of recursive functions which is class consistently partially learnable but not confidently partially learnable.*

Proof. The following example essentially modifies the construction of the programme $g(d)$ in Theorem 4 so that a subclass of $\mathcal{C}$ may be class consistently partially learnable. For each number d, let $g(d)$ be a programme for a partial-recursive function $\varphi_{g(d)}$ which is defined as follows.

- Set $\varphi_{g(d),s}(0) = d$ for all s.
- Initialize the markers $a_0, a_1, a_2, \ldots$ by setting $a_{i,0} = \langle i, 0\rangle + 1$ for $i \in \mathbb{N}$.
- At stage $s+1$, consider each marker $a_{i,s} = \langle i, r\rangle + 1$ such that $a_{i,s} \leq s+1$, and execute the following instructions in succession. Set $\varphi_{g(d),s+1}(x) = 0$ for all $x = \langle i, j\rangle + 1 \leq s+1$ such that $j \neq r$ if $\varphi_{g(d),s}$ is not already defined on x. Next, check whether $\varphi_{i,s+1}(a_{i,s}) \downarrow \in \{0, 1\}$ holds; if so, let $\varphi_{g(d),s+1}(a_{i,s}) = 1 - \varphi_{i,s+1}(a_{i,s})$ if $\varphi_{g(d)}$ is not already defined on the input $a_{i,s}$. Now, for each i such that $\langle i, m\rangle + 1 \leq s+1$ for some m, let $u = \max(\{m : \langle i, m\rangle + 1 \leq s+1\})$. Associate the marker $a_{i,s+1}$ with $\langle i, u+1\rangle + 1$ if at least one of the following two conditions applies; otherwise, let $a_{i,s+1} = a_{i,s}$.
  1. There is a $j < i$ with $\langle j, m\rangle + 1 \leq s+1$ for some m such that $a_{j,s+1} \neq a_{j,s}$.
  2. If $a_{i,s} = \langle i, r\rangle + 1$, then the inequality $|\{0, 1, \ldots, r\} - W_{d,s+1}| < i$ holds.

Let $\mathcal{C} = \{f : W_d \text{ is cofinite } \wedge f \text{ is a total recursive extension of } \varphi_{g(d)}\}$. One may prove that $\mathcal{C}$ is class consistently partially learnable, and an argument exactly analogous to that in Theorem 4 shows that it is not confidently partially learnable. ∎

Essentially class-consistent learners can finitely often be inconsistent with the input text; in partial learning, this consistency requirement is still a proper restriction. The following result establishes a connection between the learning success criteria of semantic convergence in the limit and essentially class consistent partial convergence; it suggests that there may be ample examples of essentially class consistently partially learnable classes of recursive functions.

Theorem 18. *Every behaviourally correctly learnable class of recursive functions is essentially class consistently partially learnable.*

Remark 19. Jain and Stephan [10, Theorem 15] constructed a consistently partially learnable class of recursive functions which is not behaviourally correctly learnable. It follows that the converse of the preceding theorem does not hold in general. Furthermore, they showed [10, Theorem 19] that there are classes which are explanatorily learnable with at most one mind change as well as class consistently explanatorily learnable by a partial-recursive learner, but nonetheless cannot be class consistently partially learnt on canonical text. Consequently, Theorem 18 is no longer true if one replaces essential class consistency with general class consistency in the conclusion, and so this watered-down variant of consistency is indeed a more general learning notion than ordinary consistency.

To ascertain that essential class consistency constitutes a real learning constraint, one can show that the class of all $\{0, 1\}$-valued recursive functions is not partially learnable under this criterion.

Theorem 20. *The class $REC_{0,1}$ is not essentially class consistently partially learnable.*

One can take the above result one step further and construct an example of a confidently partially learnable class of recursive functions which is not essentially class consistently partially learnable.

Theorem 21. *There is a class of recursive functions which is confidently partially learnable but not essentially class consistently partially learnable.*

Proof. Let $M_0, M_1, M_2, \ldots$ be a recursive enumeration of all partial-recursive learners.

For each M_e define a function $\varphi_{g(e)}$ by starting with $\sigma_{e,0} = e$ and taking $\sigma_{e,k+1}$ to be the first extension of $\sigma_{e,k}$ found such that $M_e(\sigma_{e,k+1})$ outputs an index d with $\varphi_d(x) \downarrow\neq \sigma_{e,k+1}(x)$ for some $x < |\sigma_{e,k+1}|$. $\varphi_{g(e)}(x)$ takes as value $\sigma_{e,k}(x)$ for the first k found where this is defined.

Furthermore, for each e, k where $\sigma_{e,k}$ is defined, let $\varphi_{h(e,k)}$ be the partial recursive function ψ extending $\sigma_{e,k}$ such that for all $x \geq |\sigma_{e,k}|$, $\psi(x)$ is the least a such that either $M_e(\psi(0)\psi(1)\ldots\psi(x-1)a) > x$ or $M_e(\psi(0)\psi(1)\ldots\psi(x-1)a) = M_e(\psi(0)\psi(1)\ldots\psi(x-1)b)$ for some $b < a$.

Let $\mathcal{C}_1$ contain all those $\varphi_{g(e)}$ which are total and $\mathcal{C}_2$ contain all $\varphi_{h(e,k)}$ where M_e is total and $\varphi_{g(e)} = \sigma_{e,k}$, that is, the construction got stuck at stage k. The class $\mathcal{C}_1$ is obviously explanatorily learnable; for the class $\mathcal{C}_2$, an explanatory learner identifies first the e and then simulates the construction of $\varphi_{g(e)}$ and updates the hypothesis always to $h(e,k)$ for the largest k such that $\sigma_{e,k}$ has already been found. Hence both classes are explanatorily learnable, hence their union $\mathcal{C}$ is confidently partially learnable.

However $\mathcal{C}$ is not essentially class consistently partially learnable, as it is now shown. So consider a total learner M_e. If $\varphi_{g(e)}$ is total then M_e is inconsistent on this function infinitely often and so M_e does not essentially class consistently partially learn $\mathcal{C}$. So consider the k with $\varphi_{g(e)} = \sigma_{e,k}$. Note that the inductive definition of $\varphi_{h(e,k)}$ results in a total function. If M_e outputs on $\varphi_{h(e,k)}$ each index only finitely often, then M_e does not partially learn $\varphi_{h(e,k)}$. If M_e outputs an index d infinitely often, then for all sufficiently long $\tau a \preceq \varphi_{h(e,k)}$ with $M_e(\tau a) = d$ it holds that there is a $b < a$ with $M(\tau b) = d$ as well. By assumption, $\sigma_{e,k+1}$ does not exist and can be neither τa nor τb. Hence τa is not extended by φ_d and so M_e outputs an inconsistent index for almost all times where it conjectures d; again M_e does not essentially class consistently partially learn $\mathcal{C}$. ∎

As a consequence of the preceding theorem, one has the corollary that essentially class consistent partial learning is not closed under finite unions.

Corollary 22. *Essentially class consistent learning is not closed under finite unions; that is, there are essentially class consistently partially learnable classes $\mathcal{C}_1, \mathcal{C}_2$, such that $\mathcal{C}_1 \cup \mathcal{C}_2$ is not essentially class consistently partially learnable.*

A complete characterisation of the classes of recursive functions which are consistently partially learnable relative to an oracle A, classified according to whether

A has hyperimmune or hyperimmune-free Turing degree, was obtained in [10]. The theorem below asserts that a recursive learner with access to a PA-complete oracle may essentially class consistently partially learn REC. Since the class of hyperimmune-free, PA-complete degrees is nonempty, as demonstrated in [11], one may conclude that for partial learning, essential class consistency is indeed a weaker criterion than general consistency, even when learning with oracles. The proof utilises the fact that there is a one-one numbering of all recursive functions plus all functions of finite domain.

Theorem 23. *If A is a PA-complete set, then REC is essentially class consistently partially learnable using A as an oracle.*

Proof. Let $\psi_0, \psi_1, \psi_2, \ldots$ be a one-one numbering of the recursive functions plus the functions with finite domain. For example, Kummer [13] provides such a numbering. Let g be a recursive function such that $\psi_e = \varphi_{g(e)}$ for all e. There is a recursive sequence $(e_0, x_0, y_0), (e_1, x_1, y_1), \ldots$ of pairwise distinct triples such that $\psi_e(x) \downarrow = y$ iff the triple (e, x, y) appears in this sequence.

On input $\sigma = f(0) \circ f(1) \circ \ldots \circ f(n)$, the learner M searches for the first $s \geq n$ such that for all $t \leq s$ either $e_t \neq e_s$ or $x_t > n$ or $y_t = f(x_t)$; that is, s is the first stage where ψ_{e_s} — to the extent it can be judged from the triples enumerated until stage s — is consistent with σ. Then M determines using the PA-complete oracle an $d \leq e_s$ such that either ψ_d extends σ or there is no $c \leq e_s$ such that ψ_c extends σ; note that in that second case the oracle can provide "any false d" below e. The learner conjectures then $g(d)$ for the index d determined this way.

If now e is the unique ψ-index of the function f to be learnt, then for all sufficiently long inputs σ, the above e_s satisfies $e_s \geq e$ as for each $d < e$ either there are only finitely many triples having d in the first component with all of them appearing before n or there is a $t \leq n$ with $e_t = d \wedge x_t \leq n \wedge y_t \neq f(x_t)$. Hence, the s selected satisfies $e_s \geq e$ and therefore the d provided satisfies that ψ_d extends σ. Furthermore, there are infinitely many n with $e_n = e$ and for those the choice is $s = n$ and, if n is sufficiently large, $d = e$. Hence the learner outputs infinitely often $g(e)$ and almost always an index $g(d)$ with $\varphi_{g(d)}$ being consistent with the input seen so far. ∎

6 Conclusion

In conclusion, confident partial learning appears to be a fairly robust learning notion that is neither too restrictive nor too powerful. Essentially class consistent partial learning may be a more balanced criterion compared to global consistency, for there is quite a rich collection of essentially class consistently partially learnable classes of recursive functions, which includes all classes that are behaviourally correctly learnable. Though the results on these two notions are quite complete, there is still potential for further work on characterising the omniscient degrees of inference for confident partial learning and essentially class consistent partial learning.

References

[1] Bārzdiņš, J.: Two theorems on the limiting synthesis of functions. In: Theory of Algorithms and Programs, vol. 1, pp. 82–88. Latvian State University (1974) (in Russian)

[2] Bārzdiņš, J.: Inductive inference of automata, functions and programs. In: Proceedings of the 20th International Congress of Mathematicians, Vancouver, pp. 455–560 (1974) (in Russian); English translation in American Mathematical Society Translations: Series 2 109, 107–112 (1977)

[3] Blum, L., Blum, M.: Towards a mathematical theory of inductive inference. Information and Control 28, 125–155 (1975)

[4] Feldman, J.: Some decidability results on grammatical inference and complexity. Information and Control 20, 244–262 (1972)

[5] Gao, Z., Stephan, F., Wu, G., Yamamoto, A.: Learning Families of Closed Sets in Matroids. In: Dinneen, M.J., Khoussainov, B., Nies, A. (eds.) WTCS 2012 (Calude Festschrift). LNCS, vol. 7160, pp. 120–139. Springer, Heidelberg (2012)

[6] Mark Gold, E.: Language identification in the limit. Information and Control 10, 447–474 (1967)

[7] Grieser, G.: Reflective inductive inference of recursive functions. Theoretical Computer Science A 397(1-3), 57–69 (2008)

[8] Hanf, W.: The Boolean algebra of logic. Bulletin of the American Mathematical Society 81, 587–589 (1975)

[9] Jain, S., Osherson, D., Royer, J.S., Sharma, A.: Systems that learn: an introduction to learning theory. MIT Press, Cambridge (1999)

[10] Jain, S., Stephan, F.: Consistent partial identification. In: COLT 2009, pp. 135–145 (2009)

[11] Jockusch Jr., C.G., Soare, R.I.: $\prod_1^0$ classes and degrees of theories. Transactions of the American Mathematical Society 173, 33–56 (1972)

[12] Lakatos, I.: The role of crucial experiments in science. Studies in History and Philosophy of Science 4(4), 319–320 (1974)

[13] Kummer, M.: Numberings of $\mathcal{R}_1 \cup F$. In: Börger, E., Kleine Büning, H., Richter, M.M. (eds.) CSL 1988. LNCS, vol. 385, pp. 166–186. Springer, Heidelberg (1989)

[14] Osherson, D.N., Stob, M., Weinstein, S.: Systems that learn: an introduction to learning theory for cognitive and computer scientists. MIT Press, Cambridge (1986)

[15] Popper, K.R.: The logic of scientific discovery. Hutchinson, London (1959)

[16] Rogers Jr., H.: Theory of recursive functions and effective computability. MIT Press, Cambridge (1987)

[17] Wiehagen, R., Zeugmann, T.: Learning and Consistency. In: Lange, S., Jantke, K.P. (eds.) GOSLER 1994. LNCS (LNAI), vol. 961, pp. 1–24. Springer, Heidelberg (1995)

[18] Zeugmann, T., Zilles, S.: Learning recursive functions: a survey. Theoretical Computer Science 397(1-3), 4–56 (2008)

Automatic Learning from Positive Data and Negative Counterexamples

Sanjay Jain[1,⋆] and Efim Kinber[2]

[1] School of Computing, National University of Singapore, Singapore 117417
sanjay@comp.nus.edu.sg
[2] Department of Computer Science, Sacred Heart University, Fairfield, CT 06825–1000, U.S.A.
kinbere@sacredheart.edu

Abstract. We introduce and study a model for learning in the limit by finite automata from positive data and negative counterexamples. The focus is on learning classes of languages with a membership problem computable by finite automata (so-called automatic classes). We show that, within the framework of our model, finite automata (automatic learners) can learn all automatic classes when memory of a learner is restricted by the size of the longest datum seen so far. We also study capabilities of automatic learners in our model with other restrictions on the memory and how the choice of negative counterexamples (arbitrary, or least, or the ones whose size is bounded by the longest positive datum seen so far) can impact automatic learnability.

1 Introduction

In the paper [JLS10], the authors introduced an "automatic" variant of the well-known Gold's model for learning in the limit from positive data: the family of target languages is computable by a finite automaton (automatic family), and a learner is a finite automaton itself (automatic learning). More specifically, a family of target languages is defined by a regular index set, and the membership problem in these languages is regular in the sense that one finite automaton recognizes a combination (so called "convolution") of an index and a word if and only if the word is in the language defined by the index. They also considered three different natural types of limits on the size of the (long-term) memory available to the learner before outputting the next conjecture: (a) memory is bounded by the size of the longest positive input datum seen so far (plus a constant); (b) memory is bounded by the size of the current hypothesis (plus a constant); (c) the learner can store in the memory the last hypothesis only.

The authors of [JLS10] established that automatic learners are much weaker than unrestricted recursive learners — even when learning automatic classes. In particular, not every automatic class is automatically learnable. Moreover, they showed the following modification of D. Angluin's result from [Ang80]:

[⋆] Supported in part by NUS grant number C-252-000-087-001.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 66–80, 2012.

An automatic class is learnable by a recursive learner iff it satisfies Angluin's tell-tale condition. The authors of [JLS10] also obtained a number of interesting results showing differences between automatic learners on automatic classes with different limitations on the memory mentioned above, as well as impact on the learners of the requirement of *consistency* — when every conjecture must be consistent with the input seen so far.

Since automatic learners are not able to learn many automatic classes from positive data alone, it is natural to ask: under which conditions *all* automatic classes of languages are automatically learnable? In [JK08], the authors introduced and motivated a notion of learning languages in the limit from full positive data and a finite number of negative counterexamples provided to the learner whenever it's hypothesis contains data that is not a part of the target language. This approach to learning in the limit arguably is more natural than learning just from positive examples — for instance, children learning languages get corrected when using wrong words [HPTS84] (yet, as is probably the case in natural learning processes, in this model of learning, the learner does not get *all* negative examples). In a sense, this model combines two different and popular approaches to learning in the limit — learning languages from positive data and learning concepts from subset queries and counterexamples ([Ang88]), whereas none of these two approaches by itself adequately represents the process of language acquisition. In [JK08], the authors considered three different types of negative counterexamples provided to the learner: (a) arbitrary, (b) least, and (c) bounded by the size of the largest positive datum seen so far (the latter type is motivated by possible computational limitations on the "teacher" providing counterexamples). In this paper, we adapt the notion of learning with negative counterexamples to automatic learning of automatic classes. Our major result (Theorem 8) is that such automatic learners, even when required to be *iterative* (that is, whose memory stores just the last hypothesis), can learn *every* automatic class! On the other hand, interestingly, we have not been able to make such learners consistent with data seen so far. Yet, Theorem 9 shows that consistency can be achieved if the learners always receive the least negative counterexamples. On the other hand, as it follows from a result in [JLS10], there are automatic classes that cannot be learned even by non-automatic learners if the size of counterexamples is bounded by the size of the longest positive input datum seen so far (Theorem 11). Still, with this bound on the size of counterexamples, automatic learners with memory limited by the size of the longest positive input datum seen so far can learn automatic classes consisting only of infinite languages (Theorem 10).

We also show that some automatic classes cannot be learned automatically using bounded negative counterexamples with memory limited by the size of the current hypothesis (and, thus, when only the last hypothesis can be stored in the memory) — see Theorem 13, but can be learned automatically with memory limited by size of the longest positive datum seen so far even without negative counterexamples.

Theorem 14 shows the advantage of negative counterexamples, even for automatic iterative learners, compared to not having negative counterexamples, even for unrestricted recursive learners. Our last result (Theorem 15) shows that not every automatic class can be learned automatically and *monotonically* — that is, when every next conjecture includes the positive data covered by the previous one — even if the least negative counterexamples are provided.

A number of related problems remain open. In particular, we do not know whether every automatic family can be consistently learnt using arbitrary negative counterexamples. We also do not know whether iterative automatic learners or automatic learners with memory bounded by hypothesis size, receiving bounded negative counterexamples, can learn all automatic classes having only infinite languages. Some relations between various memory bounds on automatic learners using bounded negative counterexamples are also open.

2 Preliminaries

The set of natural numbers is denoted by N. Let Σ denote a finite alphabet. The set of all strings over the alphabet Σ is denoted by Σ^*. The empty string is denoted by ϵ. A string of length n is treated as a function from $\{0, 1, \ldots, n-1\}$ to Σ. Thus, $x = x(0)x(1), \ldots, x(n-1)$, where x is a string of length n. The length of a string x is denoted by $|x|$. We say that a string w is length-lexicographically smaller than string w' (written $w <_{ll} w'$) iff $|w| < |w'|$ or $|w| = |w'|$ and w is lexicographically below w' (where we assume some canonical ordering of elements of Σ). We let $w \leq_{ll} w'$ denote that either $w = w'$ or $w <_{ll} w'$. We let $\mathrm{succ}_S(w)$ denote the least w' such that $w <_{ll} w'$, and $w' \in S$ (if there is no such string, then we let $\mathrm{succ}_S(w)$ to be undefined).

We let $\emptyset, \subseteq$ and $\subset$ respectively denote emptyset, subset and proper subset. We let $\mathrm{card}(S)$ denote the cardinality of set S. When considering sets of natural numbers, we let $\max(S), \min(S)$ respectively denote the maximum and minimum of a set S, where $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$. When we are considering sets of strings S, we let $\max(S)$ and $\min(S)$ be the length-lexicographically largest and smallest string in S respectively, where if $S = \emptyset$, then we take $\max(S) = \#$ (where # is a special pause symbol, see below).

Convolution of two strings $x = x(0)x(1)\ldots x(n-1)$ and $y = y(0)y(1)\ldots y(m-1)$, denoted $conv(x, y)$, is defined as follows. Let x', y' be strings such that $x'(i) = x(i)$ for $i < n$, $x'(i) = \#$ for $n \leq i < \max(\{m, n\})$, $y'(i) = y(i)$ for $i < m$, and $y'(i) = \#$ for $m \leq i < \max(\{m, n\})$, where $\# \notin \Sigma^*$ is a special padding symbol. Thus, x', y' are obtained from x, y by padding the smaller string with #'s. Then, $conv(x, y) = z$, where $|z| = \max(\{m, n\})$ and $z(i) = (x'(i), y'(i))$, for $i < \max(\{m, n\})$. Note that z is a string over the alphabet $(\Sigma \cup \{\#\}) \times (\Sigma \cup \{\#\})$. Similarly, one can define convolution of more than two strings. Intuitively, giving a convolution of two strings to a machine means giving two strings in parallel, with the shorter string being padded with #s.

We say that an n-ary relation R is *automatic*, if $\{conv(x_1, x_2, \ldots, x_n) : (x_1, x_2, \ldots, x_n) \in R\}$ is regular. Similarly, an n-ary function f is automatic if $\{conv(x_1, x_2, \ldots, x_n, y) : f(x_1, x_2, \ldots, x_n) = y\}$ is regular.

A family of languages, $(L_\alpha)_{\alpha\in I}$ is said to be an *automatic family* if the index set I is regular and the set $\{conv(\alpha, x) : \alpha \in I, x \in L_\alpha\}$ is regular. Here the sets L_α are sets of strings over some finite alphabet. We often identify an automatic family $(L_\alpha)_{\alpha\in I}$ with the class $\mathcal{L} = \{L_\alpha : \alpha \in I\}$, where the indexing is implicit. We say that the automatic family $(L_\alpha)_{\alpha\in I}$ is 1–1 (or the indexing is 1–1), if for all $\alpha, \beta \in I$, $L_\alpha = L_\beta$ implies $\alpha = \beta$.

It can be shown that any family, relation or function that is first order definable using other automatic relations or functions is itself automatic.

Lemma 1 ([BG00], [KN95]). *Any relation that is first-order definable from existing automatic relations is automatic.*

We often implicitly use the above fact in our proofs. The present work considers learnability of automatic families in the presence of counterexamples. For this, let us consider a definition of a learner. This definition is given in a form slightly different from the one traditional in inductive inference. When there are no memory restrictions, this definition turns out to be essentially the same as the traditional definition. We use a different form to make it easier to consider automatic learners.

A *text* T is a mapping from N to $\Sigma^* \cup \{\#\}$. Here $\# \notin \Sigma^*$ denotes pauses in the presentation of data. We let $T[n]$ denote the initial sequence of T of length n, that is, $T[n] = T(0)T(1)\ldots T(n-1)$. The content of a text T, denoted content(T), is $\{T(i) : i \in N\} - \{\#\}$. Similarly, content$(T[n]) = \{T(i) : i < n\} - \{\#\}$. We let σ range over initial sequences of texts. We let Λ denote the empty sequence. We let $SEQ(S)$ denote the set of all finite sequences σ such that content$(\sigma) \subseteq S$. We let $\sigma\diamond\tau$ denote the concatenation of two sequences σ and τ. By abusing notation, for $x \in \Sigma^* \cup \{\#\}$, we use $\sigma\diamond x$ to denote the concatenation of σ with the sequence containing just one element x.

Definition 2. Suppose Σ, Δ are finite alphabets used for languages and memory of learners respectively, where $\# \notin \Sigma^*$. Suppose J is a regular index set (over some finite alphabet) for the hypothesis space used by the learner. Below ? is a special symbol not in J, which stands for "repeat the previous conjecture."

(a) A *learner* is a mapping from $\Delta^* \times (\Sigma^* \cup \{\#\})$ to $\Delta^* \times (J \cup \{?\})$. A learner has an initial memory $mem_0 \in \Delta^*$, and initial hypothesis $hyp_0 \in J \cup \{?\}$.

(b) Suppose a learner $\mathbf{M}$ with initial memory mem_0 and initial hypothesis hyp_0 is given. Below, σ is a sequence over $\Sigma^* \cup \{\#\}$ and $x \in \Sigma^* \cup \{\#\}$. We extend the definition of $\mathbf{M}$ to sequences by inductively defining
$\mathbf{M}(\Lambda) = (mem_0, hyp_0)$;
$\mathbf{M}(\sigma\diamond x) = \mathbf{M}(mem, x)$, where $\mathbf{M}(\sigma) = (mem, hyp)$, for some $hyp \in J \cup \{?\}$.
Additionally, for $|\sigma| \geq 1$, we inductively define $\mathbf{M}(mem, \sigma\diamond x) = \mathbf{M}(mem', x)$, where $\mathbf{M}(mem, \sigma) = (mem', hyp')$, for some $hyp' \in J \cup \{?\}$.

(c) We say that $\mathbf{M}$ converges on a text T to a hypothesis β (written: $\mathbf{M}(T)\downarrow_{hyp} = \beta$) iff there exists a t such that,
(i) $\mathbf{M}(T[t]) \in \Delta^* \times \{\beta\}$, and
(ii) for all $t' \geq t$, $\mathbf{M}(T[t]) \in \Delta^* \times \{\beta, ?\}$.

Intuitively, $\mathbf{M}(\sigma) = (mem, hyp)$ means that the memory and hypothesis of the learner $\mathbf{M}$ after having seen the sequence σ are *mem* and *hyp* respectively. We can think of a learner as receiving a text T for the language L, one element at a time. At each input, the learner updates its previous memory, and outputs a new conjecture (hypothesis), where ? denotes repeating the previous hypothesis. If the sequence of hypotheses converges to a grammar for L, then we say that the learner **TxtEx**-learns the language L from the text T (here **Ex** denotes "explains", and **Txt** denotes learning from text). Now we define learnability formally.

Definition 3. (Based on Gold [Gol67])

Suppose $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ is a target class, and $\mathcal{H} = \{H_\beta : \beta \in J\}$ is a hypothesis space, where both $\mathcal{L}$ and $\mathcal{H}$ are automatic families of languages.

(a) We say that $\mathbf{M}$ **TxtEx**-learns the language L (using hypothesis space $\mathcal{H}$) from a text T iff $\mathbf{M}(T)\downarrow_{hyp} = \beta$ such that $H_\beta = L$.
(b) We say that $\mathbf{M}$ **TxtEx**-learns a language L (using hypothesis space $\mathcal{H}$) iff $\mathbf{M}$ **TxtEx**-learns L from all texts for the language L (using hypothesis space $\mathcal{H}$).
(c) We say that $\mathbf{M}$ **TxtEx**-learns $\mathcal{L}$ (using hypothesis space $\mathcal{H}$) iff $\mathbf{M}$ **TxtEx**-learns all languages in $\mathcal{L}$ (using hypothesis space $\mathcal{H}$).
(d) $\mathbf{TxtEx} = \{\mathcal{L} : (\exists \mathbf{M})[\mathbf{M}\ \mathbf{TxtEx}\text{-learns } \mathcal{L} \text{ using some hypothesis space}]\}$.

We drop the reference to "using hypothesis space $\mathcal{H}$", when the hypothesis space is clear from the context. A hypothesis space $\mathcal{H}$ is said to be *class preserving* [LZ93] for learning a class $\mathcal{L}$ if $\mathcal{L} = \mathcal{H}$. A hypothesis space $\mathcal{H}$ is said to be *class comprising* [LZ93] for learning a class $\mathcal{L}$ if $\mathcal{L} \subseteq \mathcal{H}$.

Definition 4. Suppose a learner $\mathbf{M}$ using an automatic family $\mathcal{H} = \{H_\beta : \beta \in J\}$ as the hypothesis space is given.

(a) [JLS10] A learner $\mathbf{M}$ is called an *automatic learner* iff its graph is automatic. That is, $\{conv(mem, x, mem', hyp') : \mathbf{M}(mem, x) = (mem', hyp')\}$ is regular.
(b) [Wie76] $\mathbf{M}$ is said to be *iterative* iff, for all finite sequences σ, $\mathbf{M}(\sigma) = (mem, hyp)$ implies $mem = hyp$.
$\mathbf{M}$ is said to be *word-size* memory bounded iff there exists a constant c such that for all finite sequences σ, $\mathbf{M}(\sigma) = (mem, hyp)$ implies $|mem| \leq \max(\{|w| : w \in \text{content}(\sigma)\}) + c$.
$\mathbf{M}$ is said to be *hypothesis-size* memory bounded iff there exists a constant c such that for all finite sequences σ, $\mathbf{M}(\sigma) = (mem, hyp)$ implies $|mem| \leq |hyp| + c$.
(Note that if a learner is iterative then its memory is hypothesis-size bounded, but hypothesis-size bound on the memory does not imply that a learner is iterative.)
(c) [Bār74] $\mathbf{M}$ is said to be *consistent* iff, for all finite sequences σ, if $\mathbf{M}(\sigma) = (mem, hyp)$ with $hyp \neq ?$, then $\text{content}(\sigma) \subseteq H_{hyp}$.

(d) [Wie90] $\mathbf{M}$ is said to be *monotonic* iff for all texts T, for all $t < t'$, if $\mathbf{M}(T[t]) = (mem, hyp), \mathbf{M}(T[t']) = (mem', hyp')$, $hyp \neq ?$, $hyp' \neq ?$, then $\text{content}(T) \cap H_{hyp} \subseteq \text{content}(T) \cap H_{hyp'}$.

Note that the above constraints are required even on texts for languages outside the class $\mathcal{L}$. Note that when a learner gets positive data only, then a learner's conjecture may contain data that is not in the target language. In this situation, the learner may not be able to know that it went "beyond" the target language, as it does not receive any negative data. To address this issue, Jain and Kinber [JK08] considered the notion of learning with negative counterexamples. In this, for every hypothesis, a learner receives as input a negative counterexample, if there exists any. Thus, intuitively, the learner gets two input texts: one for positive data as above, and another for negative counterexamples.

Definition 5 (Based on [JK08]). Suppose Σ, Δ are finite alphabets used for languages and memory of learners respectively, where $\# \notin \Sigma^*$. Suppose J is a regular index set for the hypothesis space used by the learner.

(a) A *learner* learning using negative examples is a mapping from $\Delta^* \times (\Sigma^* \cup \{\#\}) \times (\Sigma^* \cup \{\#\})$ to $\Delta^* \times (J \cup \{?\})$.
A learner has an initial memory $mem_0 \in \Delta^*$, and initial hypothesis $hyp_0 \in J \cup \{?\}$.

(b) Suppose a learner $\mathbf{M}$ with initial memory mem_0 and initial hypothesis hyp_0 is given. We extend the definition of $\mathbf{M}$ to sequences as follows. Below, σ, τ are sequences over $\Sigma^* \cup \{\#\}$ with $|\sigma| = |\tau|$, and $x, y \in \Sigma^* \cup \{\#\}$.
$\mathbf{M}(\Lambda, \Lambda) = (mem_0, hyp_0)$;
$\mathbf{M}(\sigma \diamond x, \tau \diamond y) = \mathbf{M}(mem, x, y)$, where $\mathbf{M}(\sigma, \tau) = (mem, hyp)$, for some $hyp \in J \cup \{?\}$.
Additionally, for $|\sigma| = |\tau| \geq 1$, we inductively define $\mathbf{M}(mem, \sigma \diamond x, \tau \diamond y) = \mathbf{M}(mem', x, y)$, where $\mathbf{M}(mem, \sigma, \tau) = (mem', hyp')$, for some $hyp' \in J \cup \{?\}$.

(c) We say that $\mathbf{M}$ converges on text T with negative counterexample text T' to a hypothesis β (written: $\mathbf{M}(T, T')\downarrow_{hyp} = \beta$) iff there exists a t such that
(i) $\mathbf{M}(T[t], T'[t]) \in \Delta^* \times \{\beta\}$, and
(ii) for all $t' \geq t$, $\mathbf{M}(T[t], T'[t]) \in \Delta^* \times \{\beta, ?\}$.

Intuitively, $\mathbf{M}(\sigma, \tau) = (mem, hyp)$ means that the memory and the hypothesis of the learner $\mathbf{M}$ after having seen the sequence σ and the negative counterexample sequence τ is *mem* and *hyp*, respectively. Below, **NC** in the criteria names denotes learning from negative counterexample. **B** and **L** in **BNC**, **LNC**, denote "bounded" and "least".

Definition 6 (Based on [JK08]). Suppose $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ is a target class, and $\mathcal{H} = \{H_\beta : \beta \in J\}$ is a hypothesis space, where both $\mathcal{L}$ and $\mathcal{H}$ are automatic families of languages over an alphabet Σ. Below, for ease of notation, we take $H_? = \emptyset$.

(a) (i) We say that T' is a *counterexample text* for $\mathbf{M}$ on an input text T for a language L iff for all n, where $\mathbf{M}(T[n], T'[n]) = (mem, hyp)$,
if $H_{hyp} \subseteq L$, then $T'(n) = \#$, and
if $H_{hyp} \not\subseteq L$, then $T'(n) \in H_{hyp} - L$.

(ii) We say that T' is a *least-counterexample text* for $\mathbf{M}$ on an input text T for a language L iff for all n, where $\mathbf{M}(T[n], T'[n]) = (mem, hyp)$,
if $H_{hyp} \subseteq L$, then $T'(n) = \#$, and
if $H_{hyp} \not\subseteq L$, then $T'(n) = \min(H_{hyp} - L)$.

(iii) We say that T' is a *bounded counterexample text* for $\mathbf{M}$ on an input text T for a language L iff for all n, where $\mathbf{M}(T[n], T'[n]) = (mem, hyp)$,
if $H_{hyp} \cap \{x \in \Sigma^* : x \leq \max(\text{content}(T[n]))\} \subseteq L$, then $T'(n) = \#$, and
if $H_{hyp} \cap \{x \in \Sigma^* : x \leq \max(\text{content}(T[n]))\} \not\subseteq L$, then $T'(n) \in H_{hyp} \cap \{x \in \Sigma^* : x \leq \max(\text{content}(T[n]))\} - L$.
(That is, the size of a counterexample is bounded by the size of the longest positive datum seen so far; consequently, if the size of the least counterexample to the current conjecture exceeds this bound, no counterexample is provided.)

(b) We say that $\mathbf{M}$ **NCEx**-learns the language L (using hypothesis space $\mathcal{H}$) iff for all texts T for L, for all counterexample texts T' for $\mathbf{M}$ on input text T, $\mathbf{M}(T, T')\downarrow_{hyp} = \beta$ such that $H_\beta = L$.

(c) We say that $\mathbf{M}$ **NCEx**-learns $\mathcal{L}$ (using hypothesis space $\mathcal{H}$) if it **NCEx**-learns all languages in $\mathcal{L}$ (using hypothesis space $\mathcal{H}$).

(d) **NCEx** $= \{\mathcal{L} : (\exists \mathbf{M})[\mathbf{M}$ **NCEx**-learns $\mathcal{L}$ using some hypothesis space $]\}$.
One can similarly define learnability criteria **LNCEx** and **BNCEx** for learning from least-counterexample or bounded counterexamples.

Furthermore, automatic, consistent, monotonic learning and various memory restricted learning criteria can be similarly defined for learning from counterexamples. Here for word-size memory constraint, we bound the memory by the largest word seen in either the text (for positive data) or the counterexample text. Also, for consistency we require that the learner is consistent with positive examples as well as negative counterexamples, that is, for any text T and corresponding negative counterexample text T', if $\mathbf{M}(T[t], T'[t]) = (mem, hyp)$ with $hyp \neq ?$, then $\text{content}(T[t]) \subseteq H_{hyp}$ and $\text{content}(T'[t]) \cap H_{hyp} = \emptyset$.

We use "**Auto**" in the name of the learning criteria to denote that we require the learners to be automatic. For example, **AutoTxtEx** denotes **TxtEx**-learning by an automatic learner. Similarly, we use **Cons** and **Mon** in the name of the learning criteria to denote that the learners are consistent and monotonic, respectively. Similarly, we use **Word** and **Hyp** in the name of the learning criteria to denote that the memory of the learners is appropriately bounded. For **It** memory restriction, as is common in the literature, we replace the term "**Ex**" in the name of the criterion by "**It**".

For example, **AutoWordNCEx** denotes **NCEx** learnability by a learner which is automatic and word memory size bounded. **AutoMonNCIt** denotes **NCEx** learnability by a learner which is automatic, monotonic and iterative.

3 Results

We begin with an easily provable useful technical proposition.

Proposition 7 *Suppose $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ is an automatic family, where the indexing is 1–1. Then, there exists a constant c such that the following hold.*

(a) [JOPS11] For all $\alpha \in I$ such that L_α is finite, $|\alpha| \leq c + \max(\{|x| : x \in L_\alpha\})$.

(b) For all $\alpha \in I, u \in \Sigma^*$, let $ProbExt(\alpha, u) = \{\beta : L_\alpha \subset L_\beta \subseteq L_\alpha \cup \{x : x \leq_{ll} u\}\}$. Then, for all $\alpha \in I, u \in \Sigma^*$ and $\beta \in ProbExt(\alpha, u)$, $|\beta| \leq \max(\{|\alpha|, |u|\}) + c$.

(c) For all $\alpha \in I$, for all $u \in \Sigma^*$, there exists a $\beta \in I$ such that $|\beta| \leq |u| + c$ and $L_\beta \cap \{x : |x| \leq |u|\} = L_\alpha \cap \{x : |x| \leq |u|\}$.

Intuitively, part (a) of the above proposition says that the indices for finite sets are not too big in an automatic family. Part (b) of the proposition says that if L_α and L_β differ only on strings $\leq_{ll} u$, then the index for β is not much bigger than $\max(\{|\alpha|, |u|\})$. Part (c) of the above proposition says that for any index α and string u, there exists a short β such that L_β is consistent with L_α for strings below u.

Our first major result shows that automatic **NCEx**-learners with word-size memory limit can learn any automatic class.

Theorem 8. *Let $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ be an automatic family. Then,*

*(a) $\mathcal{L} \in$ **AutoNCIt**. The learner uses a class preserving hypothesis space.*

*(b) $\mathcal{L} \in$ **AutoWordNCEx**. The learner uses the hypothesis space $(H_\alpha)_{\alpha \in I}$, where $H_\alpha = L_\alpha$.*

Proof. Due to space restrictions, we only show part (a). Part (b) can be proven in a way similar to Theorem 10.

Without loss of generality assume that the indexing $(L_\alpha)_{\alpha \in I}$ is 1–1 (otherwise, we can ignore the non-minimal indices of I, which can be automatically determined as (non) minimal indices can be expressed as a first order formula using automatic relations (see Lemma 1)). Furthermore, assume that I is infinite (otherwise, the theorem is trivial). Let c be as in Proposition 7 (for $\mathcal{L}$).

Let i_0 be a special symbol which we take to be length-lexicographically smaller than all members of I. This is for ease of presentation of the proof.

Suppose L is the target language. The aim of the learner **M** is to find an α such that $L_\alpha \subseteq L$ and $L \subseteq L_\alpha$. The learner can check if $L_\alpha \subseteq L$ using the counterexamples. However, the learner may not easily be able to check if $L \subseteq L_\alpha$, as it may have forgotten some past data. To overcome this problem is the main aim of the construction.

The learner keeps memory of the form $conv(\alpha, u, \beta, b)$, where $\alpha \in I \cup \{i_0\}$, $\beta \in I$, $u \in \Sigma^* \cup \{\#\}$, and $b \in \{0, 1\}$. In case $\alpha = i_0$ in the memory, then we will have $b = 1$ (that is, the memory will never be of the form $conv(i_0, u, \beta, 0)$).

The hypothesis of the learner is directly linked to its memory: If, [$b = 1$ or [$b = 0$ and $|\beta| \leq \max(\{|\alpha|, |u|\}) + c$]], then $H_{conv(\alpha,u,\beta,b)} = L_\beta$; otherwise,

$H_{\mathit{Conv}(\alpha,u,\beta,b)} = L_\alpha$ (note that the above implicitly gives the hypothesis space used by the learner, which is class preserving as $\alpha = i_0$ implies $b = 1$).

Intuitively, α (when $\alpha \neq i_0$) is the index for which the learner is currently testing (or last tested) whether $L_\alpha = L$ (in case the learner finds $L_\alpha \neq L$, it may continue to keep the same α for some time until it finds an appropriate replacement for it). The α used by the learner will always have the property that $L_\alpha \subseteq L$. Thus the learner then needs to check if $L \subseteq L_\alpha$. Though the learner can check for any future elements seen in the input whether they belong to L_α (this is kept track of by using the parameter b in the memory), the learner may not be able to check whether the past data belonged to L_α, as it may have forgotten them. For this purpose, learner keeps track of a parameter u which length-lexicographically bounds any elements in the past which may be in $L - L_\alpha$ (how the learner keeps track of u will be clearer later). The learner uses the parameter β to search for any potential index such that $L_\alpha \subset L_\beta \subseteq L$. If such a β exists, then the learner replaces α above by β, and continues the process. In case such a β does not exist, then α would be the only possible index for L. The learner uses Proposition 7(b) to bound the search for such β in case the learner, since it has started testing for L_α, has not seen an element in $L - L_\alpha$.

We now proceed formally. Let T be a text for the input language L and T' be a sequence of counterexamples. Suppose $\mathbf{M}(T[n], T'[n]) = (\mathit{mem}_n, \mathit{hyp}_n)$, where $\mathit{mem}_n = \mathit{conv}(\alpha_n, u_n, \beta_n, b_n)$. We will always have $\mathit{hyp}_n = \mathit{mem}_n$. Thus, the learner is iterative. The invariants maintained by the learner related to the memory are as follows. For ease of notation below, we take $L_{i_0} = \emptyset$. For all n:

(I1) $\alpha_n \leq_{ll} \alpha_{n+1} \leq_{ll} \beta_n \leq_{ll} \beta_{n+1}$.
(I2) $L_{\alpha_n} \subseteq L_{\alpha_{n+1}} \subseteq L$.
(I3) For all $\alpha' <_{ll} \beta_n$ such that $\alpha' \neq \alpha_n$ and $\alpha' \in I$, $L_{\alpha'} \neq L$.
(I4) $\max(\mathrm{content}(T[n]) - L_{\alpha_n}) \leq_{ll} u_n$. Furthermore $u_n \leq_{ll} u_{n+1}$.

Let m be the least number such that $\alpha_m = \alpha_n$.
(I5) $b_n = 0$ iff $\alpha_n \neq i_0$ and $\{T(s) : m \leq s < n\} \subseteq L_{\alpha_n}$.
(I6) If $b_n = 0$, then $u_n = u_m$; otherwise $u_n = \max(\{u_m\} \cup \{T(s) : m \leq s < n\} - L_{\alpha_n})$.
(I7) If $m < n$ then, $\beta_{n-1} = \beta_n$ iff $[b_{n-1} = 0$ and $|\beta_{n-1}| > \max(\{|\alpha_n|, |u_n|\}) + c]$.

We now specify how the learner computes $\alpha_n, u_n, \beta_n, b_n$. Initially, $\alpha_0 = i_0$, β_0 is the length-lexicographically least element of I, $b_0 = 1$ and u_0 is the length-lexicographically least element of Σ^*. We now describe how the memory of the learner is updated after receiving input $T(n), T'(n)$ (where the previous memory is $\mathit{conv}(\alpha_n, u_n, \beta_n, b_n)$).

Let $u_{n+1} = u_n$, if $T(n) = \#$ or $[\alpha_n \neq i_0$ and $T(n) \in L_{\alpha_n}]$; otherwise, $u_{n+1} = \max(\{u_n, T(n)\})$. For defining, $\alpha_{n+1}, \beta_{n+1}, b_{n+1}$, consider the following cases.

Case 1: $[b_n = 1$ or $[b_n = 0$ and $|\beta_n| \leq \max(\{|\alpha_n|, |u_n|\}) + c]]$.
In this case $H_{\mathit{hyp}_n} = L_{\beta_n}$.

Case 1a: $[\alpha_n = i_0$ or $L_{\alpha_n} \subset L_{\beta_n}]$ and $T'(n) = \#$.

In this case, $L_{\beta_n} \subseteq L$ and either $\alpha_n = i_0$ or $L_{\alpha_n} \subset L_{\beta_n}$. Thus, α_n is not a correct index for L (note that $i_0 \notin I$).
Let $\alpha_{n+1} = \beta_{n+1} = \beta_n$ and $b_{n+1} = 0$.

Case 1b: $[\alpha_n \neq i_0$ and $\neg[L_{\alpha_n} \subset L_{\beta_n}]]$ or $T'(n) \neq \#$.
In this case, either $\alpha_n = \beta_n$ or $L_{\beta_n} \neq L$ (note that $(L_\alpha)_{\alpha \in I}$ is 1–1).
Let $\alpha_{n+1} = \alpha_n$, $\beta_{n+1} = \mathrm{succ}_I(\beta_n)$.
Let $b_{n+1} = 1$, if $b_n = 1$ or $T(n) \notin L_{\alpha_n}$; otherwise $b_{n+1} = 0$.

Case 2: Not Case 1.
In this case $H_{hyp_n} = L_{\alpha_n}$.
Let $\alpha_{n+1} = \alpha_n$, $\beta_{n+1} = \beta_n$.
Let $b_{n+1} = 1$, if $b_n = 1$ or $T(n) \notin L_{\alpha_n}$; otherwise $b_{n+1} = 0$.

It is now easy to verify that the learner is automatic and word size memory bounded. Definition of $\alpha_0, \beta_0, b_0, u_0$ clearly maintain the invariants. We now show that the construction maintains the invariants while defining $\alpha_{n+1}, \beta_{n+1}, u_{n+1}$, b_{n+1}. Note that in Case 1a, $\alpha_n \neq \beta_n = \alpha_{n+1}$, which is the only case which changes value of α_n. (I1) is clearly maintained by both cases ($\beta_{n+1} \geq_{ll} \beta_n$ in both cases, and α_{n+1} is either α_n or β_n). (I2) is maintained as the only time $\alpha_{n+1} \neq \alpha_n$ is via Case 1a, where $L_{\alpha_n} \subseteq L_{\beta_n} \subseteq L$ holds. (I3) is maintained as in Case 1a, $L_{\beta_n} \subseteq L$ and either $\alpha_n = i_0$ or $L_{\alpha_n} \subset L_{\beta_n}$; in Case 1b, $L_{\beta_n} \neq L$ or $\beta_n = \alpha_n$, and in Case 2 $\alpha_{n+1} = \alpha_n$ and $\beta_{n+1} = \beta_n$. (I4), (I5) and (I6) are also maintained by definition of u_{n+1} and b_{n+1} in both cases. Note that in Case 1a, $L_{\alpha_n} \subseteq L_{\beta_n} = L_{\alpha_{n+1}}$. (I7) is trivially maintained by Case 1a; Case 1b and Case 2 also maintain (I7) as Case 1b makes $\beta_{n+1} \neq \beta_n$ and Case 2 makes $\beta_{n+1} = \beta_n$ (note the conditions for Cases 1 and 2).

Now, suppose $L \in \mathcal{L}$. By invariants (I1) and (I3), $\alpha_n \leq_{ll} \alpha'$, for α' such that $L_{\alpha'} = L$. It follows using (I1) that $\lim_{n\to\infty} \alpha_n$ converges, to say α. Here, note that $\alpha \neq i_0$, as eventually by Case 1b, a β_n would be chosen such that $L_{\beta_n} \subseteq L$, making $\alpha_{n+1} = \beta_n$ via Case 1a. If $L_\alpha = L$, then clearly by (I5), $\lim_{n\to\infty} b_n$ also converges; If $L_\alpha \neq L$, then by (I1) and (I3), $\lim_{n\to\infty} \beta_n$ converges, (since β_n is then bounded by the index for L) and thus by (I7) $\lim_{n\to\infty} b_n$ converges. Thus, in either case $\lim_{n\to\infty} b_n$ converges, to say b. If $L_\alpha \neq L$, then using (I1) and (I3), we have that $\lim_{n\to\infty} \beta_n$ converges; if $L_{\alpha_n} = L$, then by (I5) $\lim_{n\to\infty} b_n = 0$, and thus by (I6) $\lim_{n\to\infty} u_n$ converges, and thus by (I7) $\lim_{n\to\infty} \beta_n$ converges. Hence, in both cases we have that $\lim_{n\to\infty} \beta_n$ converges, to say β. Thus, by (I4), (I7) we have that $\lim_{n\to\infty} u_n$ converges, to say u. Thus, the memory of the learner converges to $conv(\alpha, u, \beta, b)$. By (I2) we have that $L_\alpha \subseteq L$. By, (I7) we have that $|\beta| > \max(\{|\alpha|, |u|\}) + c$ and $b = 0$. Thus, $H_{\mathrm{C}onv(\alpha,u,\beta,b)} = L_\alpha$. Furthermore, using the invariants (I3), (I4) and Proposition 7(b), we have that $L_\alpha = L$.

Thus, **M NCIt**-learns $\mathcal{L}$. ∎

Hypotheses of the learner in the above theorem are not consistent with the data seen so far. We can make the learner consistent if it receives least counterexamples whenever it's hypothesis contains data that is not a part of the target language.

Theorem 9 (Frank Stephan, personal communication). *Let $\mathcal{L}$ be an automatic class. Then, $\mathcal{L} \in$ **AutoWordConsLNCEx** as witnessed by a learner using a class comprising hypothesis space.*

Now we turn to automatic **BNCEx**-learning with word-size memory limit.

Theorem 10. *Let $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ be an automatic family which consists only of infinite languages. Then, $\mathcal{L} \in$ **AutoWordBNCEx**. The learner uses the hypothesis space $(H_\alpha)_{\alpha \in I}$, where $H_\alpha = L_\alpha$.*

Proof. Without loss of generality assume that the indexing $(L_\alpha)_{\alpha \in I}$ is 1–1 (otherwise, we can ignore the non-minimal indices of I, which can be automatically determined). Furthermore, assume that I is infinite (otherwise, the theorem is trivial). Let c be as in Proposition 7 (for $\mathcal{L}$).

For ease of presentation, the size of the memory of the learner is word size bounded only for the case when the input language is in the class $\mathcal{L}$. One can easily convert such a learner to always having word-size memory bound by remembering the length-lexicographically largest word seen in the text/counterexample text, and if the memory tries to exceed the appropriate bound (relevant constant plus the size of the remembered largest word), then abandoning the learning process.

The learner $\mathbf{M}$ has memory of the form: (α, w, u, β), where $\alpha, \beta \in I$, $w, u \in \Sigma^* \cup \{\#\}$. Let T be a text for the input language L and T' be a sequence of counterexamples. Suppose $\mathbf{M}(T[n], T'[n]) = (mem_n, hyp_n)$, where $mem_n = (\alpha_n, w_n, u_n, \beta_n)$.

Intuitively, α_n is the index for which the learner is currently testing if $L_{\alpha_n} = L$. The length-lexicographically largest element seen in the input $T[n]$ is denoted by w_n. The length-lexicographically largest element seen in the text T before the learner starts testing for α_n is denoted by u_n.

If $L_{\alpha_n} \not\subseteq L$, $L \in \mathcal{L}$ and $\mathbf{M}$ conjectures L_{α_n} infinitely often then the learner will eventually get a counterexample for it as every language in $\mathcal{L}$ is infinite. For the elements received after the learner starts testing for α_n, the learner can check if they belong to L_{α_n} as the elements are received. However, the learner may have forgotten the elements it had seen before it starts testing for L_{α_n} (note that all the forgotten elements would be $\leq_{ll} u_n$, though we do not exactly know which). For testing whether such elements are in $L - L_{\alpha_n}$, the learner checks if there is some $\beta \in I$ which satisfies: $L_{\alpha_n} \subset L_\beta \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}$ and $L_\beta \cap \{x : x \leq_{ll} u_n\} \subseteq L$. Such β's (satisfying $L_{\alpha_n} \subset L_\beta \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}$) are finite in number and can be determined using Proposition 7(b).

We proceed formally now. The invariants maintained by the learner related to the memory are as follows. For all n:

(I1) $w_n = \max(\text{content}(T[n]))$.
(I2) $\alpha_n \leq_{ll} \alpha_{n+1}$.
(I3) For all $\alpha' <_{ll} \alpha_n$ with $\alpha' \in I$, $L_{\alpha'} \neq L$, where $\text{content}(T'[n]) \cap L_{\alpha'} \neq \emptyset$ or there exists an $x \leq_{ll} w_n$, $x \in L - L_{\alpha'}$.

Let m be the least number such that $\alpha_m = \alpha_n$.

(I4) $u_n = \max(\text{content}(T[m]))$.
(I5) If $n = m$, then $\beta_n = \alpha_n$.
(I6) **(i)** $\text{content}(T[n]) - \{x : x \leq_{ll} u_n\} \subseteq L_{\alpha_n}$,
(ii) for all β such that $\alpha_n <_{ll} \beta <_{ll} \beta_n$, if $L_{\alpha_n} \subset L_\beta \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}$, then $L_\beta \not\subseteq L$, and
(iii) if $m < n$ and $|\beta_{n-1}| \leq \max(\{|\alpha_n|, |u_n|\}) + c$, then $\beta_{n-1} <_{ll} \beta_n$.
(iv) if $m < n$ and $|\beta_{n-1}| > \max(\{|\alpha_n|, |u_n|\}) + c$, then $\beta_{n-1} = \beta_n$.

The hypothesis of the learner is directly obtainable from memory as follows. If $|\beta_n| \leq \max(\{|\alpha_n|, |u_n|\}) + c$, then $hyp_n = \beta_n$; otherwise, $hyp_n = \alpha_n$. Thus, it is enough to specify how the learner computes $\alpha_n, w_n, u_n, \beta_n$.

Initially, $\alpha_0 = \beta_0 =<_{ll}$-least element of I, $w_0 = u_0 = \#$. We now describe how memory of the learner is updated after receiving input $T(n), T'(n)$ (where the previous memory is $(\alpha_n, w_n, u_n, \beta_n)$).

Case 1: $T(n) \notin L_{\alpha_n} \cup \{\#\}$ or $T'(n) \in L_{\alpha_n}$ or $[T'(n) = \#, |\beta_n| \leq \max(\{|\alpha_n|, |u_n|\}) + c$ and $L_{\alpha_n} \subset L_{\beta_n} \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}]$.
This case implies that $L_{\alpha_n} \neq L$ as either $T(n) \in L - L_{\alpha_n}$ or $T'(n) \in L_{\alpha_n} - L$ or $[L_{\beta_n} \cap \{x : x \leq_{ll} u_n\} \subseteq L_{\beta_n} \cap \{x : x \leq_{ll} w_n\} \subseteq L$ and $L_{\beta_n} \cap \{x : x \leq_{ll} u_n\} \not\subseteq L_{\alpha_n}]$. Furthemore, note that either $L_{\alpha_n} \cap \text{content}(T'[n+1]) \neq \emptyset$, or there exists a $x \leq_{ll} w_{n+1}$ such that $x \in L - L_{\alpha_n}$.
Let $\alpha_{n+1} = \beta_{n+1} = \text{succ}_I(\alpha_n)$. Let $w_{n+1} = u_{n+1} = \max(\text{content}(T[n+1]))$. Note that w_{n+1}, u_{n+1} can be computed using w_n and $T(n)$.
Case 2: Not Case 1 and $|\beta_n| \leq \max(\{|\alpha_n|, |u_n|\}) + c$
Note that in this case $hyp_n = \beta_n$. Furthermore, either $L_{\beta_n} \not\subseteq L$ (when, $T'(n) \neq \#$) or $\neg[L_{\alpha_n} \subset L_{\beta_n} \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}]$ (as Case 1 does not hold).
Let $\alpha_{n+1} = \alpha_n$, $u_{n+1} = u_n$, $w_{n+1} = \max(\text{content}(T[n+1]))$, $\beta_{n+1} = \text{succ}_I(\beta_n)$.
Case 3: Not Case 1 and $|\beta_n| > \max(\{|\alpha_n|, |u_n|\}) + c$
Note that in this case $hyp_n = \alpha_n$. Furthermore, $T(n) \in L_{\alpha_n}$ and $T'(n) = \#$.
Let $\alpha_{n+1} = \alpha_n$, $\beta_{n+1} = \beta_n$, $u_{n+1} = u_n$, and $w_{n+1} = \max(\text{content}(T[n+1]))$.

Clearly, the learner **M** is automatic.

The invariants (I1), (I2), (I4), (I5), (I6)(iii), (iv) are clearly maintained by the construction. For (I3) note that Case 1 is the only case where $\alpha_{n+1} \neq \alpha_n$, and in this case $L_{\alpha_n} \neq L$. For (I6)(i), note that if $T(n)$ is not in L_{α_n}, then by Case 1, $\alpha_{n+1} \neq \alpha_n$; thus, using (I4), (I6)(i) holds. For (I6)(ii), note that in Case 1, $\beta_{n+1} = \alpha_{n+1}$, in Case 3 $\beta_{n+1} = \beta_n$ and in Case 2, $L_{\beta_n} \not\subseteq L$ or $\neg[L_{\alpha_n} \subset L_{\beta_n} \subseteq L_{\alpha_n} \cup \{x : x \leq_{ll} u_n\}]$; thus, (I6)(ii) is maintained in all the cases.

By (I5), (I6)(iii), (iv), we have that length of β is at most a constant more than $\max(\{|\alpha_n|, |w_n|\})$. Furthermore, by (I1), (I3) and Proposition 7(c), we have that $|\alpha_n|$ is bounded in length by a constant plus $\max(\text{content}(T[n]) \cup \text{content}(T'[n]))$. Thus, **M** is word-size memory bounded.

Now, for $L = L_{\alpha'}$, $\alpha' \in I$, by invariant (I3), $\alpha_n \leq_{ll} \alpha'$. Thus, by (I2) $\lim_{n\to\infty} \alpha_n$ converges, to say α. Thus, by (I4), $\lim_{n\to\infty} u_n$ converges, to say u.

Furthermore, using the invariants (I5) and (I6)(iii), we have that, for all but finitely many n, $|\beta_n| > \max(\{|\alpha_n|, |u_n|\}) + c$. Thus, by (I3), (I6)(i), (I6)(ii), and Proposition 7(b), we have that either, $L_\alpha \not\subseteq L$ or $L_\alpha = L$. Furthermore, by definition of hyp_n, for all but finitely many n, $hyp_n = \alpha$. Thus, if $L_\alpha \not\subseteq L$, then by cardinality of L being infinite, we must have $T'(n) \in L_\alpha$ (and thus Case 1 holding) for large enough n, a contradiction.

It follows that $L_\alpha = L$. Thus, the learner **M** **NCEx**-learns $\mathcal{L}$. ∎

Yet a result from [JK08] can be used to show that some automatic classes cannot be **BNCEx**-learned (even by a non-automatic learner).

Theorem 11. *[JK08] Let $\mathcal{L} = \{\Sigma^*\} \cup \{L_x : x \in \Sigma^*\}$, where $L_x = \{y : y \leq_{ll} x\}$. Then, $\mathcal{L}$ is an automatic family and $\mathcal{L} \notin$* **BNCEx***.*

The following corollary shows that, for the unrestricted automatic learnability, as well as automatic learnability with all types of memory restrictions, there are automatic classes that are **NCEx**-learnable, but not **BNCEx**-learnable.

Corollary 12. *(a)* **AutoNCIt** − **AutoBNCIt** $\neq \emptyset$.
(b) **AutoHypNCEx** − **AutoHypBNCEx** $\neq \emptyset$.
(c) **AutoWordNCEx** − **AutoWordBNCEx** $\neq \emptyset$.
(d) **AutoNCEx** − **AutoBNCEx** $\neq \emptyset$.

Our next result shows that some automatic class, while not **HypBNCEx**-learnable, can be automatically learned with word-size memory without negative counterexamples.

Theorem 13. **AutoWordTxtEx** − **HypBNCEx** $\neq \emptyset$.

Let $\Sigma = \{0\}$. Let $L_0 = \{0^{2n} : n \geq 0\}$.

Let $L_{1^i} = \{0^{2n} : n \leq i\} \cup \{0^{2i+1}\}$. Let $L_{(2^i,3^j)} = L_{1^i} \cup \{0^{2j}\}$.

Let $\mathcal{L} = \{L_\alpha : \alpha \in I\}$, where $I = \{0, 1^i, (2^i, 3^j) : i, j \in N\}$.

Then $\mathcal{L}$ witnesses Theorem 13. We omit the detailed proof.

The next theorem shows that automatic iterative learners using negative counterexamples still can sometimes learn automatic classes that cannot be learned using positive data alone.

Theorem 14. (**AutoWordNCEx** ∩ **AutoWordBNCEx** ∩ **AutoNCIt** ∩ **AutoBNCIt**) − **TxtEx** $\neq \emptyset$.

Let $\Sigma = \{a\}$. Let $L_\epsilon = a^*$, and $L_w = L_0 - \{w\}$, for $w \in a^+$. Let $\mathcal{L} = \{L_w : w \in a^*\}$. Then $\mathcal{L}$ witnesses Theorem 14. We omit the detailed proof.

Our last result shows that monotonic (even non-automatic) learners cannot learn some automatic classes, even using least counterexamples.

Theorem 15. *There exists an automatic class $\mathcal{L} = \{L_\alpha : \alpha \in I\}$ such that $\mathcal{L} \notin$* **MonLNCEx**.

Proof. Let $\Sigma = \{0\}$. Let $L_0 = 0^*$, $L_{1^j} = \{0^i : i \leq j\}$, $L_{conv(1^j,2^k)} = \{0^i : i \leq j\} \cup \{0^k\}$. Let $I = \{0, 1^j, conv(1^j, 2^k) : j, k \in N\}$, and $\mathcal{L} = \{L_\alpha : \alpha \in I\}$.

Clearly, $\mathcal{L}$ is an automatic family. Suppose, by way of contradiction, that $\mathbf{M}$ is a monotonic learner which **LNCEx**-learns $\mathcal{L}$. Consider the shortest σ such that $\mathbf{M}(\sigma, \#^{|\sigma|})$ is for an infinite language. Note that there exists such a σ as $\mathbf{M}$ learns L_0. Now consider a text T extending σ for L_{1^j}, where $j = \max(\text{content}(\sigma) \cup \bigcup_{s<|\sigma|} L_{\mathbf{M}(\sigma[s],\#^s)})$. Let T' be the least-counterexample text for $\mathbf{M}$ on the text T. Then $\mathbf{M}(T, T')$ must converge to a grammar g for L_{1^j}. Thus, content(T') is finite. Let $x \in L_{\mathbf{M}(\sigma,\#^{|\sigma|})} - (\text{content}(T) \cup \text{content}(T'))$. Let m be such that content(T) = content$(T[m])$, content(T') = content$(T'[m])$, $\sigma \subseteq T[m]$, and $\mathbf{M}(T[m], T'[m]) = g$. Then $\mathbf{M}$ is not monotonic on $T[m] \diamond x$, where counterexamples provided are least counterexamples. ■

4 Conclusions

In this paper we considered learning automatic families by automatic learners which receive negative counterexamples. Various versions of memory restriction and counterexamples were considered. Table 1 gives a summary of results regarding learning all classes of a particular type for various criteria.

Table 1. Summary of results on when all classes of particular type are learnable

Learning Criterion	Aut. Classes of Infinite Languages	All Automatic Classes	Consistent Learning for All Aut. Classes
Auto(Word, Hyp)NCEx **AutoNCIt**	yes	yes	open
Auto(Word)LNCEx	yes	yes	yes
Auto(Word)BNCEx	yes	no	no
AutoHypBNCEx **AutoBNCIt**	open	no	no

We showed that there is an automatic class which is in **AutoWordBNCEx**−**AutoHypBNCEx**, though at this point we do not know if **AutoWordBNCEx** properly contains **AutoHypBNCEx**. It is also open whether **AutoBNCEx** ⊆ **AutoWordBNCEx**. Note that the corresponding problems in **AutoTxtEx** learning (without using negative counterexample) are also open [JLS10]. Regarding monotonic learning, we showed that there are automatic families which cannot be **LNCEx**-learnt by any monotonic (even non-automatic) learners.

Acknowledgements. We thank the anonymous referees for several helpful comments and pointing out a bug in the earlier version of the paper and an improvement of Theorem 8. We thank Frank Stephan for discussion as well as pointing out Theorem 9, which strengthened an earlier version of this theorem.

References

[Ang80] Angluin, D.: Inductive inference of formal languages from positive data. Information and Control 45, 117–135 (1980)

[Ang88] Angluin, D.: Queries and concept learning. Machine Learning 2(4), 319–342 (1988)

[Bār74] Bārzdiņš, J.: Inductive inference of automata, functions and programs. In: Proceedings of the 20th International Congress of Mathematicians, Vancouver, pp. 455–460 (1974) (in Russian); English translation in American Mathematical Society Translations: Series 2 109, 107–112 (1977)

[BG00] Blumensath, A., Grädel, E.: Automatic structures. In: 15th Annual IEEE Symposium on Logic in Computer Science (LICS), pp. 51–62. IEEE Computer Society (2000)

[Gol67] Gold, E.M.: Language identification in the limit. Information and Control 10(5), 447–474 (1967)

[HPTS84] Hirsh-Pasek, K., Treiman, R., Schneiderman, M.: Brown and Hanlon revisited: Mothers' sensitivity to ungrammatical forms. Journal of Child Language 11, 81–88 (1984)

[JK08] Jain, S., Kinber, E.: Learning languages from positive data and negative counterexamples. Journal of Computer and System Sciences 74(4), 431–456 (2008); Special Issue: Carl Smith memorial issue

[JLS10] Jain, S., Luo, Q., Stephan, F.: Learnability of Automatic Classes. In: Dediu, A.-H., Fernau, H., Martín-Vide, C. (eds.) LATA 2010. LNCS, vol. 6031, pp. 321–332. Springer, Heidelberg (2010)

[JOPS11] Jain, S., Ong, Y.S., Pu, S., Stephan, F.: On automatic families. In: Arai, T., Feng, Q., Kim, B., Wu, G., Yang, Y. (eds.) Proceedings of the 11th Asian Logic Conference, in Honor of Professor Chong Chitat's 60th Birthday, 2009, pp. 94–113. World Scientific (2011)

[KN95] Khoussainov, B., Nerode, A.: Automatic Presentations of Structures. In: Leivant, D. (ed.) LCC 1994. LNCS, vol. 960, pp. 367–392. Springer, Heidelberg (1995)

[LZ93] Lange, S., Zeugmann, T.: Language learning in dependence on the space of hypotheses. In: Proceedings of the Sixth Annual Conference on Computational Learning Theory, pp. 127–136. ACM Press (1993)

[Wie76] Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. Journal of Information Processing and Cybernetics (EIK) 12(1-2), 93–99 (1976)

[Wie90] Wiehagen, R.: A Thesis in Inductive Inference. In: Dix, J., Jantke, K., Schmitt, P. (eds.) NIL 1990. LNCS, vol. 543, pp. 184–207. Springer, Heidelberg (1991)

Regular Inference as Vertex Coloring

Christophe Costa Florêncio[1] and Sicco Verwer[2,⋆]

[1] Department of Computer Science, University of Amsterdam, The Netherlands
[2] Institute for Computing and Information Sciences, Radboud University Nijmegen, The Netherlands

Abstract. This paper is concerned with the problem of supervised learning of deterministic finite state automata, in the technical sense of identification in the limit from complete data, by finding a minimal DFA consistent with the data (regular inference).

We solve this problem by translating it in its entirety to a vertex coloring problem. Essentially, such a problem consists of two types of constraints that restrict the hypothesis space: *inequality* and *equality* constraints.

Inequality constraints translate to the vertex coloring problem in a very natural way. Equality constraints however greatly complicate the translation to vertex coloring. In previous coloring-based translations, these were therefore encoded either dynamically by modifying the vertex coloring instance on-the-fly, or by encoding them as satisfiability problems. We provide the first translation that encodes both types of constraints together in a pure vertex coloring instance. This offers many opportunities for applying insights from combinatorial optimization and graph theory to regular inference. We immediately obtain new complexity bounds, as well as a family of new learning algorithms which can be used to obtain both exact hypotheses, as well as fast approximations.

1 Introduction

The regular inference problem consists of learning (finding) a smallest deterministic finite state automaton (DFA) that is consistent with a given set of labeled strings, rejecting the negative strings and accepting the positive strings. The decision version of finding a DFA with a given upper bound on its size (number of states) was shown to be **NP**-complete in [3, 18], and an inapproximability result was demonstrated in [24]. In spite of these hardness results, quite a few DFA identification algorithms exist, see [11]. In particular, a recently proposed algorithm based on a *translation* of the regular inference problem into satisfiability (SAT) has shown promising results [16].

The translation in [16] is based on an earlier translation of regular inference to graph coloring in [10]. Graph coloring is the problem of assigning a color to every node in a given graph such that nodes with the same color do not share

[⋆] The second authos is supported by STW project 11763 Integrating Testing And Learning of Interface Automata (ITALIA).

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 81–95, 2012.

an edge. Determining whether there exists a coloring that uses at most $k \geq 3$ colors is a well-known **NP**-complete problem, see, e.g., [13]. The main idea of this translation into graph coloring is to use a distinct color for every state of the learned DFA. The nodes in the graph coloring instance represent the labeled strings and share an edge if one of them is positive and the other negative. The graph coloring problem thus ensures that pairs of positive and negative examples cannot obtain the same color, and therefore cannot end in the same state, making the resulting DFA consistent. The size of this DFA is determined by the amount of colors used in the graph coloring problem. Finding the minimum is done by iterating over this amount.

The above mentioned reduction from [10], however, was not purely based on graph coloring. In addition to the *inequality constraints*, denoting that two vertices cannot be assigned the same color, so-called *equality constraints* are needed to model regular inference. These constraints denote that two vertices should be assigned the same color if two other vertices are assigned the same color. Together, the equality and inequality constraints can efficiently encode the regular inference problem. Unfortunately, however, it has remained unknown how to encode such constraints in a graph coloring problem instance. In [10], they were encoded dynamically by creating new graph coloring instances that satisfied them on-the-fly. In [16], they were encoded directly into satisfiability instead of in the intermediary graph coloring instance. In this paper, we develop the first construction that encodes them directly into graph coloring.

In terms of complexity (size), our encoding of the equality constraints is comparable to the encoding to satisfiability described in [16]: they both require $O(|C|^2 \cdot |V|)$ additional clauses or vertices, where C is the set of colors and V is the size of the data set (the APTA, see Section 2). The inequality constraints, however, are much easier to encode in graph coloring, requiring only a single edge for every constraint compared to the $O(|C|^2)$ (or $O(|C|)$ for some that can be encoded more efficiently, see [16]) clauses that are needed for every such constraint in a satisfiability instance. In addition, using our encoding we can make use of sophisticated solvers for graph coloring, including techniques for symmetry-breaking, many local-search based approaches, cutting-plane algorithms, etc. see, e.g., [21].

2 Background and Notation

2.1 Regular Inference

A *deterministic finite state automaton* (DFA) is one of the basic and most commonly used finite state machines. Below, we provide a concise description of DFAs, the reader is referred to [25] for a more elaborate overview. A DFA $A = \langle Q, T, \Sigma, q_0, F \rangle$ is a directed graph consisting of a set of *states* Q (nodes) and labeled *transitions* T (directed edges). The *start state* $q_0 \in Q$ is a specific state of the DFA and any state can be an *accepting state* (final state) in $F \subseteq Q$. The labels of transitions are all members of a given *alphabet* Σ. A DFA A can be used to *generate* or *accept* sequences of symbols (strings) using a process called *DFA computation.* This process begins in q_0, and iteratively *activates* (or *fires*) an outgoing transition $t_i = \langle q_{i-1}, q_i, l_i \rangle \in T$ with label $l_i \in \Sigma$ from the *source state* it is in, q_{i-1}, moving the process to the *target state* q_i pointed to by t_i. A computation $q_0 t_1 q_1 t_2 q_2 \ldots t_n q_n$ is *accepting* if the state it *ends* in (its last state) is an accepting state, i.e., $q_n \in F$, otherwise it is *rejecting.* The labels of the activated transitions form a string $l_1 \ldots l_n$. A DFA accepts exactly those strings formed by the labels of accepting computations, it rejects all others. A DFA is *deterministic*, which means that for every state q and every label l there exists at most one outgoing transition from q with label l. The set of all strings accepted by a DFA A is called the *language* $L(A)$ of A.

Given a pair of finite sets of positive example strings S_+ and negative example strings S_-, called the *input sample*, the goal of *regular inference* (or *DFA identification/learning*) is to find a (non-unique) *smallest* DFA A that is *consistent* with $S = \{S_+, S_-\}$, i.e., such that every string in S_+ is accepted by A, and every string in S_- is rejected by A. Typically, the size of a DFA is measured by the number of states it contains. Seeking this DFA is an active research topic in the grammatical inference community, see, e.g., [11].

For many years, the state-of-the-art in DFA identification has been the *evidence-driven state-merging* (EDSM) algorithm [20]. State-merging is a common technique from grammatical inference for learning a small language model by combining (merging) the states of a large initial DFA model, see, e.g., [11]. Essentially, EDSM is a *greedy method* that tries to find a good local optimum efficiently. In addition, an earlier state-merging method called RPNI has been shown to *converge efficiently* (from polynomial time and data) to the global optimum in the limit [23]. EDSM participated in and won (in a tie) the Abbadingo DFA learning competition in 1997 [20].

Since our method is based on the simple yet effective state-merging approach, we now briefly explain this approach. For more information, the reader is referred to [11]. The key idea of state-merging is to first construct a tree-shaped DFA A from the input sample S, and then to merge (combine) the states of A. This initial DFA A is called an *augmented prefix tree acceptor* (APTA). An example is shown in Figure 1.

Definition 1. *The* APTA $A = (\langle Q, T, \Sigma, q_0, F \rangle, R)$ *for an input sample* $\{S_+, S_-\}$ *consists of a DFA* $\langle Q, T, \Sigma, q_0, F \rangle$ *and a set of rejecting states* R*, where* Σ *is the alphabet of* $S_+ \cup S_-$*,* $q_0 = \epsilon$ *(the empty word),* $Q = \{a \in \Sigma^* \mid \exists b \in \Sigma^* : ab \in (S_+ \cup S_-)\}$*,* $T = \{\langle a, a', l \rangle \in Q \times Q \times \Sigma \mid a' = al\}$*,* $F = S_+$*, and* $R = S_-$*.*

A *merge* of two states q and q' combines the states into one: it creates a new state q^* that has the incoming and outgoing transitions of both q and q', which are subsequently removed from A. Such a merge is only allowed if the states are *consistent*, i.e., it is not the case that q is accepting while q' is rejecting or vice versa. When a merge introduces a non-deterministic choice, i.e., q^* is the source of two transitions with the same label l, the target states of these transitions q_1 and q_2 are merged as well. This is called the *merging for determinization* process and is continued until there are no non-deterministic choices left. However, if this

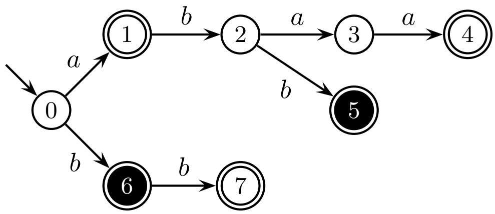


Fig. 1. An augmented prefix tree acceptor for $S = (S_+ = \{a, abaa, bb\}, S_- = \{abb, b\})$. The start state is the state with an arrow pointing to it from nowhere.

process at some point merges two inconsistent states, the original states q and q' are also considered inconsistent and the merge will fail. The result of a successful merge is a new DFA that is smaller than before, and still consistent with the input sample S. A state-merging algorithm iteratively applies this state merging process until no more consistent merges are possible.

In the grammatical inference community, there has been some research into developing advanced and efficient search techniques based on the EDSM heuristic. The idea is to increase the quality of a solution by searching other paths in addition to the path determined by the greedy EDSM heuristic. Examples of such advanced techniques are dependency-directed backtracking [22], using mutually (in)compatible merges [1], and searching most-constrained nodes first [19]. A comparison of different search techniques for EDSM can be found in [8]. Recently, instead of wrapping a search technique around EDSM, a *translation* of the regular inference problem into satisfiability (SAT) was proposed in order to use a state-of-the-art SAT-solver to search for an optimal solution [16]. The main advantage of such an approach is that it makes use directly of advanced search techniques such as conflict analysis, intelligent back-jumping, and clause learning, see, e.g., [5]. The winning contribution to the 2010 Stamina DFA learning competition was a combination of this SAT-based approach and EDSM with a modified heuristic [26]. Other recently proposed improvements are the parallelization of the algorithm [2], and the use of ensembles of learned DFAs [12].

2.2 Translating Regular Inference

The idea of translating the regular inference problem to other computational problems for which dedicated solvers exist is not new. In fact, one of the earliest regular inference algorithms due to Biermann [6] is of this type. Biermann proposed to solve the regular inference problem by mapping it to constraint satisfaction. In this translation, every state is represented by a natural number, constraints on the possible values of states are added that enforce consistency, and the aim is to minimize the range of these numbers, which translates back to minimizing the number of states in the resulting DFA. More recently, Grinchtein et al. [15] adapted this translation in order to map regular inference to satisfiability (SAT) instead of constraint satisfaction. The numeric constraints from

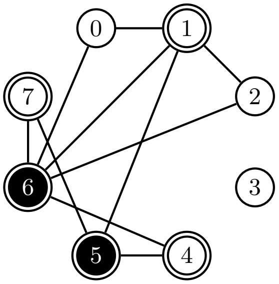


Fig. 2. The consistency graph corresponding to the APTA of Figure 1. Some states in the consistency graph are not directly inconsistent, but inconsistent due to determinization. For instance states 2 and 6 are inconsistent because the strings *abb* (negative) and *bb* (positive) would end in the same state if these states were merged.

the constraint satisfaction problem are encoded using either a unary or a binary scheme into clauses and literals for the satisfiability problem.

Another type of translation is that of Coste [10], who maps regular inference to graph coloring based on the state-merging approach. The main idea of this translation is to use a distinct color for every state of the identified DFA. Every node in the graph coloring problem corresponds to a distinct state in the APTA. Two vertices v and w in this graph are connected by an edge (cannot be assigned the same color), if merging v and w results in an inconsistency in the original regular inference problem:

Definition 2. *The* consistency graph $G_c = (V, E_c)$ *for an APTA* $(\langle Q, T, \Sigma, q_0, F\rangle, R)$ *consists of a set of vertices* V *and edges* E_c *such that* $V = Q$*, and* $E_c = \{\{a, a'\} \in \Sigma^* \times \Sigma^* \mid \exists b \in \Sigma^* : ab \in F$ *and* $a'b \in R\}$*.*

The edges in this graph are called *inequality constraints*. Figure 2 shows an example of such a graph. In addition to these inequality constraints, *equality constraints* are required: if the parents of two states (in the APTA) with the same incoming transition label are merged, then these states must be merged too (encoding the merging for determinization procedure).

Definition 3. *The set of* equality constraints E_e *for an APTA* $A = (\langle Q, T, \Sigma, q_0, F\rangle, R)$ *is the set of pairs of paired states* $\langle (a, b), (al, bl)\rangle \subset Q^2 \times Q^2$ *with* $a, b \in \Sigma^*$ *and* $l \in \Sigma$*.*

For graph coloring problem, these equality constraints encode that two parent states a and b can get the same color only if their child states al and bl get the same color. Until now it has been unclear how to encode such constraints in a graph coloring problem instance. In [10], these were encoded by modifying the graph according to the consequences of these constraints. This implies that a new graph coloring instance has to be solved every time an equality constraint is used. This is clearly not very efficient. Thirteen years later, this graph coloring encoding in [10] was used by Heule and Verwer as a basis for a more efficient translation to satisfiability [16], which encodes the equality constraints directly.

In the following, we develop a novel construction that encodes the equality constraints directly into graph coloring.

2.3 Graph Coloring

We briefly discuss graph coloring in this subsection, and assume that the reader is familiar with the more basic concepts from graph theory.

A *coloring* of a graph is a function from its vertices to *colors* (or *color classes*). The term colors is due to historical reasons; it was originally studied in the context of coloring maps. In the remainder, we will simply use natural numbers as names for these colors.

A coloring is called *proper* for graph G if no two connected vertices in G have the same color, and *optimal* for G if it is both proper and assigns the smallest possible number of colors to the vertices of G. This number is known as the *chromatic number* of G, denoted by $\chi(G)$. We will write $\text{color}(x) = c$ when vertex x is labeled with color c. We write $x =_c y$ to indicate that vertices x, y are members of the same color class.

When we call an optimal coloring *unique*, this is taken to mean unique *up to recoloring*. Recoloring can be understood as renaming, i.e., applying a substitution σ to the color labels of G such that, whenever for any two vertices x, y from G, $x =_c y$, then $\sigma[x] =_c \sigma[y]$, and when $x \neq_c y$, then $\sigma[x] \neq_c \sigma[y]$. Note that for every recoloring, its inverse exists.

3 Encoding Equality Constraints into the Graph

In this section we will show how equality constraints can either be encoded into the graph, or can be reduced to simple checks after a coloring has been generated.

3.1 Graphs with Chromatic Number ≤ 2

The 1-colorable graphs are obviously exactly the edgeless graphs. Since all vertices of the graph are members of the same color class, the issue of equality constraints is irrelevant in this case.[1]

Also note that a target automaton with just one state always generates Σ^*; for any sample for such a language, $S_- = \emptyset$.

It is a well-known fact that the 2-colorable graphs are exactly the bipartite graphs. For this class, a coloring can be found in polynomial time with a parity-based algorithm: pick an arbitrary vertex v and label all vertices in the graph with their distance to v (this can be done with depth-first search). We obtain a

[1] As an aside, it should be noted that $\chi(G_c(S)) = 1$ does *not* imply that the target automaton consists of just one state. This can be easily seen by considering any sample with $S_- = \emptyset$. This implies the complete absence of conflicts, but this may simply be due to a sample not being representative for the target language.

bipartite graph, one partition of which consists of all vertices at even distance from v, and the other partition of which consists of all vertices at odd distance from v. Each partition can then be regarded as a color class. Two vertices obviously have the same color if and only if they are members of the same partition.

Equality constraints can be ignored when both of the pairs of vertices involved in such a constraint are from the same connected component of the bipartite graph. It suffices to check that the constraint is not violated *after* the coloring has been assigned to the consistency graph.[2]

However, in the case that G_c consists of multiple connected components, equality constraints may block certain merges, resulting in $\chi(G_e) > 2$.

3.2 Graphs with Chromatic Number ≥ 3

When the chromatic number of the consistency graph is three or more, equality constraints have to be taken into account. This requires the conbstruction of a graph that, for each equality constraint, includes a gadget as seen in Figure 3.

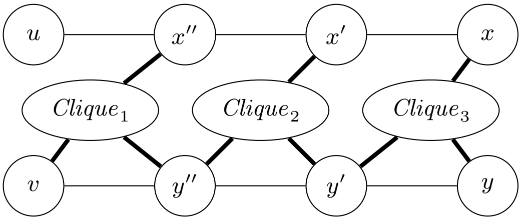


Fig. 3. This gadget encodes equality constraints into a graph. A thick line represents a set of edges that connect a vertex (circle) to all vertices in a clique (ellipse).

This construction is formally defined as follows:

Definition 4. *Given a consistency graph $G_c = G_c(S) = (V, E)$, let $\chi = \chi(G_c)$, and let $Clique_1$, $Clique_2$ and $Clique_3$ be three disjoint cliques of size $\chi - 2$.*

Let E_e be the set of equality constraints for APTA(S)*, and let $G_e = (V \cup V', E \cup E')$ be the smallest graph such that, for each equality constraint $e = \langle(u, v), (x, y)\rangle \in E_e$,*

1. *$Clique_1$, $Clique_2$ and $Clique_3$ are in the graph;*
2. *vertices x', x'', y', y'' are in the graph;*
3. *v, x'', y'' are connected to all vertices in $Clique_1$;*
4. *x', y'', y' are connected to all vertices in $Clique_2$;*
5. *y', x, y are connected to all vertices in $Clique_3$;*
6. *u is connected to x'', $x''$to x', v to y'', y'' to y', x to x', and y to y'.*

We are now in the position to state a lemma which will play a key role in the remainder of this paper:

[2] Technically speaking, even this check is not necessary: a violation can only occur if the sample is inconsistent, and such a case is excluded by definition.

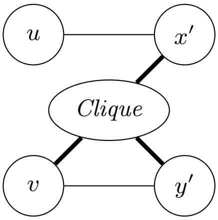


Fig. 4. The subgraph discussed in Lemma 1

Lemma 1. *Given a graph G, let $\chi = \chi(G)(\geq 3)$, and let G' be an induced subgraph of G which is isomorphic to $G''[u, v, x', y']$, where G'' is the graph from Figure 4, with its clique of size $\chi - 2$.*

Then, given any optimal coloring for G':

1. *either $x' =_c v$, or $x' =_c y'$;*
2. *if it is the case that $u =_c v$, then we also have $x' =_c y'$.*

Proof. Let C be the set of all colors used in some optimal coloring of G, and let C_1 be the colors assigned to the subgraph *Clique*. Since *Clique* is of size $\chi - 2$, $\chi(\mathit{Clique}) = \chi - 2$, thus $|C - C_1| = 2$.

Because v is connected to all vertices in *Clique*, it has to be assigned a color c_v from $C - C_1$. Since y' is connected to v and to all vertices in *Clique*, it has the color $C - C_1 - c_v$.

Vertex x' is connected to all vertices in *Clique*, so it has a color from $C - C_1$. Since this set contains just 2 colors, either $x' =_c v$, or $x' =_c y'$. Since x' is connected to u, it has a color from $C - C_1 - c_u$. If $u =_c v$, this set is a singleton and contains just the color assigned to y'. □

We are now in a position to prove correctness of our construction.

Proposition 1. *Let $G_c = G_c(S)$, and $\chi(G_c) \geq 3$. Let $G_e = G_e(G_c)$ (as given in Definition 4).*

Then, given an optimal coloring for G_e, for any equality constraint $e = \langle (u, v), (x, y) \rangle \in E(G_c)$, if the vertices corresponding to u and v are in the same color class, then so are x and y.

Proof. Given an equality constraint which states that merging u and v requires merging x and y, we show the following:

1. in our construction, if vertices u and v are in the same color class, then x and y aree members of the same color class, for any minimal coloring;
2. if vertices u and v are *not* in the same color class, then x and y can be in the same color class, but not necessarily;
3. we show that our construction is correct for any combination of 3 colors;
4. we show that it remains correct for any combination of more than 3 colors.

Demonstrating these four points together proves the proposition. First, let $C(G) = \{c_1, \ldots c_\chi\}$ be the set of all colors used in any optimal coloring of graph G ($\chi = \chi(G)$).

Point 1 can be demonstrated by applying Lemma 1 three times: if $u =_c v$, then $x'' =_c y''$; if $x'' =_c y''$, then $x' =_c y'$; if $x' =_c y'$, then $x =_c y$.

Thus $u =_c v$ implies $x =_c y$.

We now proceed to demonstrate point 2 using the same method: if $u \neq_c v$, then by Lemma 1 an optimal coloring exists with $x'' \neq_c y''$; similarly for $x' \neq_c y'$; and thus $x \neq_c y$. If $u \neq_c v$, then by Lemma 1 an optimal coloring exists with $x'' =_c y''$; therefore $x' =_c y'$; and thus $x =_c y$.

Point 3 can best be demonstrated by case analysis, i.e., simply enumerating all possible colorings (up to recoloring). As the reader may check, Figure 5 exhaustively enumerates all possible cases for $\chi = 3$. i.e. all unique colorings.

We conclude by demonstrating point 4:

Points 1 and 2 hold for any $\chi \geq 3$, so it suffices to generalize point 3 to cases where $\chi \geq 4$. Let G_χ be a gadget as in Definition 4, for some $\chi \geq 4$, and G_3 the same for $\chi = 3$. It is clear that G_3 is an induced subgraph of G_χ. To be more precise, G_χ can be obtained from G_3 by adding $i - 3$ distinct new vertices to each of its three central cliques and connecting them in the obvious way.

It is easy to see that, in the case that $x =_c y$ and $u =_c v$, we can obtain an optimal coloring for G_χ by picking (a recoloring of) one of the lowest three colorings from Figure 5. This colors a subgraph isomorphic to G_3, the colors for the vertices not in this subgraph are the 'new' ones added to each of the central cliques $Clique_i$ are obtained simply by non-deterministically assigning them from $C - C(Clique_i) - C(N(Clique_i))$ (where $C(G)$ yields the colors assigned to vertices in G, and $N(G)$ yields the union of neighborhoods of all vertices in G).

In the case that $x =_c y$ and $u \neq_c v$, colorings can be obtained from the middle three colorings from Figure 5. The top left vertex, u, can be assigned any color as long as $u \neq_c v$, since it's not connected to any of the cliques. It is connected only to x'', and we have $x'' =_c v$. If v gets assigned a color such that color$(v) > 3$, we get $v \neq_c y''$, so we get a proper coloring when no vertex in $Clique_1$ is assigned color(v) or color(y'') (which is 1 or 2 in the figure). The same line of reasoning can be applied to x and y: If x and y get assigned a color such that color$(x) > 3$, we obtain an admissible coloring just when no vertex in $Clique_3$ is assigned color(x) or color(x'') (which is 1 or 2 in the figure).

For the case that $x \neq_c y$, consider the top three colorings from Figure 5. A complicating factor is that the lower right vertex, y, has multiple options for coloring; for a gadget for χ colors, there are $\chi - 1$ options. It is easy to see though that the only restriction on color(y) is that $y \neq_c y'$, which implies $y \neq_c x$, since $y' =_c x$ for all three gadgets. So, if color$(y) > 3$, the other of the $\chi - 1$ options can be assigned to vertices in $Clique_3$ and an admissible coloring is obtained. For vertices x, u, v, reasoning from the previous paragraphs applies.

We have thus demonstrated the validity of all four points, which concludes the proof. □

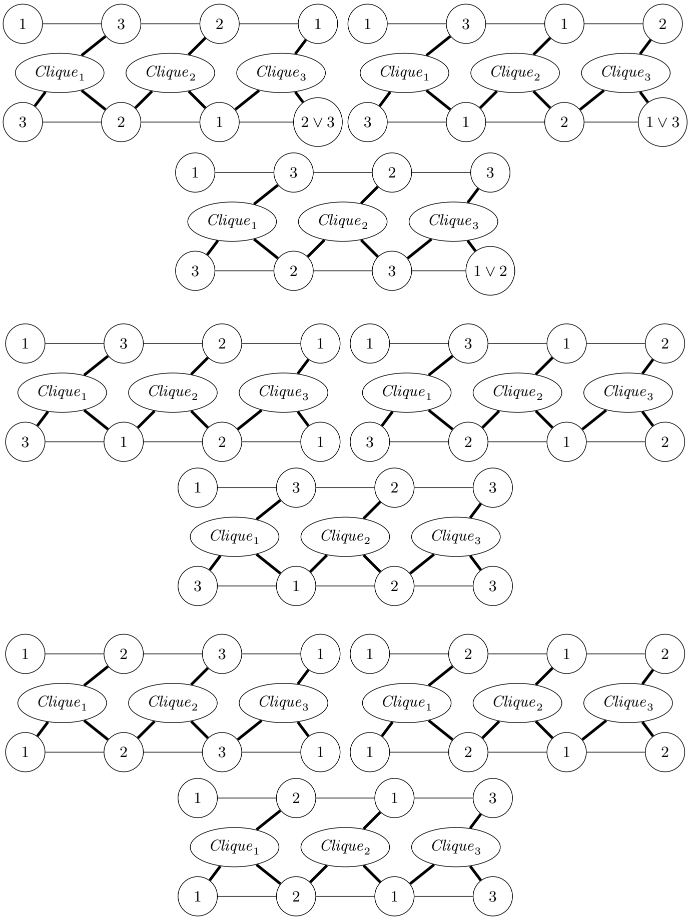

1 3 2 1
$Clique_1$ $Clique_2$ $Clique_3$
3 2 1 $2 \vee 3$
1 3 1 2
$Clique_1$ $Clique_2$ $Clique_3$
3 1 2 $1 \vee 3$
1 3 2 3
$Clique_1$ $Clique_2$ $Clique_3$
3 2 3 $1 \vee 2$
1 3 2 1
$Clique_1$ $Clique_2$ $Clique_3$
3 1 2 1
1 3 1 2
$Clique_1$ $Clique_2$ $Clique_3$
3 2 1 2
1 3 2 3
$Clique_1$ $Clique_2$ $Clique_3$
3 1 2 3
1 2 3 1
$Clique_1$ $Clique_2$ $Clique_3$
1 2 3 1
1 2 1 2
$Clique_1$ $Clique_2$ $Clique_3$
1 2 1 2
1 2 1 3
$Clique_1$ $Clique_2$ $Clique_3$
1 2 1 3

Fig. 5. Possible colorings for the $\chi(G_c) = 3$ construction

4 More Efficient Equality Constraints

The translation described above encodes the equality constraints from the regular inference problem, but unfortunately it is not very efficient: in the worst case it can require up to $O(\|S\|^2)$ cliques of size $\chi - 2$. Since S (the input sample) can get very large, this quadratic relation is highly undesirable. In [16], a similar problem was observed for a translation of regular inference to satisfiability. There, it was solved by introducing additional variables that encode the equality constraints globally, i.e., for the resulting automaton model instead of per pair of APTA states. Below, we show that such a global encoding is also possible for our translation to graph coloring and that it reduces this quadratic relation to a linear one.

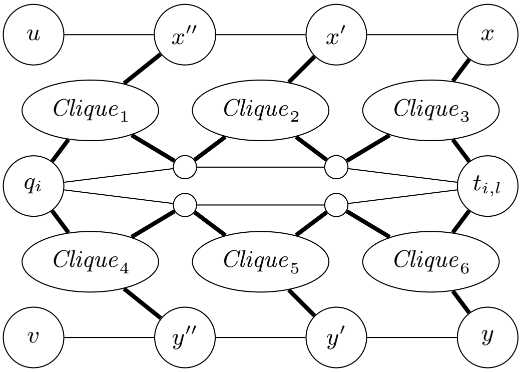


Fig. 6. This gadget encodes equality constraints into a graph more efficiently than the gadget from Figure 3. There exists one node q_i for every color i, and one node $t_{i,l}$ for every color i and symbol l from the alphabet. The gadget is repeated for every possible color i, and the q_i nodes connected to each other in a clique of their own. Although the number of gadgets required per equality constraint is increased, the resulting encoding is more efficient due to the overlap in the created subgraphs: every pair of nodes (v, y) or (u, x) needs to be connected only once to every $(q_i, t_{i,l})$.

The key idea is to introduce two additional sets of nodes that encode the states of the resulting automaton model and the transitions between them. The first set contains a clique of χ vertices, one for every state of the automaton. The second set contains $\chi \cdot |\Sigma|$ pairwise non-connected vertices, one for every possible transition of the automaton. We denote the vertices from the first set using q_i (state i in the resulting automaton), and those from the second set using $t_{i,l}$ (the target of the transition from state i with label a). We now replace the v and y vertices from the gadget in Figure 3 by the q_i and $t_{i,l}$ vertices shown in Figure 6. This construction is identical to the previous one, except that it connects every pair of vertices (u, x) that is used in an equality constraint $\langle u, v, x, y \rangle \in E_e$ for a label l $(v = ul)$ to $(q_i, t_{i,l})$ for all $0 \leq i < \chi$. If two pairs of vertices (u, x) and (v, y) were connected by the gadget in Figure 3 in the translation described in the previous section, they are now connected through the vertices $(q_i, t_{i,l})$ from the two gadgets in Figure 6.

As shown below, this is sufficient to correctly encode every equality constraint.

Proposition 2. *By replacing every occurrence of v by q_i and y by $t_{i,l}$ for all $0 \leq i < \chi$ in Definition 4, we obtain the construction in Figure 6. Let G_e be the graph resulting from this construction. Then, given a minimal coloring for G_e, for any equality constraint $\langle (u, v), (x, y) \rangle \in E_e$, if the vertices corresponding to u and v are in the same color class, then so are x and y. Furthermore, no other constraints are encoded by the gadget in Figure 6.*

Proof. Due to the clique connecting the nodes q_i, there exists an q_i in C for any color class C. Thus, if u and v are in the same color class C, then there exists a q_i that is in this class as well. By Proposition 1, there exists in G_e a gadget that forces $t_{i,l}$ to be in the same color class C' as x since u and q_i are both in C. Similarly, there exists a gadget that forces $t_{i,l}$ to be in the same color class as y. Clearly, this is only possible if y is also in C'.

It is also straightforward to see that this new gadget does not impose any constraints other than equality. If u and v are in a different class, then they are in the same class with different q_i and q_j, and thus x and y are in the same class as different $t_{i,l}$ and $t_{j,l}$, which can belong to different color classes. Furthermore, since the gadget connects (u, x) with $(q_i, t_{i,l})$ for all i only if there is a transition from u to x in the APTA with label l, no constraints are constructed for pairs of states (u, x) and (v, y) with differently labeled transitions between (u, x) and (v, y). □

The size of the resulting translation is significantly smaller than before since every pair of nodes (u, x) that occurs on one side of an equality constraint for label l now connects through the gadget from Figure 3 to all pairs of nodes $(q_i, t_{i,l})$ for $0 \leq i < \chi$, resulting in $O(\|S\| \cdot \chi)$ gadgets instead of $O(\|S\|^2)$.

5 Learning Algorithm

Definition 5. *Let* LEARN(S) *be the following algorithm:*

```
Require: Sample $S = (S_+, S_-)$, CHROM_NR(), COLOR()
  $A :=$ APTA $(S_+, S_-)$
  $G_c(= (V_c, E_c)) := G_c(A)$ {consistency graph for $A$}
  upp_bound $:= |V_c|$
  $\chi :=$ CHROM_NR($G_c$)
  for $i = \chi$ to upp_bound do
    $G_e := G_e(\mathrm{APTA}(S), i)$ {consistency graph with equality constraints for $A$
    assuming $i$ colors}
    $C =$ COLOR($G_e, i$) {proper coloring for $G_e$ with $i$ colors}
    if $C$ defined then
      BREAK
    end if
  end for
  for all $c$ in $C$ do
    MERGE all states in $A$ that correspond to vertices in $c$
  end for
  compute normal form $A'$ of $A$ {only observable part, no 'reject' labels}
  return $A'$
```

Here, CHROM_NR and COLOR are user-specified algorithms for determining chromatic number and computing a vertex coloring, respectively. Note that $\chi(G_c)$ may be underestimated without affecting correctness so simply the constant 1 would be acceptable as CHROM_NR.

It was shown in [14] that an algorithm that enumerates all DFAs with monotonically increasing size until it finds one consistent with a sample, identifies all DFAs in the limit. Thus, if we assume CHROM_NR(G_c) $\leq \chi(G_e)$, and we choose the APTA-generating algorithm and COLOR() so that they yield the first automaton in such an enumeration consistent with the sample, we obtain:

Theorem 1. *The algorithm given in Definition 5 solves the regular inference problem, that is, it finds a minimal automaton consistent with given positive and negative data.*

Proof. It is clear that COLOR finds a optimal coloring for G_e. Since there is a one-to-one correspondence between color classes of G_e and states in the hypothesized automaton, the hypothesis is always an automaton of minimal size (w.r.t. the sample). Since G_c is an induced subgraph of G_e, the hypothesis does not violate any inequality constraints and thus accepts all of S_+ and rejects all of S_-. By Proposition 1, the resulting automaton also respects all equality constraints. Thus the hypothesis is always an automaton of minimal size consistent with given positive and negative data. □

Corollary 1. *The algorithm given in Definition 5 identifies in the limit from positive and negative data the class of all deterministic finite state automata.*

It should be clear that our algorithm is consistent, order-independent and set-driven. We leave open the questions of conservative learning and the possibility of an incremental learning algorithm.

6 Bounds

Recall that we established an upper bound on the number of equality constraints of $\|S\| \cdot \chi$ (Section 4). Since the gadget consists of $4+3(\chi-2)$ vertices, in the case that $\chi \geq 3$, we obtain an upper bound of $s \cdot \chi \cdot (4+3(\chi-2))+s = s \cdot (3\chi^2+2\chi+1)$ vertices in G_e, where $s = \|S\|$ and $\chi = \chi(G_c)$.

Note that this does not necessarily imply that our learning algorithm has quadratic space requirements. Depending on the choice of algorithm for COLOR, it may not be necessary to explicitly represent G_e with, for example, an adjacency matrix. Instead, a representation of G_c could be used, and the additional edges and vertices necessary for representing equality constraints could be computed on the fly just when the coloring algorithm requires them. It will in general be necessary to keep track of the colors assigned to the additional vertices, but for the cliques in the equality subgraphs a representation can be used that requires, for every such clique, only as many bits as $\chi(G_e)$.

The fastest known (exact) vertex coloring algorithm has a time bound of $O(2^v v)$ ([7], v being the number of vertices in G_e), and, given graphs of chromatic number 3 or 4, the tighter bounds of $O(1.3289^v)$ ([4]) and $O(1.7504^v)$ ([9]), respectively. Combined with our bound for the size of G_e, assuming that the algorithm has to iterate from 1 to χ, and assuming CHROM_NR simply yields 1, we obtain the following time bounds ($s = \|S\|$ and $\chi = |A|$):

1. target automaton has 1 or 2 states: $f(s)$, with f some polynomial function;
2. target automaton has exactly 3 states: $O(1.3289^{22s})$;
3. target automaton has exactly 4 states: $O(1.3289^{22s} + 1.7504^{41s})$;
4. target automaton has 5 or more states: $O((\chi - 2) \cdot 2^{s\cdot(3\chi^2+2\chi+1)} \cdot (3\chi^2 + 2\chi + 1))$.

7 Discussion

Algorithms based on semidefinite programming techniques are known that find optimal colorings for perfect graphs in polynomial time. These can often also be used to find approximate colorings for non-perfect graphs in polynomial time. The algorithm discussed in [17], for example, has a hyperparameter which allows the user to obtain solutions anywhere on the spectrum between solutions that use few colors but are not necessarily proper, and proper colorings that may be far removed from a optimal coloring.

The former corresponds with an automaton inconsistent with the sample, the latter with an automaton with more states than the target automaton. This makes a learning algorithm based on such an approach flexible; the user can decide which trade-off is appropriate for the problem at hand by setting the value of this hyperparameter on a case-by-case basis.

References

[1] Abela, J., Coste, F., Spina, S.: Mutually Compatible and Incompatible Merges for the Search of the Smallest Consistent DFA. In: Paliouras, G., Sakakibara, Y. (eds.) ICGI 2004. LNCS (LNAI), vol. 3264, pp. 28–39. Springer, Heidelberg (2004)

[2] Akram, H.I., Batard, A., de la Higuera, C., Eckert, C.: PSMA: A parallel algorithm for learning regular languages. In: NIPS Workshop on Learning on Cores, Clusters and Clouds (2010)

[3] Angluin, D.: On the complexity of minimum inference of regular sets. Information and Control 39(3), 337–350 (1978)

[4] Beigel, R., Eppstein, D.: 3-coloring in time $O(1.3289^n)$. Journal of Algorithms 54(2), 168–204 (2005)

[5] Biere, A., Heule, M.J.H., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability. Frontiers in Artificial Intelligence and Applications, vol. 185. IOS Press (February 2009)

[6] Biermann, A.W., Feldman, J.A.: On the synthesis of finite-state machines from samples of their behavior. IEEE Trans. Comput. 21(6), 592–597 (1972)

[7] Björklund, A., Husfeldt, T., Koivisto, M.: Set partitioning via inclusion-exclusion. SIAM J. Comput. 39(2), 546–563 (2009)

[8] Bugalho, M., Oliveira, A.L.: Inference of regular languages using state merging algorithms with search. Pattern Recognition 38, 1457–1467 (2005)

[9] Byskov, J.M.: Enumerating maximal independent sets with applications to graph colouring. Operations Research Letters 32(6), 547–556 (2004)

[10] Coste, F., Nicolas, J.: Regular inference as a graph coloring problem. In: Workshop on Grammatical Inf., Automata Ind., and Language Acq., ICML 1997 (1997)

[11] de la Higuera, C.: Grammatical Inference: Learning Automata and Grammars. Cambridge University Press (2010)

[12] García, P., Vázquez de Parga, M., López, D., Ruiz, J.: Learning Automata Teams. In: Sempere, J.M., García, P. (eds.) ICGI 2010. LNCS (LNAI), vol. 6339, pp. 52–65. Springer, Heidelberg (2010)

[13] Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co. (1979)

[14] Gold, E.M.: Complexity of automaton identification from given data. Information and Control 37(3), 302–320 (1978)

[15] Grinchtein, O., Leucker, M., Piterman, N.: Inferring Network Invariants Automatically. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 483–497. Springer, Heidelberg (2006)

[16] Heule, M.J.H., Verwer, S.: Exact DFA Identification Using SAT Solvers. In: Sempere, J.M., García, P. (eds.) ICGI 2010. LNCS (LNAI), vol. 6339, pp. 66–79. Springer, Heidelberg (2010)

[17] Karger, D., Motwani, R., Sudan, M.: Approximate graph coloring by semidefinite programming. J. ACM 45, 246–265 (1998)

[18] Kearns, M.J., Valiant, L.: Cryptographic limitations on learning Boolean formulae and finite automata. J. ACM 41, 67–95 (1994)

[19] Lang, K.J.: Faster algorithms for finding minimal consistent DFAs. Technical report, NEC Research Institute (1999)

[20] Lang, K.J., Pearlmutter, B.A., Price, R.A.: Results of the Abbadingo One DFA Learning Competition and a New Evidence-Driven State Merging Algorithm. In: Honavar, V.G., Slutzki, G. (eds.) ICGI 1998. LNCS (LNAI), vol. 1433, pp. 1–12. Springer, Heidelberg (1998)

[21] Malaguti, E., Toth, P.: A survey on vertex coloring problems. International Transactions in Operational Research 17(1), 1–34 (2010)

[22] Oliveira, A.L., Marques-Silva, J.P.: Efficient search techniques for the inference of minimum sized finite state machines. In: String Processing and Information Retrieval, pp. 81–89 (1998)

[23] Oncina, J., Garcia, P.: Inferring regular languages in polynomial update time. In: Pattern Recognition and Image Analysis. Series in Machine Perception and Artificial Intelligence, vol. 1, pp. 49–61. World Scientific (1992)

[24] Pitt, L., Warmuth, M.K.: The minimum consistent DFA problem cannot be approximated within any polynomial. In: STOC, pp. 421–432 (1989)

[25] Sudkamp, T.A.: Languages and Machines: an introduction to the theory of computer science, 3rd edn. Addison-Wesley (2006)

[26] Walkinshaw, N., Lambeau, B., Damas, C., Bogdanov, K., Dupont, P.: STAMINA: a competition to encourage the development and assessment of software model inference techniques. Empirical Software Engineering, 1–34 (to appear)

Sauer's Bound for a Notion of Teaching Complexity

Rahim Samei, Pavel Semukhin, Boting Yang, and Sandra Zilles

Department of Computer Science, University of Regina, Canada
{samei20r,semukhip,boting,zilles}@cs.uregina.ca

Abstract. This paper establishes an upper bound on the size of a concept class with given recursive teaching dimension (RTD, a teaching complexity parameter.) The upper bound coincides with Sauer's well-known bound on classes with a fixed VC-dimension. Our result thus supports the recently emerging conjecture that the combinatorics of VC-dimension and those of teaching complexity are intrinsically interlinked.

We further introduce and study RTD-maximum classes (whose size meets the upper bound) and RTD-maximal classes (whose RTD increases if a concept is added to them), showing similarities but also differences to the corresponding notions for VC-dimension.

Another contribution is a set of new results on maximal classes of a given VC-dimension.

Methodologically, our contribution is the successful application of algebraic techniques, which we use to obtain a purely algebraic characterization of teaching sets (sample sets that uniquely identify a concept in a given concept class) and to prove our analog of Sauer's bound for RTD.

Keywords: VC-dimension, teaching, Sauer's bound, maximum classes.

1 Introduction

An important combinatorial result, proven by Sauer [7] and independently by Shelah [8], states that the size of any concept class of Vapnik-Chervonenkis dimension (VC-dimension, [11]) d is at most $\sum_{i=0}^{d} \binom{m}{i}$, where m is the number of instances the concept class is defined over.

In Computational Learning Theory, this bound (typically called *Sauer's bound*) has proven helpful—if not essential—for a variety of studies, most notably for the definition and analysis of *maximum classes*. A concept class of VC-dimension d over a finite instance space X is maximum, if its size meets Sauer's bound.[1] Maximum classes exhibit a number of interesting structural properties, *e.g.*, their complements as well as their restrictions to subsets of the instance space are maximum [6, 12]. These structural properties have remarkable implications. For example, maximum classes form one of the few general cases of concept classes known to have labeled and unlabeled sample compression

[1] In this paper, we restrict ourselves to finite instance spaces.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 96–110, 2012.

schemes of the size of their VC-dimension [3, 5]. Moreover, the *recursive teaching dimension* (RTD, a complexity parameter of the recently introduced recursive teaching model [13]) of any maximum class equals its VC-dimension [2].

Recent work [2] indicates connections between the VC-dimension and the RTD; besides maximum classes, several other types of concept classes are shown to have an RTD upper-bounded by their VC-dimension. An open question is whether or not the RTD has an upper bound linear in the VC-dimension. Thus recursive teaching is the only model known so far that could potentially establish a close connection between the complexity of learning from a teacher and the complexity of learning from randomly chosen examples (the VC-dimension being an essential complexity parameter for the latter).

This paper establishes a further connection between RTD and VC-dimension: its main result is an analog of Sauer's bound for RTD. We prove that the size of any concept class of RTD r is at most $\sum_{i=0}^{r} \binom{m}{i}$, where m is the size of the instance space. This new evidence of a strong connection between learning from a teacher and learning from randomly chosen examples suggests that the study of the recursive teaching dimension deserves more attention. Our result is proven using algebraic methods, which first provide us with a purely algebraic characterization of *teaching sets*. A teaching set for a concept c in a concept class C is a set of labeled examples that is consistent with c but with no other concept in C; thus it uniquely identifies c in C. Our algebraic characterization of teaching sets, a second highlight of this paper, is the main ingredient of our proof of Sauer's bound for RTD, but it may be of independent interest. In particular, the algebraic techniques applied here may provide new proof ideas for combinatorial studies in Computational Learning Theory, e.g., we give an example for an alternative proof to Kuzmin and Warmuth's result that maximum classes are shortest-path-closed [5]. Previously, methods from algebra yielded an alternative proof of Sauer's bound for the VC-dimension [10].

Our Sauer-type bound for RTD naturally allows us to define and study the concept of *RTD-maximum classes*—classes whose size meets the upper bound. To distinguish RTD-maximum classes from maximum classes in the original sense, we refer to the latter as *VCD-maximum classes*. Although every VCD-maximum class is shown to be RTD-maximum, RTD-maximum classes turn out to exhibit slightly different properties. For example, their complements are not necessarily RTD-maximum. We further study *RTD-maximal classes*—classes whose RTD increases if any new concept is added to them. Such classes are not necessarily RTD-maximum.

In studying RTD-maximum and RTD-maximal classes, we discover some new interesting properties of VCD-maximal classes. In particular, we provide bounds on the size of VCD-maximal classes, shown in the appendix.

2 Preliminaries

Let X be a finite set, called *instance space*. Elements of X are called instances. A *concept* on X is a subset of X. Each concept c is identified with a function

$c(x)$ defined as follows: $c(x) = 1$ if $x \in c$ and $c(x) = 0$ if $x \notin c$. For $\ell \in \{0, 1\}$, $\bar{\ell}$ is defined as $\bar{\ell} = 1 - \ell$.

A *concept class* C on X is a set of concepts on X, that is, $C \subseteq 2^X$. $\overline{C}$ denotes the complement of C. For $Y \subseteq X$, let $C|_Y$ denote the restriction of C to Y, that is, $C|_Y = \{c \cap Y : c \in C\}$. Similarly, $c|_Y$ means $c \cap Y$. To simplify notation, the restriction $C|_{X \setminus \{x\}}$ will be also denoted as $C - x$, and $c|_{X \setminus \{x\}}$ will be denoted as $c - x$. The *reduction* of C to Y is defined as $C^Y = \{c \subseteq Y : c \cup c' \in C$ for all $c' \subseteq X \setminus Y\}$. In other words, $c \in C^Y$ if and only if all possible extensions of the concept c from Y to X belong to C. If X_1 and X_2 are two disjoint instance spaces, $C_1 \subseteq 2^{X_1}$ and $C_2 \subseteq 2^{X_2}$, then the *direct product* of C_1 and C_2 is a concept class on $X_1 \cup X_2$ defined as $C_1 \times C_2 = \{c_1 \cup c_2 : c_1 \in C_1$ and $c_2 \in C_2\}$. If the class C_1 contains only a single concept and $C_2 = 2^{X_2}$, then the class $C_1 \times C_2$ is called a *cube*. If $|X_2| = d$, then such a cube is called a *d-dimensional cube* (or *d-cube* for short).

A set $S \subseteq X$ is *shattered* by the class C if $C|_S = 2^S$. The *VC-dimension* of a class C is defined as $\mathrm{VCD}(C) = \max\{|S| : S$ is shattered by $C\}$ [11]. Let $\Phi_d(m) = \sum_{i=0}^{d} \binom{m}{i}$. Sauer's lemma states that if $\mathrm{VCD}(C) = d$, then $|C| \leq \Phi_d(|X|)$ [7, 8]. Let $\mathrm{VCD}(C) = d$; then C is called *VCD-maximum* if $|C| = \Phi_d(|X|)$, that is, if the size of C matches the upper bound from Sauer's lemma (cf. [12]). A class is called *maximal* with respect to VC-dimension (or *VCD-maximal*) if adding any new concept to the class increases its VC-dimension.

A *labeled example* is a pair (x, ℓ), where $x \in X$ and $\ell \in \{0, 1\}$. For a set S of labeled examples, $X(S)$ denotes $X(S) = \{x \in X : (x, \ell) \in S$ for some $\ell\}$. A set S of labeled examples is a *teaching set* for a concept c in a class C, if c is the only concept from C which is consistent with S. For simplicity, we then also call $X(S)$ a teaching set since the labels of examples from S are uniquely determined by $X(S)$ and c. The collection of all teaching sets for c in C is denoted $\mathrm{TS}(c, C)$.

The *teaching dimension* of c in C is $\mathrm{TD}(c, C) = \min\{|S| : S \in \mathrm{TS}(c, C)\}$. The teaching dimension of C is defined as $\mathrm{TD}(C) = \max_{c \in C} \mathrm{TD}(c, C)$ [4, 9]. We will also refer to the *minimal* teaching dimension $\mathrm{TD}_{min}(C) = \min_{c \in C} \mathrm{TD}(c, C)$.

The following definitions are based on [2, 13]. A *teaching plan* for a concept class C is a sequence $P = ((c_1, S_1), \ldots, (c_n, S_n))$, where $C = \{c_1, \ldots, c_n\}$ and $S_i \in \mathrm{TS}(c_i, \{c_i, \ldots, c_n\})$ for all $i = 1, \ldots, n$. The *order* of the teaching plan P is $\mathrm{ord}(P) = \max_{i=1,\ldots,n} |S_i|$. The *recursive teaching dimension* of C is

$$\mathrm{RTD}(C) = \min\{\mathrm{ord}(P) : P \text{ is a teaching plan for } C\}.$$

For a teaching plan $P = ((c_1, S_1), \ldots, (c_n, S_n))$ of C whose order is equal to $\mathrm{RTD}(C)$, the set S_i is called a *recursive teaching set* for c_i in C with respect to the plan P, and $|S_i|$ is called the *recursive teaching dimension* of c_i in C with respect to the plan P, denoted $\mathrm{RTD}(c_i, C)$. The words "with respect to the plan P" may be omitted if there is no ambiguity. We will also use the notation $\mathrm{RTD}^*(C) = \max_{X' \subseteq X} \mathrm{RTD}(C|_{X'})$.

The RTD has the following properties [2, 13]:

- RTD is monotonic, i.e, $\mathrm{RTD}(C') \leq \mathrm{RTD}(C)$ whenever $C' \subseteq C$.

- RTD equals the order of any *canonical teaching plan*, i.e., a teaching plan $((c_1, S_1), \ldots, (c_n, S_n))$ with $|S_i| = \mathrm{TD}_{min}(\{c_i, \ldots, c_n\})$ for all $i = 1, \ldots, n$.
- $\mathrm{RTD}(C) = \max_{C' \subseteq C} \mathrm{TD}_{min}(C')$.

3 Algebraic Characterization of Teaching Sets

In this section we give an algebraic characterization of the teaching sets for a concept c in a concept class C. Let $X = \{x_1, \ldots, x_m\}$ be a finite instance space, and let $C = \{c_1, \ldots, c_n\}$ be a concept class on X. Consider a vector space $\mathbf{F}_2^n$ of dimension n over the field $\mathbf{F}_2$ (i.e., the field consisting of 2 elements). For each polynomial $f(x_1, \ldots, x_m)$ with variables from X and coefficients from $\mathbf{F}_2$, we define a vector $f = (f_1, \ldots, f_n)$ from $\mathbf{F}_2^n$ as follows

$$f_i = f(c_i(x_1), \ldots, c_i(x_m)) \text{ for } i = 1, \ldots, n.$$

Note that we use the same notation for a polynomial and a vector. We also associate each concept $c_i \in C$ with the ith standard basis vector $c_i = (0, \ldots, 1, \ldots, 0)$ of $\mathbf{F}_2^n$. Again, we are using the same notation for a concept and a vector. This should not cause confusion as the exact meaning of such notation will be clear from the context. For instance, by "the vector x_1x_2" we mean the vector in $\mathbf{F}_2^n$ that corresponds to the polynomial x_1x_2. Similarly, an equality like $c = f(x_1, x_2)$ should be interpreted as the equality between two vectors, the one corresponding to the concept c and the one corresponding to the polynomial $f(x_1, x_2)$.

To illustrate these notations, let us consider the following concept class:

	x_1	x_2	x_3
c_1	0	1	0
c_2	1	0	1
c_3	0	1	1

In this class, $x_1 = (0, 1, 0)$, $x_2 = (1, 0, 1)$, $x_3 = (0, 1, 1)$, $0 = (0, 0, 0)$ and $1 = (1, 1, 1)$. In our notations, $c_1 = (1, 0, 0)$, $c_2 = (0, 1, 0)$ and $c_3 = (0, 0, 1)$. So we have $x_1 + x_2 = 1$, $x_1x_2 = 0$, $c_1 = x_3 + 1$, $x_2x_3 = (0, 0, 1)$ and hence $c_3 = x_2x_3$.

The following theorem provides an algebraic description of teaching sets.

Theorem 1. *Let $C = \{c_1, \ldots, c_n\} \subseteq 2^X$. A set of instances $\{z_1, \ldots, z_k\} \subseteq X$ is a teaching set for a concept c_i if and only if $c_i = f(z_1, \ldots, z_k)$ for some polynomial f over $\mathbf{F}_2$.*

Proof. Suppose $\{z_1, \ldots, z_k\}$ is a teaching set for c_i. It is not hard to see that in this case $c_i = p_1 \cdots p_k$, where $p_t = z_t$ if $c_i(z_t) = 1$ and $p_t = z_t + 1$ if $c_i(z_t) = 0$.

To prove the other implication, consider $c_i \in C$ and assume that $c_i = f(z_1, \ldots, z_k)$ but $\{z_1, \ldots, z_k\}$ is not a teaching set for c_i. Hence there is another concept $c_j \neq c_i$ from C which coincides with c_i on $\{z_1, \ldots, z_k\}$, that is, $c_i(z_t) = c_j(z_t)$ for all $t = 1, \ldots, k$. Thus the following equalities hold

$$f_i = f(c_i(z_1), \ldots, c_i(z_k)) = f(c_j(z_1), \ldots, c_j(z_k)) = f_j.$$

So, the ith and jth coordinates of the vector $f(z_1, \ldots, z_k)$ are equal. By definition, c_i corresponds to the standard basis vector $(0, \ldots, 1, \ldots, 0)$ which has only one coordinate equal to 1, namely, the ith coordinate. Since we assumed that $c_i = f(z_1, \ldots, z_k)$ and showed that $f_i = f_j$, the vector $f(z_1, \ldots, z_k)$ must have at least two coordinates equal to 1, namely, the ith and jth coordinates. This contradicts the assumption that $c_i = f(z_1, \ldots, z_k)$. □

4 RTD-Maximum Classes

The next theorem is the main result of our paper. It provides a Sauer-type bound on the size of a concept class with a given RTD.

Theorem 2. *Let $C \subseteq 2^X$ and $|X| = m$. If* $\mathrm{RTD}(C) = r$ *then* $|C| \leq \Phi_r(m)$.

Proof. Let P_m^r be the collection of monomials over $\mathbf{F}_2$ of the form $x_{i_1} \cdots x_{i_k}$, where $0 \leq k \leq r$ and $1 \leq i_1 < \cdots < i_k \leq m$. In case when $k = 0$ we let the corresponding monomial be equal to the constant 1. Note that $|P_m^r| = \Phi_r(m)$.

Let $c_1, c_2, \ldots, c_n$ be all the concepts from C listed in the same order as they appear in some teaching plan for C of order r. In particular, for every $s = 1, \ldots, n$, we have $\mathrm{TD}(c_s, \{c_s, \ldots, c_n\}) \leq r$.

We will show that the vector space $\mathbf{F}_2^n$ is spanned by the vectors that correspond to the monomials from P_m^r. The theorem then follows from a well-known linear algebra fact that the size of a spanning set cannot be smaller than the dimension of the vector space.

We will show by induction that each c_s lies in the span of P_m^r. Since $\mathrm{TD}(c_1, C) \leq r$, by Theorem 1, c_1 is equal to a polynomial of the form $p_{i_1} \cdots p_{i_k}$ for some $k \leq r$, where each p_t is equal to x_t or $x_t + 1$. It is not hard to see that the product $p_{i_1} \cdots p_{i_k}$ lies in the span of P_m^r, e.g., $(x_1 + 1)(x_2 + 1) = x_1 x_2 + x_1 + x_2 + 1$, etc.

Now suppose that $c_1, \ldots, c_s$ are in the span of P_m^r. Let $\mathbf{F}_2^{s,0}$ be the subspace of $\mathbf{F}_2^n$ consisting of the vectors whose the last $n - s$ coordinates are zeros. Similarly, let $\mathbf{F}_2^{0,n-s}$ be the subspace of $\mathbf{F}_2^n$ consisting of the vectors whose the first s coordinates are zeros. Also, let $(v)_{s,0}$ and $(v)_{0,n-s}$ be the projections of a vector $v \in \mathbf{F}_2^n$ to the subspaces $\mathbf{F}_2^{s,0}$ and $\mathbf{F}_2^{0,n-s}$, respectively. In particular, we have $v = (v)_{s,0} + (v)_{0,n-s}$.

Since $\mathrm{TD}(c_{s+1}, \{c_{s+1}, \ldots, c_n\}) \leq r$, applying Theorem 1 to $\{c_{s+1}, \ldots, c_n\}$ and c_{s+1} yields that $(c_{s+1})_{0,n-s} = (p_{i_1} \cdots p_{i_k})_{0,n-s}$ for some $k \leq r$ and some $i_1, \cdots, i_k$, where each p_t is equal to x_t or $x_t + 1$. In other words, $(c_{s+1} - p_{i_1} \cdots p_{i_k})_{0,n-s} = \mathbf{0}$, which means that $c_{s+1} - p_{i_1} \cdots p_{i_k}$ belongs to the subspace $\mathbf{F}_2^{s,0}$. As before, the product $p_{i_1} \cdots p_{i_k}$ lies in the span of P_m^r. Moreover, by the induction hypothesis, the vectors $c_1, \ldots, c_s$ are in the span of P_m^r, and hence the subspace $\mathbf{F}_2^{s,0}$ is contained in the span of P_m^r. Hence c_{s+1} lies in the span of P_m^r. □

The Sauer-type bound in Theorem 2 is tight for any r and m, in particular, it is met by all VCD-maximum classes of VC-dimension r. This suggests the following definition.

Definition 1. Let $C \subseteq 2^X$, $|X| = m$, and $\mathrm{RTD}(C) = r$. C is called RTD-*maximum* if $|C| = \Phi_r(m)$, and C is called RTD-*maximal* if $\mathrm{RTD}(C \cup \{c\}) > r$ for any concept $c \notin C$.

RTD-maximum classes have the following properties.

Proposition 1. (i) *Every* VCD-*maximum class C is also* RTD-*maximum with* $\mathrm{RTD}(C) = \mathrm{VCD}(C)$.

(ii) *There are* RTD-*maximum classes that are not* VCD-*maximum.*

(iii) *There is a class C for which both C and $\overline{C}$ are* RTD-*maximum, but neither C nor $\overline{C}$ is* VCD-*maximum.*

(iv) *There are* RTD-*maximum classes whose restrictions are not* RTD-*maximum. Furthermore, there is an* RTD-*maximum class C that has an* RTD-*maximum restriction C' such that* $\mathrm{RTD}(C') > \mathrm{RTD}(C)$.

Proof. (i) For every VCD-maximum class C, $\mathrm{RTD}(C) = \mathrm{VCD}(C)$ [2]. It follows from Theorem 2 and Definition 1 that C is RTD-maximum.

(ii) If an RTD-maximum class C is not VCD-maximum, then $\mathrm{RTD}(C) < \mathrm{VCD}(C)$. Table 1 shows an RTD-maximum class C_1 with $\mathrm{RTD}(C_1) = 2$ and $\mathrm{VCD}(C_1) = 3$.

(iii) C_1 in Table 1 is RTD-maximum with $\mathrm{RTD}(C_1) = 2$, and $\overline{C_1}$ is RTD-maximum with $\mathrm{RTD}(\overline{C_1}) = 1$. As $\mathrm{VCD}(C_1) = 3$ and $\mathrm{VCD}(\overline{C_1}) = 2$, neither C_1 nor $\overline{C_1}$ is VCD-maximum.

(iv) C_2 in Table 1 is RTD-maximum and $\mathrm{RTD}(C_2) = 1$, however, $\mathrm{RTD}(C_2 - x_4) = 2$ and $C_2 - x_4$ is not RTD-maximum. Furthermore, consider the RTD-maximum class C_1 in Table 1. Clearly, $C_1 - x_4$ is RTD-maximum and $\mathrm{RTD}(C_1) = 2 < \mathrm{RTD}(C_1 - x_4) = 3$. □

A consequence of the proof of Theorem 2 is that, for RTD-maximum classes, all instance sets of size $\mathrm{RTD}(C)$ are used as recursive teaching sets.

Table 1. C_1 and $\overline{C_1}$ are RTD-maximum but neither C_1 nor $\overline{C_1}$ is VCD-maximum. C_2 is RTD-maximum but $C_2 - x_4$ is not.

$c_i \in C_1$	x_1	x_2	x_3	x_4
c_1	0	0	0	0
c_2	1	0	0	0
c_3	0	1	0	0
c_4	0	0	1	0
c_5	0	0	0	1
c_6	1	1	0	0
c_7	1	0	1	0
c_8	0	1	1	0
c_9	0	1	0	1
c_{10}	0	0	1	1
c_{11}	1	1	1	1

$c_i \in \overline{C_1}$	x_1	x_2	x_3	x_4
c_1	1	0	0	1
c_2	1	1	1	0
c_3	1	1	0	1
c_4	1	0	1	1
c_5	0	1	1	1

$c_i \in C_2$	x_1	x_2	x_3	x_4
c_1	0	0	0	0
c_2	1	0	0	0
c_3	0	1	0	0
c_4	0	0	1	0
c_5	0	1	1	1

Corollary 1. *Let $C \subseteq 2^X$ be* RTD*-maximum, $|X| = m$, and* $\mathrm{RTD}(C) = r$*. Let $X' \subseteq X$ be any subset of size r. Then for any teaching plan P for C of order r, there is a concept $c \in C$ and a recursive teaching set S for c with respect to P, such that $X(S) = X'$.*

Proof. Let $X' = \{x_{i_1}, \ldots, x_{i_r}\}$, and P be a teaching plan for C of order r such that $c_1, c_2, \ldots, c_n$ are all concepts from C listed in the same order as they appear in P. Assume that X' does not appear as a recursive teaching set in the plan P. Then, in the proof of Theorem 2 we can always represent the concept c_{s+1} inside the class $\{c_{s+1}, \ldots, c_n\}$ as a polynomial $f(z_1, \ldots, z_r)$ over $\mathbf{F}_2$ such that $\{z_1, \ldots, z_r\} \neq \{x_{i_1}, \ldots, x_{i_r}\}$. (This follows from Theorem 1 and the fact that X' is not used as a recursive teaching set.) As a consequence, we can span $\mathbf{F}_2^n$ without using the monomial $x_{i_1} \cdots x_{i_r}$, which implies that $|C| = \dim(\mathbf{F}_2^n) \leq \Phi_r(m) - 1$. Hence C is not RTD-maximum. This is a contradiction. □

Another corollary of Theorem 2 is that for an RTD-maximum class, teaching sets of size 1 cannot be used too early in any teaching plan.

Corollary 2. *Let $C \subseteq 2^X$ be* RTD*-maximum, $|X| = m$, and* $\mathrm{RTD}(C) = r$*. For an arbitrary teaching plan for C, let $(c_1, c_2, \ldots, c_n)$ be the sequence of all concepts of C listed in the plan. Then for any positive integer $i < \Phi_{r-1}(m-1)$, we have* $\mathrm{TD}(c_i, \{c_i, \ldots, c_n\}) > 1$*.*

Proof. Assume there is a teaching plan for C such that $\mathrm{TD}(c_i, \{c_i, \ldots, c_n\}) = 1$ for some $i < \Phi_{r-1}(m-1)$. Let $(x, \ell) \in \mathrm{TS}(c_i, \{c_i, \ldots, c_n\})$ for some $x \in X$ and $\ell \in \{0, 1\}$. Then for any $c \in \{c_{i+1}, \ldots, c_n\}$, $c(x) = \bar{\ell}$. So, $|\{c_{i+1}, \ldots, c_n\}| = |\{c_{i+1}, \ldots, c_n\}|_{X \setminus \{x\}}|$. Consequently,

$$\begin{aligned} |C| &= |\{c_1, \ldots, c_i\}| + |\{c_{i+1}, \ldots, c_n\}| = i + |\{c_{i+1}, \ldots, c_n\}| \\ &= i + |\{c_{i+1}, \ldots, c_n\}|_{X \setminus \{x\}}| \ \leq \ i + \Phi_r(m-1), \text{ by Theorem 2} \\ &< \Phi_{r-1}(m-1) + \Phi_r(m-1) = \Phi_r(m). \end{aligned}$$

Thus C is not RTD-maximum. This is a contradiction. □

As mentioned in Section 1, the complement of any VCD-maximum class is VCD-maximum. RTD-maximum classes do not possess this property.

Proposition 2. *There is an* RTD*-maximum class whose complement is not* RTD*-maximum.*

Proof. Consider the RTD-maximum class C with $\mathrm{RTD}(C) = 3$ in Table 2. $\overline{C}$ is not RTD-maximum because $\mathrm{RTD}(\overline{C}) = 2$ and $6 < \Phi_2(5)$. □

Still, the complement of an RTD-maximum class of RTD 1 is RTD-maximum.

Proposition 3. *Let C be an* RTD*-maximum class over X with $|X| \geq 2$. If* $\mathrm{RTD}(C) = 1$*, then $\overline{C}$ is* RTD*-maximum and* $\mathrm{RTD}(\overline{C}) = |X| - 2$*.*

Table 2. C is RTD-maximum (recursive teaching sets are underlined), but $\overline{C}$ is not

$c_i \in C$	x_1	x_2	x_3	x_4	x_5
c_1	$\underline{1}$	$\underline{1}$	$\underline{1}$	1	1
c_2	1	1	$\underline{0}$	$\underline{1}$	$\underline{1}$
c_3	$\underline{1}$	$\underline{1}$	0	$\underline{1}$	0
c_4	$\underline{1}$	$\underline{1}$	0	0	$\underline{1}$
c_5	0	$\underline{1}$	1	$\underline{1}$	$\underline{1}$
c_6	$\underline{1}$	0	1	$\underline{1}$	$\underline{1}$
c_7	0	0	1	$\underline{1}$	$\underline{1}$
c_8	$\underline{1}$	$\underline{1}$	0	0	0
c_9	$\underline{1}$	0	$\underline{1}$	0	$\underline{1}$
c_{10}	$\underline{1}$	0	0	0	$\underline{1}$
c_{11}	0	$\underline{1}$	$\underline{1}$	$\underline{1}$	0
c_{12}	0	$\underline{1}$	0	$\underline{1}$	0
c_{13}	0	$\underline{1}$	$\underline{1}$	0	$\underline{1}$

$c_i \in C$	x_1	x_2	x_3	x_4	x_5
c_{14}	0	$\underline{1}$	0	0	$\underline{1}$
c_{15}	$\underline{1}$	0	$\underline{1}$	$\underline{1}$	0
c_{16}	$\underline{1}$	0	0	$\underline{1}$	0
c_{17}	0	$\underline{1}$	$\underline{1}$	0	0
c_{18}	0	$\underline{1}$	0	0	0
c_{19}	0	0	$\underline{1}$	$\underline{1}$	0
c_{20}	0	0	0	$\underline{1}$	0
c_{21}	$\underline{1}$	0	$\underline{1}$	0	0
c_{22}	$\underline{1}$	0	0	0	0
c_{23}	0	0	$\underline{1}$	0	$\underline{1}$
c_{24}	0	0	$\underline{1}$	0	0
c_{25}	0	0	0	0	$\underline{1}$
c_{26}	0	0	0	0	0

$c_i \in \overline{C}$	x_1	x_2	x_3	x_4	x_5
c_1	0	0	0	1	1
c_2	0	1	0	1	1
c_3	1	0	0	1	1
c_4	1	1	1	0	0
c_5	1	1	1	0	1
c_6	1	1	1	1	0

Proof. By induction on $|X|$. For $|X| = 2$ the proof is trivial. Suppose that for $|X| < m$ the statement of the theorem is true. Now consider the case $|X| = m > 2$. Let $c_1 \in C$ with $\mathrm{TD}(c_1, C) = 1$, and w.l.o.g., let $\{(x_1, 1)\}$ be a teaching set for c_1 in C. Then we can write C as a disjoint union of $\{c_1\}$ and $\{0\} \times C_1$, where $C_1 = (C \setminus \{c_1\}) - x_1$ is a maximum class of $\mathrm{RTD}(C_1) = 1$ on $X \setminus \{x_1\}$. So, the complement of C is equal to the disjoint union $\overline{C} = (\{0\} \times \overline{C_1}) \cup (\{1\} \times C_2)$, where $C_2 = 2^{X \setminus \{x_1\}} \setminus \{c_1 - x_1\}$ is a class of size $2^{m-1} - 1$ on $X \setminus \{x_1\}$.

By the induction hypothesis, there is a teaching plan of order $m - 3$ for $\overline{C_1}$. Take such a plan and extend every recursive teaching set S from this plan to $S \cup \{(x_1, 0)\}$. As a result, we obtain a teaching plan for $\{0\} \times \overline{C_1}$ of order $m - 2$, which we call P_1. Note that C_2 is a VCD-maximum class with $\mathrm{VCD}(C_2) = |X \setminus \{x_1\}| - 1 = m - 2$, and hence $\mathrm{RTD}(C_2) = m - 2$. Since $\mathrm{RTD}(\{1\} \times C_2) = \mathrm{RTD}(C_2)$, there is a teaching plan of order $m - 2$ for $\{1\} \times C_2$, which we call P_2.

Every recursive teaching set from P_1 contains $(x_1, 0)$, which distinguishes the concepts in $\{0\} \times \overline{C_1}$ from those in $\{1\} \times C_2$. So, P_1 and P_2 can be merged to a teaching plan for $\overline{C}$ of order $m - 2$. Thus $\mathrm{RTD}(\overline{C}) \leq m - 2$. Further, $|\overline{C}| = 2^m - |C| = 2^m - (m + 1) = \Phi_{m-2}(m)$. Hence, by Theorem 2, $\mathrm{RTD}(\overline{C}) = m - 2$, and $\overline{C}$ is RTD-maximum. □

The RTD-maximum class C in the proof of Proposition 2 fulfills $\mathrm{RTD}(C) + \mathrm{RTD}(\overline{C}) = |X|$. In contrast to this, note that a class C is VCD-maximum if and only if $\mathrm{VCD}(C) + \mathrm{VCD}(\overline{C}) = |X| - 1$. Necessity of the condition was proven by Rubinstein et al. [6]. Sufficiency is easy to see, as was pointed out by an anonymous reviewer of this paper: Suppose C with $\mathrm{VCD}(C) = d$ is not VCD-maximum. Then $|C| < \Phi_d(|X|)$ and thus $|\overline{C}| > 2^{|X|} - \Phi_d(|X|) = \Phi_{|X|-d-1}(|X|)$, which implies $\mathrm{VCD}(\overline{C}) > |X| - d - 1$. The same reasoning implies that the condition is sufficient as well when VCD is replaced by RTD throughout.

Proposition 4. *Let $C \subseteq 2^X$ and $|X| = m$. If* $\mathrm{RTD}(C) + \mathrm{RTD}(\overline{C}) = m - 1$, *then C is* RTD-*maximum.*

Recall that $\mathrm{RTD}^*(C) = \max_{X' \subseteq X} \mathrm{RTD}(C|_{X'})$. We obtain the following property.

Proposition 5. *Let $C \subseteq 2^X$ and $|X| = m$. If* $\mathrm{RTD}^*(C) \leq r$, *then $|C| \leq \Phi_r(m)$. The inverse statement is not true in general.*

Proof. Since $\mathrm{RTD}^*(C) \leq r$, $\mathrm{RTD}(C) \leq r$ and by Theorem 2, $|C| \leq \Phi_r(m)$. An example[2] for a class C with $|C| \leq \Phi_r(m)$ and $\mathrm{RTD}^*(C) > \mathrm{RTD}(C) = r$ is the class $C = \{\emptyset,\ \{x_2, x_3\},\ \{x_1, x_3\},\ \{x_1, x_2, x_3\}\}$, for which $|C| = 4$, $\mathrm{RTD}(C) = 1$ and $\mathrm{RTD}^*(C) = 2$. □

5 RTD-Maximal Classes

In this section we present some properties of RTD-maximal classes. We first show that an RTD-maximal class shatters each subset of the instance space whose size is equal to RTD.

Proposition 6. *Let $C \subseteq 2^X$ be* RTD*-maximal with* $\mathrm{RTD}(C) = r$. *Then, for any subset $X' \subseteq X$ with $|X'| = r$, C shatters X'.*

Proof. Assume that X' is not shattered by C. Then $|C|_{X'}| < 2^{|X'|}$ and we can add a new concept c_{new} to C such that $c_{new}|_{X'} \notin C|_{X'}$. Thus, $\mathrm{TD}(c_{new}, C \cup \{c_{new}\}) \leq r$. Since $\mathrm{RTD}(C) = r$, C has a teaching plan of order r. So, $C \cup \{c_{new}\}$ also has a teaching plan of order r, which starts with c_{new} and then continues with any teaching plan for C of order r. Therefore, $\mathrm{RTD}(C \cup \{c_{new}\}) \leq r$ and C is not RTD-maximal. □

As a corollary we obtain that for an RTD-maximal class, the minimal and the recursive teaching dimensions coincide.

Corollary 3. *For any* RTD*-maximal class $C \subseteq 2^X$,* $\mathrm{TD}_{min}(C) = \mathrm{RTD}(C)$.

Proof. $\mathrm{TD}_{min}(C) \leq \mathrm{RTD}(C)$ is easy to see. Assume $\mathrm{TD}_{min}(C) < \mathrm{RTD}(C)$. Then, there is a concept $c \in C$ for which $\{x_{i_1}, \ldots, x_{i_k}\}$ is a teaching set, for some $k < \mathrm{RTD}(C)$. Consider any subset $X' \subseteq X$ such that $|X'| = \mathrm{RTD}(C)$ and $\{x_{i_1}, \ldots, x_{i_k}\} \subset X'$. Then C does not shatter X', since otherwise there would exist at least one more concept $c' \in C$ with $c'|_{\{x_{i_1}, \ldots, x_{i_k}\}} = c|_{\{x_{i_1}, \ldots, x_{i_k}\}}$. This is impossible because $\{x_{i_1}, \ldots, x_{i_k}\}$ is a teaching set for c in C. Hence, by Proposition 6, C cannot be RTD-maximal. This is a contradiction. □

It is not hard to see that VCD-maximal classes of VC-dimension 1 are VCD-maximum. We now show that the same holds for RTD-maximal classes.

Proposition 7. *Let $C \subseteq 2^X$ be* RTD*-maximal. If* $\mathrm{RTD}(C) = 1$, *then C is* RTD*-maximum.*

[2] This example also provides a simpler proof of the second part of Proposition 1(iv). The latter in turn implies that the inverse of Proposition 5 is not true in general.

Proof. By induction on the size of X. For $|X| = 1$ there is only one RTD-maximal class with two concepts which is clearly RTD-maximum. Suppose that the theorem holds when $|X| = m$. Now we consider the case that $|X| = m+1$ and C is an RTD-maximal class on X with $\mathrm{RTD}(C) = 1$. Since $\mathrm{RTD}(C) = 1$, there is a concept $c \in C$ such that $\mathrm{TD}(c, C) = 1$. Let (x, ℓ) be a teaching set for c. Then, for any $c' \in C \setminus \{c\}$, $(x, \ell) \notin c'$ or equivalently, $(x, \bar{\ell}) \in c'$, which implies that $|C \setminus \{c\}| = |(C \setminus \{c\}) - x|$. Clearly, $(C \setminus \{c\}) - x$ is RTD-maximal, otherwise C would not be RTD-maximal. So, by the induction hypothesis, $|(C \setminus \{c\}) - x| = \Phi_1(m)$. Therefore, $|C| = \Phi_1(m) + 1 = \Phi_1(m+1)$ and C is RTD-maximum. □

Surprisingly, not all RTD-maximal classes are RTD-maximum.

Proposition 8. (Doliwa [1]) *There is an* RTD-*maximal class that is not* RTD-*maximum.*

Proof. Consider the RTD-maximal class C in Table 3. Since $\mathrm{RTD}(C) = 3$ and $|C| = 40 < \Phi_3(6)$, C is not RTD-maximum. □

Table 3. RTD-maximal class that is not RTD-maximum

c_i	x_1	x_2	x_3	x_4	x_5	x_6	c_i	x_1	x_2	x_3	x_4	x_5	x_6	c_i	x_1	x_2	x_3	x_4	x_5	x_6	c_i	x_1	x_2	x_3	x_4	x_5	x_6
c_1	0	1	0	1	1	0	c_{11}	1	0	1	1	1	1	c_{21}	0	1	0	0	1	0	c_{31}	0	0	0	0	0	0
c_2	0	1	1	1	0	1	c_{12}	0	0	1	0	0	0	c_{22}	1	1	0	1	1	0	c_{32}	1	1	0	1	0	1
c_3	1	0	0	0	0	0	c_{13}	1	1	1	0	0	1	c_{23}	1	0	0	0	1	0	c_{33}	0	0	0	1	0	0
c_4	1	0	0	1	1	1	c_{14}	0	1	1	0	1	0	c_{24}	1	1	0	1	1	1	c_{34}	0	0	0	0	1	0
c_5	0	0	1	1	0	0	c_{15}	1	0	1	0	1	1	c_{25}	1	1	0	0	1	1	c_{35}	1	1	0	0	0	0
c_6	1	0	0	1	1	0	c_{16}	0	0	1	1	0	1	c_{26}	0	1	0	0	0	0	c_{36}	1	0	1	0	1	0
c_7	0	0	1	0	1	1	c_{17}	1	1	1	1	0	0	c_{27}	1	0	0	0	0	1	c_{37}	0	1	0	0	0	1
c_8	1	1	1	0	1	0	c_{18}	1	1	1	0	1	1	c_{28}	0	1	0	1	0	1	c_{38}	1	1	1	1	1	0
c_9	0	1	1	0	0	1	c_{19}	0	0	1	1	1	0	c_{29}	0	1	1	1	1	0	c_{39}	1	1	1	1	0	1
c_{10}	1	0	1	0	0	0	c_{20}	1	1	1	1	1	1	c_{30}	1	1	0	0	1	0	c_{40}	0	1	1	0	0	0

6 Algebraic Proof of Shortest-Path-Closedness of VCD-Maximum Classes

In this section, we give an example of how the algebraic techniques applied to obtain our main result can also yield more elegant and insightful proofs for already known results. Our example is the proof showing that VCD-maximum classes are shortest-path-closed.

A shortest-path-closed class is a class C in which any two concepts c, c' are Hamming-connected, i.e., there are pairwise distinct instances $x_1, \ldots, x_k$ and $c_1, \ldots, c_{k-1} \in C$ such that, with $c_0 = c$ and $c_k = c'$, the concepts c_{i-1} and c_i differ only in x_i, for $1 \leq i \leq k$. It is known that VCD-maximum classes are shortest-path-closed [5], but algebraic methods provide an elegant alternative proof.

For $Z \subseteq X = \{x_1, \ldots, x_m\}$ and $t \leq m$, let $P_m^t(Z)$ be the collection of monomials over $\mathbf{F}_2$ of the form $x_{i_1} \cdots x_{i_k}$ such that $0 \leq k \leq t$, $1 \leq i_1 < \cdots < i_k \leq m$ and $\{x_{i_1}, \ldots, x_{i_k}\} \subseteq Z$.

Lemma 1. *Let $|X| = m$, $C \subseteq 2^X$, and* $\mathrm{VCD}(C) = d$. *A set of instances $Z \subseteq X$ is a teaching set for $c \in C$ if and only if c is in the span of $P_m^d(Z)$.*

Proof. Suppose $Z \subseteq X$ is a teaching set for $c \in C$. Then, by Theorem 1, $c = f$ for some polynomial f over $\mathbf{F}_2$ whose variables are in the set Z. Each such polynomial is equal to a linear combination of monomials from $P_m^t(Z)$, where $t = |Z|$. For instance, $(x_1 + 1)(x_2 + 1)x_3 = x_1x_2x_3 + x_1x_3 + x_2x_3 + x_3$, etc.

We show that, for every $t \leq m$ and $Z \subseteq X$, the monomials from $P_m^t(Z)$ are in the span of $P_m^d(Z)$. This in turn implies that f is in the span of $P_m^d(Z)$.

As in [10], we use induction on t: If $t \leq d$, there is nothing to prove. Suppose $t > d$ and every monomial from $P_m^{t-1}(Z)$ is in the span of $P_m^d(Z)$. Consider a monomial $x_{i_1} \cdots x_{i_t}$ from $P_m^t(Z)$. Since $t > d$, the set $\{x_{i_1}, \ldots, x_{i_t}\}$ is not shattered by C. Let $(a_1, \ldots, a_t)$ be a concept that is not in $C|_{\{x_{i_1},\ldots,x_{i_t}\}}$ and consider a polynomial $p(x_{i_1}, \ldots, x_{i_t}) = (x_{i_1} + a_1 + 1)(x_{i_2} + a_2 + 1) \cdots (x_{i_t} + a_t + 1)$.

As a vector in $\mathbf{F}_2^{|C|}$, p has zero coordinates because $p(c(x_{i_1}), \ldots, c(x_{i_t})) = 0$ for all $c \in C$ as at least one of the factors of p will be zero. Hence $p = \mathbf{0}$ and $x_{i_1} \cdots x_{i_t}$ can be expressed as a linear combination of monomials of smaller degree with coefficients from $\{x_{i_1}, \ldots, x_{i_t}\} \subseteq Z$, that is, the ones from $P_m^{t-1}(Z)$. To see this, consider, e.g., $(x_1+1)(x_2+1)x_3 = \mathbf{0}$; then we have $x_1x_2x_3 = x_1x_3+x_2x_3+x_3$. By the inductive hypothesis, $P_m^{t-1}(Z)$ is in the span of $P_m^d(Z)$, and hence $x_{i_1} \cdots x_{i_t}$ is in the span of $P_m^d(Z)$. So $P_m^t(Z)$ is in the span of $P_m^d(Z)$.

The implication in the other direction follows from Theorem 1. □

Theorem 3. *If C is a* VCD*-maximum class, then C is shortest-path-closed.*

Proof. In this proof, we use the symbol $\triangle$ to denote symmetric difference.

Let $C \subseteq 2^X$ be a VCD-maximum class with $|X| = m$ and $\mathrm{VCD}(C) = d$, and let $I(c)$ denote the set $\{x \in X \mid \text{there exists a } c' \in C \text{ such that } c \triangle c' = \{x\}\}$. We first show that, for every $c \in C$, $I(c)$ is a teaching set for c. By Theorem 1, the monomials from $P_m^d(X)$ span the vector space $\mathbf{F}_2^{|C|}$. Since $|P_m^d(X)| = \Phi_d(m) = |C|$, the set $P_m^d(X)$ is a basis for $\mathbf{F}_2^{|C|}$.

Let $c \in C$ and let $S \subseteq X$ be a minimal teaching set for c in the sense that no proper subset of S is a teaching set for c. Suppose $I(c) \neq S$ and let $x \in S \setminus I(c)$. By Lemma 1, there is a linear combination f_1 of monomials from $P_m^d(S)$ such that $c = f_1$. Note that $X \setminus \{x\}$ is also a teaching set for c, since otherwise $x \in I(c)$. Thus, there is a linear combination f_2 of monomials from $P_m^d(X \setminus \{x\})$ with $c = f_2$. Since $P_m^d(X)$ is a basis for $\mathbf{F}_2^{|C|}$, we have $f_1 = f_2$. As f_2 does not depend on x, f_1 does not depend on x either. Thus f_1 depends only on variables from $S \setminus \{x\}$. By Lemma 1, $S \setminus \{x\}$ is a teaching set for c, which contradicts the minimality of S. Therefore $S = I(c)$, and thus $I(c)$ is a teaching set for c.

Finally, we prove that any two concepts c_1 and c_2 in C are Hamming-connected, by induction on $|c_1 \triangle c_2|$. For $|c_1 \triangle c_2| = 1$ the proof is obvious. Suppose $|c_1 \triangle c_2| = n$ and any two concepts c, c' with $|c \triangle c'| < n$ are Hamming-connected. Since $I(c_1)$ is a teaching set for c_1, it cannot be disjoint from $c_1 \triangle c_2$. Hence there is an $x \in I(c_1) \cap (c_1 \triangle c_2)$. Let c' be the concept from C such that $c_1 \triangle c' = \{x\}$. Then $|c' \triangle c_2| = n - 1$ and by the inductive hypothesis c' and c_2 are Hamming-connected. Therefore, c_1 and c_2 are Hamming-connected. □

7 Conclusions

Our analog of Sauer's bound for RTD establishes a new connection between teaching complexity and VC-dimension. A main contribution besides obtaining this result is the successful application of algebraic proof techniques. The characterization of teaching sets obtained this way is of potential use for future studies not only in the context of the combinatorial questions we asked in this paper.

Our results on RTD-maximum and RTD-maximal classes provide deep insights into structural properties that affect the complexity of teaching a concept class. As a byproduct of our studies, we proved several new results on VCD-maximal classes. Altogether, our results might be helpful in solving the long-standing sample compression conjecture [3] and in establishing further connections between learning from a teacher and learning from randomly chosen examples. In particular, we hope that methods from algebra will turn out to be of further use in these contexts.

Acknowledgements. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). Pavel Semukhin was also supported in part by a postdoctoral fellowship from the Pacific Institute for the Mathematical Sciences.

References

[1] Doliwa, T.: Personal communication (2011)
[2] Doliwa, T., Simon, H.U., Zilles, S.: Recursive Teaching Dimension, Learning Complexity, and Maximum Classes. In: Hutter, M., Stephan, F., Vovk, V., Zeugmann, T. (eds.) ALT 2010. LNCS, vol. 6331, pp. 209–223. Springer, Heidelberg (2010)
[3] Floyd, S., Warmuth, M.K.: Sample compression, learnability, and the Vapnik-Chervonenkis dimension. Machine Learning 21(3), 269–304 (1995)
[4] Goldman, S.A., Kearns, M.J.: On the complexity of teaching. Journal of Computer and System Sciences 50, 20–31 (1995)
[5] Kuzmin, D., Warmuth, M.K.: Unlabeled compression schemes for maximum classes. J. Mach. Learn. Res. 8, 2047–2081 (2007)
[6] Rubinstein, B.I.P., Bartlett, P.L., Rubinstein, J.H.: Shifting: One-inclusion mistake bounds and sample compression. J. Comput. Syst. Sci. 75(1), 37–59 (2009)
[7] Sauer, N.: On the density of families of sets. J. Comb. Theory, Ser. A 13(1), 145–147 (1972)
[8] Shelah, S.: A combinatorial problem: Stability and order for models and theories in infinitary languages. Pac. J. Math. 4, 247–261 (1972)
[9] Shinohara, A., Miyano, S.: Teachability in computational learning. New Generation Comput. 8(4), 337–347 (1991)
[10] Smolensky, R.: Well-known bound for the VC-dimension made easy. Computational Complexity 6(4), 299–300 (1997)
[11] Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. Theory Probab. Appl. 16, 264–280 (1971)
[12] Welzl, E.: Complete range spaces. Unpublished notes (1987)
[13] Zilles, S., Lange, S., Holte, R., Zinkevich, M.: Models of cooperative teaching and learning. Journal of Machine Learning Research 12, 349–384 (2011)

Appendix: VCD-Maximal Classes

This appendix contains some new interesting properties of VCD-maximal classes. For instance, the next theorem provides a way of constructing an infinite series of equal-sized maximal classes starting from a given maximal class.

Theorem 4. *Let C be a class of VC-dimension d on a set of m instances $X = \{x_1, \dots, x_m\}$.*

(1) If C is a maximal class and for some instance $x \in X$ we have $|C - x| = |C|$, then $C + x$ is also maximal, where

$$C + x = \{c \in 2^{X \cup \{x_{m+1}\}} : c \cap X \in C \text{ and } c(x_{m+1}) = c(x)\}.$$

This process can be continued to obtain a series of maximal classes $C + x$, $(C + x) + x$, $((C + x) + x) + x$, etc.

(2) If $|C - x| < |C|$, then $C + x$ is not a maximal class.

Proof. (1) Note that $\mathrm{VCD}(C) = \mathrm{VCD}((C+x)-x)$ and $\mathrm{VCD}(C) = \mathrm{VCD}(C+x)$. These equalities follow from the fact that C is equivalent to $(C + x) - x$, and that if $C + x$ shatters a set S, then S cannot contain both x and x_{m+1}.

Suppose C is maximal and $|C - x| = |C|$ for some $x \in X$. Consider any $c \in 2^{X \cup \{x_{m+1}\}}$ such that $c \notin C + x$ and let $c - x_{m+1} = c \cap X$. We need to show that $\mathrm{VCD}(C+x\cup\{c\}) > \mathrm{VCD}(C+x)$. First, suppose $c-x_{m+1} \notin C$. Then, since C is maximal, $\mathrm{VCD}(C+x\cup\{c\}) \geq \mathrm{VCD}(C\cup\{c-x_{m+1}\}) > \mathrm{VCD}(C) = \mathrm{VCD}(C+x)$.

Now suppose $c - x_{m+1} \in C$. In this case $c(x) \neq c(x_{m+1})$ since otherwise $c \in C + x$. Also note that the concept $c - x = c \cap (X \cup \{x_{m+1}\} - x)$ does not belong to $(C + x) - x$. Indeed, suppose $c - x \in (C + x) - x$ and let $c' \in C$ be the image of $c - x$ under the equivalence transformation from $(C + x) - x$ to C. We then have that C contains two concepts, namely $c - x_{m+1}$ and c', that differ only on x since $(c - x_{m+1})(x) = c(x) \neq c(x_{m+1}) = (c - x)(x_{m+1}) = c'(x)$. This contradicts the assumption that $|C - x| = |C|$. Therefore, $c - x \notin (C + x) - x$ and we have that $\mathrm{VCD}(C+x\cup\{c\}) \geq \mathrm{VCD}((C+x)-x\cup\{c-x\}) > \mathrm{VCD}((C+x)-x) = \mathrm{VCD}(C) = \mathrm{VCD}(C + x)$. Hence $C + x$ is a maximal class.

(2) If $|C - x| < |C|$ then there are two concepts c_1 and c_2 in C that differ only in x. Consider a concept $c \notin C + x$ defined as $c = c_1 \cup \{(x_{m+1}, \ell)\}$ where ℓ is chosen so that $c(x) \neq c(x_{m+1})$. Since c coincides with c_1 on X, we have $(C + x \cup \{c\}) - x_{m+1} = C$. Furthermore, c coincides with the extension of c_2 in $C+x$ on the instances from $(X\cup\{x_{m+1}\})-x$. Hence $(C+x\cup\{c\})-x = (C+x)-x$, which is, of course, equivalent to C.

Let $\mathrm{VCD}(C + x) = d$ and suppose that $C + x \cup \{c\}$ shatters a set S of size $d + 1$. Note that S cannot contain both x and x_{m+1} since the restriction of $C + x \cup \{c\}$ to these two instances can contain only one of the two concepts $(0, 1)$ and $(1, 0)$. If S does not contain x_{m+1}, then we have $\mathrm{VCD}(C + x) = \mathrm{VCD}(C) = \mathrm{VCD}((C + x \cup \{c\}) - x_{m+1}) \geq d + 1$. On the other hand, if S does not contain x, we have $\mathrm{VCD}(C + x) = \mathrm{VCD}((C + x) - x) = \mathrm{VCD}((C + x \cup \{c\}) - x) \geq d + 1$. These contradictions show that in fact $\mathrm{VCD}(C + x \cup \{c\}) = \mathrm{VCD}(C + x)$, and hence $C + x$ is not a maximal class. □

The following proposition by Rubinstein et al. [6] follows immediately from the definition of VC-dimension.

Proposition 9. $\mathrm{VCD}(C) \le d$ *if and only if* $\overline{C}$ *contains at least one* $(m-d-1)$*-cube for each subset of* $(m-d-1)$ *instances, i.e.,* $\overline{C}^S \neq \emptyset$ *for every subset* S *of* $m-d-1$ *instances.*

We now establish a non-trivial lower bound for the size of VCD-maximal classes and show that this bound can be met when *both* VCD and $|X|$ are large.

Theorem 5. *Let* $C \subseteq 2^X$ *be a* VCD*-maximal class over a set* X *with* $|X| = m$*. If* $\mathrm{VCD}(C) = d$*, then*

$$|C| \ge 2^m - 2^{m-d-1}\binom{m}{d+1}.$$

Equivalently, if $\mathrm{VCD}(C) = m - d$*, then*

$$|C| \ge 2^m - 2^{d-1}\binom{m}{d-1}.$$

This lower bound can be met when $m \gg d$*, that is, when* $|X| - \mathrm{VCD}(C)$ *is small compared to* $|X|$*.*

Proof. We prove the second inequality. Suppose $\mathrm{VCD}(C) = m - d$ and $|C| < 2^m - 2^{d-1}\binom{m}{d-1}$. In this case, we have that $|\overline{C}| > 2^{d-1}\binom{m}{d-1}$. By Proposition 9, $\overline{C}$ must contain at least one $(d-1)$-cube for each subset of $d-1$ instances. Consider a union of $(d-1)$-cubes from $\overline{C}$ taking exactly one cube for each subset of instances of size $d-1$. Then the size of this union will be at most $2^{d-1}\binom{m}{d-1}$. Therefore, $\overline{C}$ must contain at least one concept c that does not belong to the above union of $(d-1)$-cubes. Hence, due to Proposition 9, we can add this concept c to the class C without increasing its VC-dimension, which contradicts the fact that C is maximal.

To show that the lower bound is exact for large m, we need to construct a disjoint union of $(d-1)$-cubes which consists of exactly one cube for each choice of $d-1$ instances; then the complement of such union will be a maximal class C with $\mathrm{VCD}(C) = m - d$ and $|C| = 2^m - 2^{d-1}\binom{m}{d-1}$. To do this, let us split the instance space X into disjoint blocks of size $2d$ and let $\{c_1, \ldots, c_N\}$ be the concepts that are equal to unions of such blocks. Note that $N = 2^{\lfloor m/2d \rfloor}$ and $|c_i \triangle c_j| \ge 2d$ for $i \neq j$. Now to each subset $S \subseteq X$ of size $d-1$, we assign a concept c_S from the above list such that $c_S \neq c_{S'}$ for $S \neq S'$. This can be done since for $m \gg d$, $N = 2^{\lfloor m/2d \rfloor}$ is greater than $\binom{m}{d-1}$, the number of all subsets of size $d-1$.

For each $S \subseteq X$ of size $d-1$, define a $(d-1)$-cube $C(S)$ based on c_S, that is, $C(S) = 2^S \times \{c_S|_{X \setminus S}\}$. Note that for $S \neq S'$, the cubes $C(S)$ and $C(S')$ are disjoint because, by construction, $|c_S \triangle c_{S'}| \ge 2d$. Therefore, the class C, defined as

$$C = 2^X \setminus \bigcup_{S \subseteq X:\ |S| = d-1} C(S),$$

is a maximal class of VC-dimension $m - d$ and size $2^m - 2^{d-1}\binom{m}{d-1}$. □

As a corollary we obtain that for a maximal class C with $\mathrm{VCD}(C) = |X| - O(1)$, the sum $\mathrm{VCD}(C) + \mathrm{VCD}(\overline{C})$ is bounded by $|X| + O(\log_2 |X|)$.

Theorem 6. *Let $|X| = m$. If $C \subseteq 2^X$ is a maximal class and $\mathrm{VCD}(C) = m - d$, then*

$$\mathrm{VCD}(C) + \mathrm{VCD}(\overline{C}) \leq m - 1 + (d-1)\log_2 m.$$

Proof. Since C is maximal, we have, by Theorem 5, that $|C| \geq 2^m - 2^{d-1}\binom{m}{d-1}$. Therefore, $|\overline{C}| \leq 2^{d-1}\binom{m}{d-1}$ and hence $\mathrm{VCD}(\overline{C}) \leq \log_2 |\overline{C}| \leq d - 1 + \log_2 \binom{m}{d-1}$. Taking into account that $\binom{m}{d-1} \leq m^{d-1}$, we obtain $\mathrm{VCD}(\overline{C}) \leq d - 1 + (d - 1)\log_2 m$. Since $\mathrm{VCD}(C) = m - d$, it follows that $\mathrm{VCD}(C) + \mathrm{VCD}(\overline{C}) \leq m - 1 + (d-1)\log_2 m$. □

Another property of VCD-maximal classes is that they are indecomposable in the sense that they cannot be formed by a direct product of non-trivial smaller classes.

Theorem 7. *Let $C_0 \subseteq 2^{X_0}$ and $C_1 \subseteq 2^{X_1}$ be nonempty concept classes with*

(a) $\mathrm{VCD}(C_0) > 0$ *or* $\mathrm{VCD}(C_1) > 0$ *and*
(b) $C_0 \times C_1 \neq 2^{X_0 \cup X_1}$.

Then $C_0 \times C_1$ is not a maximal class.

We will need to prove the following lemma first.

Lemma 2. *Let $C_0 \subseteq 2^{X_0}$ and $C_1 \subseteq 2^{X_1}$ be nonempty concept classes and let $c_0 \in 2^{X_0}$ and $c_1 \in 2^{X_1}$ be any two concepts with the property that for each $i \in \{0, 1\}$, if $\mathrm{VCD}(C_i) = 0$ then $\mathrm{VCD}(C_{1-i} \cup \{c_{1-i}\}) = \mathrm{VCD}(C_{1-i})$. Then $\mathrm{VCD}((C_0 \times C_1) \cup \{c_0c_1\}) = \mathrm{VCD}(C_0 \times C_1) = \mathrm{VCD}(C_0) + \mathrm{VCD}(C_1)$.*

Proof. Let $d_i = \mathrm{VCD}(C_i)$, for $i \in \{0, 1\}$, and suppose that $(C_0 \times C_1) \cup \{c_0c_1\}$ shatters a set $S \subseteq X_0 \cup X_1$ of size $d_0 + d_1 + 1$. Let $S_i = S \cap X_i$ and assume w.l.o.g. that $|S_0| = d_0 + 1$ and $|S_1| = d_1$. Therefore, $\mathrm{VCD}(C_0 \cup \{c_0\}) = d_0 + 1 > \mathrm{VCD}(C_0)$, and by the assumption we have that $d_1 > 0$. So, on the one hand, we have that $(C_0 \times C_1) \cup \{c_0c_1\}$ must contain at least $2^{d_1} > 1$ concepts that extend $c_0|_{S_0}$. But, on the other hand, $(C_0 \times C_1) \cup \{c_0c_1\}$ contains only one such concept, namely c_0c_1, since $c_0|_{S_0} \notin C_0|_{S_0}$. This contradiction proves the lemma. □

Proof (of Theorem 7). If $\mathrm{VCD}(C_0) > 0$ and $\mathrm{VCD}(C_1) > 0$, then by Lemma 2 for any concept $c \notin C_0 \times C_1$ (which exists by our assumption), we have that $\mathrm{VCD}((C_0 \times C_1) \cup \{c\}) = \mathrm{VCD}(C_0 \times C_1)$. Hence $C_0 \times C_1$ is not maximal.

Consider the case $\mathrm{VCD}(C_0) = 0$ and $\mathrm{VCD}(C_1) > 0$ (the other case is similar). Let $c_0 \notin C_0$ and $c_1 \in 2^{X_1}$ be such that $\mathrm{VCD}(C_1 \cup \{c_1\}) = \mathrm{VCD}(C_1)$ (e.g., any $c_1 \in C_1$). By Lemma 2, we have that $\mathrm{VCD}((C_0 \times C_1) \cup \{c_0c_1\}) = \mathrm{VCD}(C_0 \times C_1)$. Since $c_0c_1 \notin C_0 \times C_1$, this proves that the class $C_0 \times C_1$ is not maximal. □

On the Learnability of Shuffle Ideals

Dana Angluin[1,⋆], James Aspnes[1,⋆], and Aryeh Kontorovich[2,⋆⋆]

[1] Department of Computer Science
Yale University
New Haven, CT 06520
[2] Department of Computer Science
Ben-Gurion University of the Negev
Beer Sheva, Israel 84105

Abstract. Although PAC learning unrestricted regular languages is long known to be a very difficult problem, one might suppose the existence (and even an abundance) of natural efficiently learnable sub-families. When our literature search for a natural efficiently learnable regular family came up empty, we proposed the shuffle ideals as a prime candidate. A shuffle ideal generated by a string u is simply the collection of all strings containing u as a (discontiguous) subsequence. This fundamental language family is of theoretical interest in its own right and also provides the building blocks for other important language families. Somewhat surprisingly, we discovered that even a class as simple as the shuffle ideals is not properly PAC learnable, unless RP=NP. In the positive direction, we give an efficient algorithm for properly learning shuffle ideals in the statistical query (and therefore also PAC) model under the uniform distribution.

1 Introduction

Inferring regular languages from examples is a classic problem in learning theory. A brief sampling of areas where various automata show up as the underlying formalism include natural language processing (speech recognition, morphological analysis), computational linguistics, robotics and control systems, computational biology (phylogeny, structural pattern recognition), data mining, time series and music [10,23,25–28,35,40]. Thus, developing efficient formal-language learning techniques and understanding their limitations is of a broad and direct relevance in the digital realm.

Perhaps the most widely currently studied notion of learning is Valiant's PAC model [41], which allows for a clean, elegant theory while retaining a decent measure of empirical plausibility. Since PAC learnability is characterized by finite VC-dimension and the concept class of n-state Deterministic Finite-state Automata (DFA) has VC-dimension $\Theta(n \log n)$ [13], the PAC learning problem is solved,

[⋆] Supported in part by NSF grant CCF-0916389.

[⋆⋆] This research was supported by the ISRAEL SCIENCE FOUNDATION (grant No. 1141/12).

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 111–123, 2012.

in an information-theoretic sense, by constructing a DFA on n states consistent with a given labeled sample. Unfortunately, as shown in the works of Angluin [1], Gold [12] and Pitt and Warmuth [34], under standard complexity assumptions, finding small consistent automata is a computationally intractable task. Furthermore, attempts to circumvent the combinatorial search over automata by learning with a different representation class are thwarted by cryptographic hardness results. The papers of Pitt and Warmuth [33] and Kearns and Valiant [16] prove the existence of small automata and "hard" distributions over $\{0,1\}^n$ so that any efficient learning algorithm that achieves a polynomial advantage over random guessing will break various cryptographic hardness assumptions.

In a modified model of PAC, and with additional structural assumptions, a class of probabilistic finite state automata was shown in [8,30] to be learnable; see also the literature review therein. If the target automaton and sampling distribution are assumed to be "simple", efficient *probably exact* learning is possible [31]. When the learner is allowed to make *membership* queries, it follows from [3] that DFAs are learnable in this augmented PAC model.

The prevailing paradigm in formal language learning has been to make structural regularity assumptions about the family of languages and/or the sampling distribution in question and to employ a state-merging heuristic. Indeed, over the years a number of clever and sophisticated combinatorial approaches have been proposed for learning DFAs. Typically, an initial automaton or prefix tree consistent with the sample is first created. Then, starting with the trivial partition with one state per equivalence class, classes are merged while preserving an invariant congruence property. The automaton learned is obtained by merging states according to the resulting classes. Thus, the choice of the congruence determines the algorithm and generalization bounds are obtained from the structural regularity assumptions. This rough summary broadly characterizes the techniques of [2,8,29–31,36] and, until recently, this appears to have been the only general-purpose technique available for learning finite automata.

More recently, Cortes et al. [9,19,20] proposed a substantial departure from the state-merging paradigm. Their approach was to embed a specific family of regular languages (the piecewise-testable ones) in a Hilbert space via a kernel and to identify languages with hyperplanes. A unifying feature of this methodology is that rather than building an automaton, the learning algorithm outputs a classifier defined as a weighted sum of simple automata. In a follow-up work [21], this approach was extended to learning general discrete concepts. These results, however, provided only margin-based generalization guarantees, which are weaker than true PAC bounds.

Perhaps somewhat embarrassingly, there does not appear to be any known natural PAC-learnable family of regular languages. Let us qualify this statement to rule out the obvious objections. Many concept classes are known to be learnable over the boolean cube $\{0,1\}^n$ — conjunctions, disjunctions, decision lists, etc. [17]. Another way to claim trivial results is by importing learning problems from continuous domains. For example, the concept class of axis-aligned rectangles in $\mathbb{R}^2$ is known to be PAC-learnable [17], so certainly these rectangles are also learnable over the rational plane $\mathbb{Q}^2$. Now we may identify $\mathbb{Q}^2$ with $\{0,1\}^*$ via some bijection, thereby identifying rectangles over $\mathbb{Q}^2$ with languages $L \subset \{0,1\}^*$ (we thank Kobbi Nissim for this example). It is a simple matter to construct a bijection $\phi : \mathbb{Q}^2 \to \{0,1\}^*$ that maps rectangles to regular languages and vice versa. Observe, however, that we cannot a priori bound the size of the hypothesis automaton, since higher-precision rectangles will correspond to automata with more states. An even more basic reason to disqualify these examples is that it would be quite a stretch to call them "natural" families of regular languages.

What we mean by a PAC-learnable family of regular languages is, informally, the following. Fix some alphabet Σ. For $n \geq 1$, let $\mathcal{L}_n$ be a collection of regular languages, each of which is recognized by a DFA on n states or fewer. To avoid computational trivialities, let us rule out $|\mathcal{L}_n| = O(\text{poly}(n))$ — this way, brute-force search is infeasible. Since, as we mentioned above and will see in more detail below, the information-theoretic aspects of the learning problem are well-understood, we focus here exclusively on the algorithmic ones. We say that $\mathcal{L} = \bigcup_n \mathcal{L}_n$ is **properly PAC learnable** if there is an algorithm that takes a labeled sample of size m and finds a consistent hypothesis in $\hat{L} \in \mathcal{L}_n$, in time $O(\text{poly}(m,n))$. We say that $\mathcal{L}$ is **improperly PAC learnable** if there is an algorithm that takes a labeled sample of size m and finds a consistent hypothesis with description length $O(\text{poly}(n))$, in time $O(\text{poly}(m,n))$. A formal definition is given in Section 2.

Main results. Our main results concern the PAC-learnability of shuffle ideals. A shuffle ideal generated by a string u is simply the collection of all strings containing u as a (discontiguous) subsequence (see Figure 1 for an illustration). Despite being a particularly simple subfamily of the regular languages, shuffle ideals play a prominent role in formal language theory. Their boolean closure forms the important family known as *piecewise-testable* languages, defined and characterized by Simon in 1975 [39]. The rich structure of this language family has made it an object of intensive study, with deep connections to computability, complexity theory, and semigroups (see [18,24] and the references therein). On a more applied front, the shuffle ideals capture some rudimentary phenomena in human-language morphology [22]. In Section 3 we show that shuffle ideals of known length are exactly [5,7] learnable in the statistical query model under the uniform distribution, though not efficiently. Permitting approximate learning, the algorithm can be made efficient; this in turn yields efficient proper PAC learning under the uniform distribution. On the other hand, in Section 4 we show that the shuffle ideals are not properly PAC-learnable under general distributions unless RP=NP. Whether the shuffle ideals can be improperly PAC learned under general distributions remains an open question.

2 Preliminaries

Notation. Throughout this paper, we consider a fixed finite alphabet Σ, whose size will be denoted by s. We assume $s \geq 2$. The elements of Σ^* will be referred to as *strings* with their length denoted by $|\cdot|$; the empty string is λ. Define the binary relation $\sqsubseteq$ on Σ^* as follows: $u \sqsubseteq v$ holds if there is a witness $\boldsymbol{i} = (i_1 < i_2 < \ldots < i_{|u|})$ such that $v_{i_j} = u_j$ for all $j \in [|u|]$. When there are several witnesses for $u \sqsubseteq v$, we may partially order them coordinate-wise, referring to the unique minimal element as the *leftmost* embedding. We will write $I_{u \sqsubseteq v}$ to denote the position of the last symbol of u in its leftmost embedding in v (if the latter exists; otherwise, $I_{u \sqsubseteq v} = \infty$).

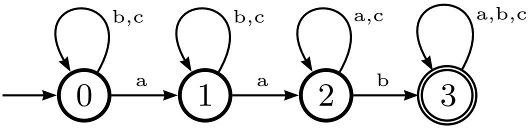


Fig. 1. The canonical DFA for recognizing the shuffle ideal of $u = aab$ over $\Sigma = \{a, b, c\}$, which accepts precisely those strings that contain u as a subsequence

Formally, the (principal) *shuffle ideal* generated by $u \in \Sigma^\ell$ is the regular language

$$Ш(u) = \{x \in \Sigma^* : u \sqsubseteq x\} = \Sigma^* u_1 \Sigma^* u_2 \Sigma^* \ldots \Sigma^* u_\ell \Sigma^*$$

(an example is given in Figure 1). The term *shuffle ideal* comes from algebra [24,32] and dates back to [11].

We use the standard $O(\cdot)$, $o(\cdot)$ notation to denote orders of magnitude. The following simple observation will be useful in the sequel.

Lemma 1. *Evaluating the relation $u \sqsubseteq x$ is feasible in time $O(|x|)$.*

Proof. If $u = \lambda$, then u is certainly a subsequence of x. If $u = au'$ where $a \in \Sigma$, we search for the leftmost occurrence of a in x. If there is no such occurrence, then u is certainly not a subsequence of x. Otherwise, we write $x = yax'$, where y contains no occurrence of a; then u is a subsequence of x if and only if u' is a subsequence of x', so we continue recursively with u' and x'. The total time for this algorithm is $O(|x|)$. □

Learnability. We assume a familiarity with the basics of the PAC learning model [17]. To recap, consider the instance space $\mathcal{X} = \Sigma^*$, concept class $\mathcal{C} \subseteq 2^{\mathcal{X}}$, and hypothesis class $\mathcal{H} \subseteq 2^{\mathcal{X}}$. An *algorithm* $\mathcal{L}$ is given access to a labeled sample $S = (X_i, Y_i)_{i=1}^m$, where the X_i are drawn iid from some unknown distribution P over $\mathcal{X}$ and $Y_i = f(X_i)$ for some unknown *target* $f \in \mathcal{C}$, and produces a *hypothesis* $h \in \mathcal{H}$. We say that $\mathcal{L}$ efficiently PAC-learns $\mathcal{C}$ if for any $\epsilon, \delta > 0$ there is an $m_0 \in \mathbb{N}$ such that for all $f \in \mathcal{C}$ and all distributions P, the hypothesis h_m generated by $\mathcal{L}$ based on a sample of size $m \geq m_0$ satisfies

$$P^m[P(\{x \in \mathcal{X} : h_m(x) \neq f(x)\}) > \epsilon] < \delta;$$

moreover, we require that both m_0 and $\mathcal{L}$'s runtime be at most polynomial in $\epsilon^{-1}, \delta^{-1}$. The learning is said to be *proper* if $\mathcal{H} = \mathcal{C}$ and *improper* otherwise.

Most learning problems can be cleanly decomposed into a computational and an information-theoretic component. The information-theoretic aspects of learning automata are well-understood. As mentioned above, the VC-dimension of a collection of DFAs grows polynomially with maximal number of states, and so any small DFA consistent with the training sample will, with high probability, have small generalization error. For shuffle ideals, an even simpler bound can be derived. If n is an upper bound on the length of the string $u \in \Sigma^*$ generating the target shuffle ideal, then our concept class contains exactly

$$\sum_{\ell=0}^{n} |\Sigma|^{\ell} = O(|\Sigma|^n)$$

members. Thus, with probability at least $1 - \delta$, any shuffle ideal consistent with a sample of size m will achieve a generalization error of

$$O\left(\frac{n \log |\Sigma| - \log \delta}{m}\right). \tag{1}$$

Hence, the problem of properly PAC-learning shuffle ideals has been reduced to finding one that is consistent with a given sample. (This justifies our informal problem statement in the introduction, where the requirements are purely algorithmic and no mention of ϵ, δ is made.) This will turn out to be computationally hard under adversarial distributions (Theorem 4), but feasible under the uniform one (Theorem 3). Actually, our positive result is somewhat stronger: since we show learnability in the statistical query (SQ) model of Kearns [15], this implies a noise-tolerant PAC-result.

3 SQ Learning under the Uniform Distribution

The main result of this section is that shuffle ideals are efficiently PAC-learnable under the uniform distribution. To be more precise, we are dealing with the instance space $\mathcal{X} = \Sigma^n$ endowed with the uniform distribution, which assigns a weight of $|\Sigma|^{-n}$ to each element of $\mathcal{X}$. Our learning algorithm is most naturally expressed in the language of *statistical queries* [15,17]. In the original definition, a statistical query χ is a binary predicate of a random instance-label pair, and the oracle returns the value $\mathbf{E}\chi$, additively perturbed by some amount not exceeding a specified tolerance parameter. We will consider a somewhat richer class of queries.

3.1 Constructing and Analyzing the Queries

For $u \in \Sigma^{\leq n}$ and $a \in \Sigma$, we define the query $\chi_{u,a}(\cdot,\cdot)$ by

$$\chi_{u,a}(x, y) = \begin{cases} 0, & u \not\sqsubseteq x \\ \mathbb{1}_{\{\sigma = a\}} - \mathbb{1}_{\{\sigma \neq a\}}/(s-1), & u \sqsubseteq x,\ y = +1 \\ \mathbb{1}_{\{\sigma \neq a\}}/(s-1) - \mathbb{1}_{\{\sigma = a\}}, & u \sqsubseteq x,\ y = -1 \end{cases}, \tag{2}$$

where σ is the symbol in x following the leftmost embedding of u (formally, $\sigma = x_{I_{x \sqsubseteq u}+1}$) and $\mathbb{1}_{\{\pi\}}$ represents the 0-1 truth value of the predicate π (recall that $s = |\Sigma|$). Our definition of the query $\chi_{u,a}$ is legitimate because (i) it can be efficiently evaluated (Lemma 1) and (ii) it can be expressed as a linear combination of $O(1)$ standard binary queries (also efficiently computable). In words, the function $\chi_{u,a}$ computes the mapping $(x, y) \mapsto \mathbb{R}$ as follows. If u is not a subsequence of x, $\chi_{u,a}(x, y) = 0$. Otherwise, $\chi_{u,a}$ checks whether the symbol σ in x following the leftmost embedding of u is equal to a, and, if x is a positive example ($y = +1$), returns 1 if $\sigma = a$, or $-1/(s-1)$ if $\sigma \neq a$. If x is a negative example ($y = -1$) then the signs of the values returned are inverted.

Suppose for now that the length $L = |\bar{u}|$ of the target shuffle ideal $\bar{u}$ is known. Our learning algorithm uses statistical queries to recover $\bar{u} \in \Sigma^L$ one symbol at a time. It starts with the empty string $u = \lambda$. Having recovered $u = \bar{u}_1, \ldots, \bar{u}_\ell$, $\ell < L$, we infer $\bar{u}_{\ell+1}$ as follows. For each $a \in \Sigma$, the SQ oracle is called with the query $\chi_{u,a}$ and a tolerance $0 < \tau < 1$ to be specified later. Our key technical observation is that the value of $\mathbf{E}\chi_{u,a}$ effectively selects the next symbol of $\bar{u}$:

Lemma 2.

$$\mathbf{E}\chi_{u,a} = \begin{cases} +\frac{2}{s}P(L,n,s), & a = \bar{u}_{\ell+1} \\ -\frac{2}{s(s-1)}P(L,n,s), & a \neq \bar{u}_{\ell+1} \end{cases}$$

where

$$P(L,n,s) = \binom{n-1}{L-1}\left(\frac{1}{s}\right)^{L-1}\left(1-\frac{1}{s}\right)^{n-L}. \tag{3}$$

Proof. Fix an unknown string $\bar{u}$ of length $L \geq 1$; by assumption, we have recovered in $u = u_1 \ldots u_\ell = \bar{u}_1 \ldots \bar{u}_\ell$ the first ℓ symbols of $\bar{u}$. Let $u' = \bar{u}0^\infty$ be the extension of $\bar{u}$ obtained by padding it on the right with infinitely many 0 symbols (we assume $0 \in \Sigma$).

Let X be a random variable representing the uniformly-chosen sample string x. Let T be the largest value for which $u'_1 \ldots u'_T$ is a subsequence of X. Let $\xi = \mathbb{1}_{\{T \geq L\}}$ be the indicator for the event that X is a positive instance, i.e., that $\bar{u}_1 \ldots \bar{u}_L = u'_1 \ldots u'_L$ is a subsequence of X.

Observe that T has a binomial distribution:

$$T \sim \mathrm{Binom}(n, 1/s); \tag{4}$$

indeed, as we sweep across X, each position X_i has a $1/s$ chance of being the next unused symbol of u'. An immediate consequence of this fact is that $\Pr[\xi = 1]$ is exactly $\sum_{k=L}^{n} \binom{n}{k}(1/s)^k(1-1/s)^{n-k}$.

Now fix $\ell < L$. Let $I_\ell = I_{u \sqsubseteq X}$ be the position of u_ℓ in the leftmost embedding of $u_1 \ldots u_\ell$ in X (0 if $\ell = 0$), or $n-1$ if $u_1 \ldots u_\ell$ is not a subsequence of X. Then $I_\ell + 1$ is the position of σ as defined in (2), or n if $u_1 \ldots u_\ell \not\sqsubseteq X_1 \ldots X_{n-1}$.

Define T_ℓ to be the number of symbols of a leftmost embedding of u' in X excluding $X_{I_\ell+1}$:

$$T_\ell = \max\left\{t : u'_1 ... u'_t \sqsubseteq X_1 ... X_{I_\ell} X_{I_\ell+2} ... X_n\right\}.$$

Like T, T_ℓ also has a binomial distribution, but now

$$T_\ell \sim \mathrm{Binom}(n-1, 1/s). \tag{5}$$

The reason is that we always omit one position in X (the one following u_ℓ if u_ℓ appears before X_n or X_n if it does not), and for each other position, there is still an independent $1/s$ chance that it is the next symbol in u'.

An important fact is that T_ℓ is independent of $X_{I_\ell+1}$. This is not immediately obvious: whether $X_{I_\ell+1}$ equals $u'_{\ell+1}$ or not affects the interpretation of later symbols in X. However, the probability that each symbol $X_{I_\ell+2}\dots$ is the next unused symbol in u' is still an independent $1/s$ whether $X_{I_\ell+1}$ consumes a symbol of u' or not. So the distribution of T_ℓ is not affected.

We now compute $\mathbf{E}\chi_{u,a}$ by averaging over all choices of T_ℓ. If $T_\ell < \ell$, then $u_1 \dots u_\ell \not\sqsubseteq X_1 \dots X_{n-1}$ and $\chi_{u,a} = 0$. If $\ell \le T_\ell \le L-2$, then X is a negative example. Each symbol in Σ contributes 1 to the mean with probability $1/s$ and $-\frac{1}{s-1}$ with probability $\frac{s-1}{s}$; the net contribution is 0. Similarly, if $T_\ell \ge L$, X is a positive example, and the probability-$(1/s)$ gain of 1 is exactly offset by the probability-$\left(\frac{s-1}{s}\right)$ loss of $\frac{1}{s-1}$.

This leaves the case $T_\ell = L-1$. Now X is positive if and only if $X_{I_\ell+1} = \bar{u}_{\ell+1}$, which occurs if $\sigma = \bar{u}_{\ell+1}$. So the conditional expectation is $1 \cdot \Pr[\sigma = \bar{u}_{\ell+1}] + \frac{1}{s-1} \cdot \Pr[\sigma \neq \bar{u}_{\ell+1}] = \frac{1}{s} + \frac{1}{s-1} \cdot \frac{s-1}{s} = 2/s$. For $a \neq \bar{u}_{\ell+1}$, the conditional expectation is is $-\frac{2}{s(s-1)}$. This can be computed directly by considering cases, or by observing that the change to $\sum_{a\in\Sigma} \chi_{u,a}(x) = 0$ always, and that all $a \neq \bar{u}_{\ell+1}$ induce same expectation by symmetry.

Since the only case that produces a nonzero conditional expectation is $T_\ell = L-1$, we have

$$\mathbf{E}\chi_{u,\bar{u}_{\ell+1}} = +\frac{2}{s}\Pr[T_\ell = L-1], \tag{6}$$

and for each $a \neq \bar{u}_{\ell+1}$,

$$\mathbf{E}\chi_{u,a} = -\frac{2}{s(s-1)}\Pr[T_\ell = L-1]. \tag{7}$$

The claim follows since $T_\ell \sim \mathrm{Binom}(n-1, 1/s)$ by (5). □

3.2 Specifying the Query Tolerance τ

The analysis in Lemma 2 suggests that inferring $\bar{u} \in \Sigma^L$ amounts to distinguishing the two possible values of $\mathbf{E}\chi_{u,a}$. If we set the query tolerance to half the larger value

$$\tau = \frac{1}{s}\Pr[T_\ell = L-1] \tag{8}$$

then s statistical queries for each prefix of $\bar{u}$ suffices to learn $\bar{u}$ exactly.

Theorem 1. *When the length L of the target string $\bar{u}$ is known, $\bar{u}$ is exactly identifiable with $O(Ls)$ statistical queries at tolerance $\tau = \frac{1}{s}P(L, n, s)$.*

In the above SQ algorithm there is no need for a precision parameter ϵ because the learning is *exact*, that is, $\epsilon = 0$. Nor is there a need for a confidence parameter δ because each statistical query is guaranteed to return an answer within the specified tolerance, in contrast to the PAC setting where the parameter δ protects the learner against an "unlucky" sample.

However, if the relationship between n and L is such that $P(L, n, s)$ is very small, then the tolerance τ will be very small, and this first SQ algorithm cannot be considered efficient. If we allow an approximately correct hypothesis ($\epsilon > 0$), we can modify the above algorithm to use a polynomially bounded tolerance.

Theorem 2. *When the length L of the target string $\bar{u}$ is known, $\bar{u}$ is approximately identifiable to within $\epsilon > 0$ with $O(Ls)$ statistical queries at tolerance $\tau = \epsilon/(3sn)$.*

Proof. We modify the SQ algorithm to make an initial statistical query with tolerance $\epsilon/3$ to estimate $\Pr[\xi = 1]$, the probability that x is a positive example. If the answer is $\leq 2\epsilon/3$, then $\Pr[\xi = 1] \leq \epsilon$ and the algorithm outputs a hypothesis that classifies all examples as negative. If the answer is $\geq 1 - 2\epsilon/3$, then $\Pr[\xi = 1] \geq 1 - \epsilon$ and the algorithm outputs a hypothesis that classifies all examples as positive.

Otherwise, $\Pr[\xi = 1]$ is between $\epsilon/3$ and $1 - \epsilon/3$, and the first SQ algorithm is used. We now show that $P(L, n, s) \geq \epsilon/(3n)$, establishing the bound on the tolerance. Let $Q(L, n, s) = \binom{n}{L}\left(\frac{1}{s}\right)^L\left(1 - \frac{1}{s}\right)^{n-L}$ and note that $Q(L, n, s) = (n/Ls)P(L, n, s)$. If $L \leq n/s$ then $Q(L, n, s)$ is at least as large as every term in the sum

$$\Pr[\xi = 0] = \sum_{k=0}^{L-1}\binom{n}{k}\left(\frac{1}{s}\right)^k\left(1 - \frac{1}{s}\right)^{n-k}$$

and therefore $Q(L, n, s) \geq \epsilon/(3L)$ and $P(L, n, s) \geq \epsilon/(3n)$. If $L > n/s$ then $Q(L, n, s)$ is at least as large as every term in the sum

$$\Pr[\xi = 1] = \sum_{k=L}^{n}\binom{n}{k}\left(\frac{1}{s}\right)^k\left(1 - \frac{1}{s}\right)^{n-k}$$

and therefore $P(L, n, s) \geq Q(L, n, s) \geq \epsilon/(3n)$. □

3.3 PAC Learning

The main result of this section is now obtained by a standard transformation of an SQ algorithm to a PAC algorithm.

Theorem 3. *The concept class $\mathcal{C} = \left\{Ш(u) : u \in \Sigma^{\leq n}\right\}$ is efficiently properly PAC learnable under the uniform distribution.*

Proof. We assume that the algorithm receives as inputs n, L, ϵ and δ. Because there are only $n + 1$ choices of L, a standard method may be used to iterate through them. We simulate the modified SQ algorithm by drawing a sample of labeled examples and using them to estimate the answers to the $O(Ls)$ calls to the SQ oracle with queries at tolerance $\tau = \epsilon/(3sn)$, as described in [15]. According to [15, Theorem 1],

$$O\left(\frac{1}{\tau^2}\log\frac{|\mathcal{C}|}{\delta}\right) = O\left(\frac{s^2n^2}{\epsilon^2}(n\log s - \log\delta)\right)$$

examples suffice to determine correct answers to all the queries at the desired tolerance, with probability at least $1 - \delta$.

□

Remark 1. Our learning algorithm and analysis are rather strongly tied to the uniform distribution. If this assumption is omitted, it might now happen that $\Pr[T_\ell = m - 1]$ is small even though positive and negative examples are mostly balanced, or there might be intractable correlations between σ and T_ℓ. It seems that genuinely new ideas will be required to handle nonuniform distributions.

4 Proper PAC Learning under General Distributions Is Hard Unless NP=RP

Our hardness result will follow the standard paradigm, exemplified in [17]. We will show that the problem of deciding whether a given labeled sample admits a consistent shuffle ideal is NP-complete. A standard argument then shows that any proper PAC learner for shuffle ideals can be efficiently manipulated into solving the decision problem, yielding an algorithm in RP. Thus, assuming RP≠NP, there is no polynomial-time algorithm that properly learns shuffle ideals.

Theorem 4. *Given two disjoint sets of strings $S, T \subset \Sigma^*$, the problem of determining whether there exists a string w such that $w \sqsubseteq x$ for each $x \in S$ and $w \not\sqsubseteq x$ for each $x \in T$ is NP-complete.*

Proof. To see that this problem is in NP, note that if S is empty, then any string of length longer than the longest string in T satisfies the necessary requirements, so that the answer in this case is necessarily "yes." If S is nonempty, then no string longer than the shortest string in S can be a subsequence of every string in S, so we need only guess a string w whose length is bounded by that of the shortest string in S and check whether w is a subsequence of every string in S and of no string in T, which takes time proportional to the sum of the lengths of all the input strings (Lemma 1).

To see that this problem is complete in NP, we reduce satisfiability of 3-CNF formulas to this question. Given a formula ϕ containing n clauses C_i, where each clause contains three literals $\ell_{i,1}$, $\ell_{i,2}$ and $\ell_{i,3}$, the question of whether ϕ is satisfiable is equivalent to the question of whether we can select exactly one literal from each clause in such a way that no two selected literals are complements of each other.

The heart of the construction is a three-way choice of one part of a subsequence for each clause of the formula. Consider the strings $x_1 = aba$ and $x_2 = baab$. The strings that are subsequences of both of these strings are precisely

$$\{\lambda, a, b, aa, ab, ba\}.$$

Thus if we were to specify positive strings x_1 and x_2, and negative strings a and b, there are exactly three strings that are subsequences of both the positive strings and not subsequences of either of the negative strings, namely, $\{aa, ab, ba\}$. We use n copies of this three-way choice to represent the choice of one literal from each of the n clauses.

We define S to contain the two positive strings:

$$u_1 = (x_1 d)^n$$
$$u_2 = (x_2 d)^n$$

where the symbol d acts to delimit the region of each string corresponding to each clause.

Our first group of negative strings, T_1, contains the n strings obtained from u_1 by deleting one occurrence of d. A string w that is a subsequence of u_1 and not a subsequence of any string in T_1 must have exactly n occurrences of d. The occurrences of d divide w into regions corresponding to the successive occurrences of x_1 in u_1 and x_2 in u_2.

Our second group of negative strings, T_2, contains the $2n$ strings obtained from u_1 by selecting a region i and replacing the x_1 in that region of u_1 by a or b. We precisely characterize the set of strings w that are subsequences of u_1 and u_2 but not of any string in T_1 or T_2 as the strings described by the regular expression $((y_1 + y_2 + y_3)d)^n$, where $y_1 = aa$, $y_2 = ab$, and $y_3 = ba$. In region i we associate the choice of string y_r with choosing the literal $\ell_{i,r}$.

Finally, our third group of negative strings, T_3, contains a string for each pair of complementary literals (say $\ell_{i,r}$ and $\ell_{j,s}$) obtained from u_1 as follows. In region i we substitute y_r for x_1, and in region j we substitute y_s for x_1. This negative string means that a consistent string w cannot make a choice of strings corresponding to the complementary literals $\ell_{i,r}$ and $\ell_{j,s}$.

Then there is a string w that is a subsequence of every string in the positive set S and of no string in the negative set $T = T_1 \cup T_2 \cup T_3$ if and only if the original formula ϕ is satisfiable, concluding the NP-completeness proof. □

5 The Difficulty of Learning Unions of Shuffle Ideals

In this section we note that the problem of learning a monotone DNF formula is efficiently reducible to the problem of learning a union of shuffle ideals. Let ϕ be a monotone DNF formula over variables $\{x_i\}$ for $i = 1, \ldots, n$. We consider an ordered alphabet $\{x_1, \ldots, x_n\}$ and a union h of shuffle ideals obtained from ϕ as follows. Each term, e.g., $x_6 x_{14} x_{22}$, of ϕ is mapped to a shuffle ideal consisting of the symbols (in order) corresponding to the variables in the term, e.g., $Ш(x_6 x_{14} x_{22})$. Then h is the union of the shuffle ideals obtained in this way.

If we map each assignment to the variables $\{x_i\}$ to the substring of

$$x_1 x_2 \cdots x_n$$

obtained by omitting the symbols corresponding to variables assigned 0, then the assignments satisfying ϕ are precisely the substrings of $x_1 x_2 \cdots x_n$ that are in the union of shuffle ideals h. Thus an efficient method of learning unions of shuffle ideals would yield an efficient method of learning for monotone DNF formulas, which so far is only known in special cases [4,6,14,37,38].

6 Discussion

We have shown that even a family of regular languages as simple as the shuffle ideals is not efficiently properly PAC learnable if RP≠NP. Thus, the search for a nontrivial (in the sense described in the introduction) properly PAC-learnable family of languages continues. On the other hand, even with classification noise, efficient proper PAC learning of shuffle ideals is possible under the uniform distribution. The major unresolved question is whether it is possible to *improperly* learn shuffle ideals under general distributions; this is the subject of ongoing research. Also open is the question of whether the alphabet size in Theorem 4 can be reduced to 2.

Acknowledgments. We thank Sarah Eisenstat for the construction reducing the alphabet size to 3 in the proof of Theorem 4 and the anonymous referees for their helpful comments.

References

[1] Angluin, D.: On the complexity of minimum inference of regular sets. Information and Control 3(39), 337–350 (1978)

[2] Angluin, D.: Inference of reversible languages. Journal of the ACM (JACM) 3(29), 741–765 (1982)

[3] Angluin, D.: Learning regular sets from queries and counterexamples. Inf. Comput. 75(2), 87–106 (1987)

[4] Angluin, D., Slonim, D.K.: Randomly fallible teachers: Learning monotone DNF with an incomplete membership oracle. Machine Learning 14(1), 7–26 (1994)

[5] Bshouty, N.H.: Exact learning of formulas in parallel. Machine Learning 26(1), 25–41 (1997)

[6] Bshouty, N.H., Eiron, N.: Learning monotone DNF from a teacher that almost does not answer membership queries. Journal of Machine Learning Research 3, 49–57 (2002)

[7] Bshouty, N.H., Jackson, J.C., Tamon, C.: Exploring learnability between exact and PAC. J. Comput. Syst. Sci. 70(4), 471–484 (2005)

[8] Clark, A., Thollard, F.: Pac-learnability of probabilistic deterministic finite state automata. Journal of Machine Learning Research (JMLR) 5, 473–497 (2004)

[9] Cortes, C., Kontorovich, L.(A.), Mohri, M.: Learning Languages with Rational Kernels. In: Bshouty, N.H., Gentile, C. (eds.) COLT. LNCS (LNAI), vol. 4539, pp. 349–364. Springer, Heidelberg (2007)

[10] de la Higuera, C.: A bibliographical study of grammatical inference. Pattern Recognition 38, 1332–1348 (2005)

[11] Eilenberg, S., Mac Lane, S.: On the groups of $H(\Pi, n)$. I. Ann. of Math. (2) 58, 55–106 (1953)

[12] Mark Gold, E.: Complexity of automaton identification from given data. Information and Control 3(37), 302–420 (1978)

[13] Ishigami, Y., Tani, S.: Vc-dimensions of finite automata and commutative finite automata with k letters and n states. Discrete Applied Mathematics 74(2), 123–134 (1997)

[14] Jackson, J.C., Lee, H.K., Servedio, R.A., Wan, A.: Learning random monotone DNF. Discrete Applied Mathematics 159(5), 259–271 (2011)

[15] Kearns, M.: Efficient noise-tolerant learning from statistical queries. J. ACM 45(6), 983–1006 (1998)

[16] Kearns, M.J., Valiant, L.G.: Cryptographic limitations on learning boolean formulae and finite automata. Journal of the ACM (JACM) 41(1), 67–95 (1994)

[17] Kearns, M., Vazirani, U.: An Introduction to Computational Learning Theory. The MIT Press (1997)

[18] Klíma, O., Polák, L.: Hierarchies of piecewise testable languages. Int. J. Found. Comput. Sci. 21(4), 517–533 (2010)

[19] Kontorovich, L.(A.), Cortes, C., Mohri, M.: Learning Linearly Separable Languages. In: Balcázar, J.L., Long, P.M., Stephan, F. (eds.) ALT 2006. LNCS (LNAI), vol. 4264, pp. 288–303. Springer, Heidelberg (2006)

[20] Kontorovich, L.(A.), Cortes, C., Mohri, M.: Kernel methods for learning languages. Theor. Comput. Sci. 405(3), 223–236 (2008)

[21] Kontorovich, L.(A.), Nadler, B.: Universal Kernel-Based Learning with Applications to Regular Languages. Journal of Machine Learning Research 10, 997–1031 (2009)

[22] Kontorovich, L.(A.), Ron, D., Singer, Y.: A Markov Model for the Acquisition of Morphological Structure. Technical Report CMU-CS-03-147 (2003)

[23] Koskenniemi, K.: Two-level model for morphological analysis. In: IJCAI, pp. 683–685 (1983)

[24] Lothaire, M.: Combinatorics on Words. Encyclopedia of Mathematics and Its Applications, vol. 17. Addison-Wesley (1983)

[25] Mohri, M.: On some applications of finite-state automata theory to natural language processing. Nat. Lang. Eng. 2, 61–80 (1996)

[26] Mohri, M.: Finite-state transducers in language and speech processing. Computational Linguistics 23(2), 269–311 (1997)

[27] Mohri, M., Moreno, P., Weinstein, E.: Efficient and robust music identification with weighted finite-state transducers. IEEE Transactions on Audio, Speech & Language Processing 18(1), 197–207 (2010)

[28] Mohri, M., Pereira, F., Riley, M.: Weighted finite-state transducers in speech recognition. Computer Speech & Language 16(1), 69–88 (2002)

[29] Oncina, J., García, P.: Identifying regular languages in polynomial time. In: Advances in Structural and Syntactic Pattern Recognition, pp. 49–61. World Scientific Publishing (1992)

[30] Palmer, N., Goldberg, P.W.: PAC-learnability of probabilistic deterministic finite state automata in terms of variation distance. Theor. Comput. Sci. 387(1), 18–31 (2007)

[31] Parekh, R., Honavar, V.G.: Learning DFA from simple examples. Mach. Learn. 44(1-2), 9–35 (2001)

[32] Păun, G.: Mathematical Aspects of Natural and Formal Languages. World Scientific Publishing (1994)

[33] Pitt, L., Warmuth, M.: Prediction-preserving reducibility. Journal of Computer and System Sciences 41(3), 430–467 (1990)

[34] Pitt, L., Warmuth, M.: The minimum consistent DFA problem cannot be approximated within any polynomial. Journal of the Association for Computing Machinery 40(1), 95–142 (1993)

[35] Rambow, O., Bangalore, S., Butt, T., Nasr, A., Sproat, R.: Creating a finite-state parser with application semantics. In: COLING (2002)

[36] Ron, D., Singer, Y., Tishby, N.: On the learnability and usage of acyclic probabilistic finite automata. Journal of Computer and System Sciences 56(2), 133–152 (1998)

[37] Sellie, L.: Learning random monotone DNF under the uniform distribution. In: COLT, pp. 181–192 (2008)

[38] Servedio, R.A.: On learning monotone DNF under product distributions. Inf. Comput. 193(1), 57–74 (2004)

[39] Simon, I.: Piecewise Testable Events. In: Brakhage, H. (ed.) GI-Fachtagung 1975. LNCS, vol. 33, pp. 214–222. Springer, Heidelberg (1975)

[40] Sproat, R., Shih, C., Gale, W., Chang, N.: A stochastic finite-state word-segmentation algorithm for Chinese. Computational Linguistics 22(3), 377–404 (1996)

[41] Valiant, L.G.: A theory of the learnable. Commun. ACM 27(11), 1134–1142 (1984)

New Analysis and Algorithm for Learning with Drifting Distributions

Mehryar Mohri[1,2] and Andres Muñoz Medina[1]

[1] Courant Institute of Mathematical Sciences, New York, NY
[2] Google Research, New York, NY

Abstract. We present a new analysis of the problem of learning with drifting distributions in the batch setting using the notion of discrepancy. We prove learning bounds based on the Rademacher complexity of the hypothesis set and the discrepancy of distributions both for a drifting PAC scenario and a tracking scenario. Our bounds are always tighter and in some cases substantially improve upon previous ones based on the L_1 distance. We also present a generalization of the standard on-line to batch conversion to the drifting scenario in terms of the discrepancy and arbitrary convex combinations of hypotheses. We introduce a new algorithm exploiting these learning guarantees, which we show can be formulated as a simple QP. Finally, we report the results of preliminary experiments demonstrating the benefits of this algorithm.

Keywords: Drifting environment, generalization bound, domain adaptation.

1 Introduction

In the standard PAC model [1] and other similar theoretical models of learning [2], the distribution according to which training and test points are drawn is fixed over time. However, for many tasks such as spam detection, political sentiment analysis, financial market prediction under mildly fluctuating economic conditions, or news stories, the learning environment is not stationary and there is a continuous drift of its parameters over time.

There is a large body of literature devoted to the study of related problems both in the on-line and the batch learning scenarios. In the on-line scenario, the target function is typically assumed to be fixed but no distributional assumption is made, thus input points may be chosen adversarially [3]. Variants of this model where the target is allowed to change a fixed number of times have also been studied [3, 4, 5, 6]. In the batch scenario, the case of a fixed input distribution with a drifting target was originally studied by Helmbold and Long [7]. A more general scenario was introduced by Bartlett [8] where the joint distribution over the input and labels could drift over time under the assumption that the L_1 distance between the distributions in two consecutive time steps was bounded. Both generalization bounds and lower bounds have been given for this scenario [9, 10]. In particular, Long [9] showed that if the L_1 distance between two consecutive distributions is at most Δ, then a generalization error of $O((d\Delta)^{1/3})$ is achievable and Barve and Long [10] proved this bound to be tight. Further improvements were presented by Freund and Mansour [11] under the assumption of a constant

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 124–138, 2012.

rate of change for drifting. Other settings allowing arbitrary but infrequent changes of the target have also been studied [12]. An intermediate model of drift based on a `near` relationship was also recently introduced and analyzed by [13] where consecutive distributions may change arbitrarily, modulo the restriction that the region of disagreement between nearby functions would only be assigned limited distribution mass at any time.

This paper deals with the analysis of learning in the presence of drifting distributions in the batch setting. We consider both the general drift model introduced by [8] and a related drifting PAC model that we will later describe. We present new generalization bounds for both models (Sections 3 and 4). Unlike the L_1 distance used by previous authors to measure the distance between distributions, our bounds are based on a notion of *discrepancy* between distributions generalizing the definition originally introduced by [14] in the context of domain adaptation. The L_1 distance used in previous analyses admits several drawbacks: in general, it can be very large, even in favorable learning scenarios; it ignores the loss function and the hypothesis set used; and it cannot be accurately and efficiently estimated from finite samples (see for example lower bounds on the sample complexity of testing closeness by [15]). In contrast, the discrepancy takes into consideration both the loss function and the hypothesis set.

The learning bounds we present in Sections 3 and 4 are tighter than previous bounds both because they are given in terms of the discrepancy which lower bounds the L_1 distance, and because they are given in terms of the Rademacher complexity instead of the VC-dimension. Additionally, our proofs are often simpler and more concise. We also present a generalization of the standard on-line to batch conversion to the scenario of drifting distributions in terms of the discrepancy measure (Section 5). Our guarantees hold for convex combinations of the hypotheses generated by an on-line learning algorithm. These bounds lead to the definition of a natural meta-algorithm which consists of selecting the convex combination of weights in order to minimize the discrepancy-based learning bound (Section 6). We show that this optimization problem can be formulated as a simple QP and report the results of preliminary experiments demonstrating its benefits. Finally we will discuss the practicality of our algorithm in some natural scenarios.

2 Preliminaries

In this section, we introduce some preliminary notation and key definitions, including that of the *discrepancy* between distributions, and describe the learning scenarios we consider.

Let $\mathcal{X}$ denote the input space and $\mathcal{Y}$ the output space. We consider a loss function $L\colon \mathcal{Y}\times\mathcal{Y} \to \mathbb{R}_+$ bounded by some constant $M > 0$. For any two functions $h, h'\colon \mathcal{X} \to \mathcal{Y}$ and any distribution D over $\mathcal{X} \times \mathcal{Y}$, we denote by $\mathcal{L}_D(h)$ the expected loss of h and by $\mathcal{L}_D(h, h')$ the expected loss of h with respect to h':

$$\mathcal{L}_D(h) = \operatorname*{E}_{(x,y)\sim D}[L(h(x), y)] \qquad \text{and} \qquad \mathcal{L}_D(h, h') = \operatorname*{E}_{x\sim D^1}[L(h(x), h'(x))], \quad (1)$$

where D^1 is the marginal distribution over $\mathcal{X}$ derived from D. We adopt the standard definition of the empirical Rademacher complexity, but we will need the following sequential definition of a Rademacher complexity, which is related to that of [16].

Definition 1. *Let G be a family of functions mapping from a set $\mathcal{Z}$ to $\mathbb{R}$ and $S = (z_1, \ldots, z_T)$ a fixed sample of size T with elements in $\mathcal{Z}$. The* empirical Rademacher complexity *of G for the sample S is defined by:*

$$\widehat{\mathfrak{R}}_S(G) = \mathop{\mathrm{E}}_{\boldsymbol{\sigma}}\left[\sup_{g \in G} \frac{1}{T} \sum_{t=1}^{T} \sigma_t g(z_t)\right], \tag{2}$$

where $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_T)^\top$, with σ_ts independent uniform random variables taking values in $\{-1, +1\}$. The Rademacher complexity *of G is the expectation of $\widehat{\mathfrak{R}}_S(G)$ over all samples $S = (z_1, \ldots, z_T)$ of size T drawn according to the product distribution $D = \bigotimes_{t=1}^T D_t$:*

$$\mathfrak{R}_T(G) = \mathop{\mathrm{E}}_{S \sim D}[\widehat{\mathfrak{R}}_S(G)]. \tag{3}$$

Note that this coincides with the standard Rademacher complexity when the distributions D_t, $t \in [1, T]$, all coincide.

A key question for the analysis of learning with a drifting scenario is a measure of the difference between two distributions D and D'. The distance used by previous authors is the L_1 distance. However, the L_1 distance is not helpful in this context since it can be large even in some rather favorable situations. Moreover, the L_1 distance cannot be accurately and efficiently estimated from finite samples and it ignores the loss function used. Thus, we will adopt instead the *discrepancy*, which provides a measure of the dissimilarity of two distributions that takes into consideration both the loss function and the hypothesis set used, and that is suitable to the specific scenario of drifting.

Our definition of discrepancy is a generalization to the drifting context of the one introduced by [14] for the analysis of domain adaptation. Observe that for a fixed hypothesis $h \in H$, the quantity of interest with drifting distributions is the difference of the expected losses $\mathcal{L}_{D'}(h) - \mathcal{L}_D(h)$ for two consecutive distributions D and D'. A natural distance between distributions in this context is thus one based on the supremum of this quantity over all $h \in H$.

Definition 2. *Given a hypothesis set H and a loss function L, the* $\mathcal{Y}$-discrepancy $\mathrm{disc}_{\mathcal{Y}}$ *between two distributions D and D' over $\mathcal{X} \times \mathcal{Y}$ is defined by:*

$$\mathrm{disc}_{\mathcal{Y}}(D, D') = \sup_{h \in H} \left|\mathcal{L}_{D'}(h) - \mathcal{L}_D(h)\right|. \tag{4}$$

In a deterministic learning scenario with a labeling function f, the previous definition becomes

$$\mathrm{disc}_{\mathcal{Y}}(D, D') = \sup_{h \in H} \left|\mathcal{L}_{D'^1}(f, h) - \mathcal{L}_{D^1}(f, h)\right|, \tag{5}$$

where D'^1 and D^1 are the marginal distributions associated to D and D' defined over $\mathcal{X}$. The target function f is unknown and could match any hypothesis h'. This leads to the following definition [14].

Definition 3. *Given a hypothesis set H and a loss function L, the* discrepancy disc *between two distributions D and D' over $\mathcal{X} \times \mathcal{Y}$ is defined by:*

$$\mathrm{disc}(D, D') = \sup_{h, h' \in H} \left|\mathcal{L}_{D'^1}(h', h) - \mathcal{L}_{D^1}(h', h)\right|. \tag{6}$$

An important advantage of this last definition of discrepancy, in addition to those already mentioned, is that it can be accurately estimated from finite samples drawn from D'^1 and D^1 when the loss is bounded and the Rademacher complexity of the family of functions $L_H = \{x \mapsto L(h'(x), h(x)) \colon h, h' \in H\}$ is in $O(1/\sqrt{T})$, where T is the sample size; in particular when L_H has a finite pseudo-dimension [14]. The discrepancy is by definition symmetric and verifies the triangle inequality for any loss function L. In general, it does not define a *distance* since we may have $\mathrm{disc}(D, D') = 0$ for $D' \neq D$. However, in some cases, for example for kernel-based hypothesis sets based on a Gaussian kernel, the discrepancy has been shown to be a distance [17].

We will present our learning guarantees in terms of the $\mathcal{Y}$-discrepancy $\mathrm{disc}_{\mathcal{Y}}$, that is the most general definition since guarantees in terms of the discrepancy disc can be straightforwardly derived from them. The advantage of the latter bounds is the fact that the discrepancy can be estimated in that case from unlabeled finite samples.

We will consider two different scenarios for the analysis of learning with drifting distributions: the *drifting PAC scenario* and the *drifting tracking scenario*.

The drifting PAC scenario is a natural extension of the PAC scenario, where the objective is to select a hypothesis h out of a hypothesis set H with a small expected loss according to the distribution D_{T+1} after receiving a sample of $T \geq 1$ instances drawn from the product distribution $\bigotimes_{t=1}^{T} D_t$. Thus, the focus in this scenario is the performance of the hypothesis h with respect to the environment distribution after receiving the training sample.

The drifting tracking scenario we consider is based on the scenario originally introduced by [8] for the zero-one loss and is used to measure the performance of an algorithm $\mathcal{A}$ (as opposed to any hypothesis h). In that learning model, the performance of an algorithm is determined based on its average predictions at each time for a sequence of distributions. We will generalize its definition by using the notion of discrepancy and extending it to other loss functions. The following definitions are the key concepts defining this model.

Definition 4. *For any sample $S = (x_t, y_t)_{t=1}^{T}$ of size T, we denote by $h_{T-1} \in H$ the hypothesis returned by an algorithm $\mathcal{A}$ after receiving the first $T-1$ examples and by $\widehat{M}_T$ its loss or mistake on x_T: $\widehat{M}_T = L(h_{T-1}(x_T), y_T)$. For a product distribution $D = \bigotimes_{t=1}^{T} D_t$ on $(\mathcal{X} \times \mathcal{Y})^T$ we denote by $M_T(D)$ the expected mistake of $\mathcal{A}$:*

$$M_T(D) = \mathop{\mathrm{E}}_{S \sim D}[\widehat{M}_T] = \mathop{\mathrm{E}}_{S \sim D}[L(h_{T-1}(x_T), y_T)].$$

Definition 5. *Let $\Delta > 0$ and let $\widetilde{M}_T$ be the supremum of $M_T(D)$ over all distribution sequences $D = (D_t)$, with $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{t+1}) < \Delta$. Algorithm $\mathcal{A}$ is said to (Δ, ϵ)-track H if there exists t_0 such that for $T > t_0$ we have $\widetilde{M}_T < \inf_{h \in H} \mathcal{L}_{D_T}(h) + \epsilon$.*

An analysis of the tracking scenario with the L_1 distance used to measure the divergence of distributions instead of the discrepancy was carried out by Long [9] and Barve and Long [10], including both upper and lower bounds for $\widetilde{M}_T$ in terms of Δ. Their analysis makes use of an algorithm very similar to empirical risk minimization, which we will also use in our theoretical analysis of both scenarios.

3 Drifting PAC Scenario

In this section, we present guarantees for the drifting PAC scenario in terms of the discrepancies of D_t and D_{T+1} , $t \in [1, T]$, and the Rademacher complexity of the hypothesis set. We start with a generalization bound in this scenario and then present a bound for the agnostic learning setting.

Let us emphasize that learning bounds in the drifting scenario should of course not be expected to converge to zero as a function of the sample size but depend instead on the divergence between distributions.

Theorem 1. *Assume that the loss function L is bounded by M. Let $D_1, \ldots, D_{T+1}$ be a sequence of distributions and let $H_L = \{(x, y) \mapsto L(h(x), y) \colon h \in H\}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $h \in H$:*

$$\mathcal{L}_{D_{T+1}}(h) \leq \frac{1}{T}\sum_{t=1}^{T} L(h(x_t), y_t) + 2\mathfrak{R}_T(H_L) + \frac{1}{T}\sum_{t=1}^{T} \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + M\sqrt{\frac{\log\frac{1}{\delta}}{2T}}.$$

Proof. We denote by D the product distribution $\bigotimes_{t=1}^{T} D_t$. Let Φ be the function defined over any sample $S = ((x_1, y_1), \ldots, (x_T, y_T)) \in (\mathcal{X} \times \mathcal{Y})^T$ by

$$\Phi(S) = \sup_{h \in H} \mathcal{L}_{D_{T+1}}(h) - \frac{1}{T}\sum_{t=1}^{T} L(h(x_t), y_t).$$

Let S and S' be two samples differing by one labeled point, say (x_t, y_t) in S and (x'_t, y'_t) in S', then:

$$\Phi(S') - \Phi(S) \leq \sup_{h \in H} \frac{1}{T}\Big[L(h(x'_t), y'_t) - L(h(x_t), y_t)\Big] \leq \frac{M}{T}.$$

Thus, by McDiarmid's inequality, the following holds:[1]

$$\Pr_{S \sim D}\Big[\Phi(S) - \mathop{\mathrm{E}}_{S \sim D}[\Phi(S)] > \epsilon\Big] \leq \exp(-2T\epsilon^2/M^2).$$

We now bound $\mathrm{E}_{S \sim D}[\Phi(S)]$ by first rewriting it, as follows:

$$\begin{aligned}
&\mathrm{E}\Big[\sup_{h \in H} \mathcal{L}_{D_{T+1}}(h) - \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{D_t}(h) + \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{D_t}(h) - \frac{1}{T}\sum_{t=1}^{T} L(h(x_t), y_t)\Big] \\
&\leq \mathrm{E}\Big[\sup_{h \in H} \mathcal{L}_{D_{T+1}}(h) - \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{D_t}(h)\Big] + \mathrm{E}\Big[\sup_{h \in H} \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{D_t}(h) - \frac{1}{T}\sum_{t=1}^{T} L(h(x_t), y_t)\Big] \\
&\leq \mathrm{E}\Big[\frac{1}{T}\sum_{t=1}^{T} \sup_{h \in H} \big(\mathcal{L}_{D_{T+1}}(h) - \mathcal{L}_{D_t}(h)\big) + \sup_{h \in H} \frac{1}{T}\sum_{t=1}^{T} \big(\mathcal{L}_{D_t}(h) - L(h(x_t), y_t)\big)\Big] \\
&\leq \frac{1}{T}\sum_{t=1}^{T} \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + \mathrm{E}\Big[\sup_{h \in H} \frac{1}{T}\sum_{t=1}^{T} \big(\mathcal{L}_{D_t}(h) - L(h(x_t), y_t)\big)\Big].
\end{aligned}$$

[1] Note that McDiarmid's inequality does not require points to be drawn according to the same distribution but only that they would be drawn independently.

It is not hard to see, using a symmetrization argument as in the non-sequential case, that the second term can be bounded by $2\mathfrak{R}_T(H_L)$. □

For many commonly used loss functions, the empirical Rademacher complexity $\mathfrak{R}_T(H_L)$ can be upper bounded in terms of that of the function class H. In particular, for the zero-one loss it is known that $\mathfrak{R}_T(H_L) = \mathfrak{R}_T(H)/2$ and when L is the L_q loss for some $q \geq 1$, that is $L(y, y') = |y' - y|^q$ for all $y, y' \in \mathcal{Y}$, then $\mathfrak{R}_T(H_L) \leq qM^{q-1}\mathfrak{R}_T(H)$. Indeed, since $x \mapsto |x|^q$ is qM^{q-1}-Lipschitz over $[-M, +M]$, by Talagrand's contraction lemma, $\mathfrak{R}_T(H_L)$ is bounded by $qM^{q-1}\widehat{\mathfrak{R}}_T(G)$ with $G = \{(x, y) \mapsto (h(x) - y) \colon h \in H\}$. Furthermore, $\widehat{\mathfrak{R}}_T(G)$ can be analyzed as follows:

$$\begin{aligned}\widehat{\mathfrak{R}}_T(G) &= \frac{1}{T} \operatorname*{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in H}\sum_{t=1}^{T}\sigma_t(h(x_t) - y_t)\right] \\ &= \frac{1}{T} \operatorname*{E}_{\boldsymbol{\sigma}}\left[\sup_{h\in H}\sum_{t=1}^{T}\sigma_t h(x_t)\right] + \frac{1}{T} \operatorname*{E}_{\boldsymbol{\sigma}}\left[\sum_{t=1}^{T} -\sigma_t y_t\right] = \widehat{\mathfrak{R}}_T(H),\end{aligned}$$

since $\mathrm{E}_{\boldsymbol{\sigma}}[\sum_{t=1}^{T} -\sigma_t y_t] = 0$. Taking the expectation of both sides yields a similar inequality for Rademacher complexities. Thus, in the statement of the previous theorem, $\mathfrak{R}_T(H_L)$ can be replaced with $qM^{q-1}\mathfrak{R}_T(H)$ when L is the L_q loss.

Observe that the bound of Theorem 1 is tight as a function of the divergence measure (discrepancy) we are using. Consider for example the case where $D_1 = \ldots = D_T$, then a standard Rademacher complexity generalization bound holds for all $h \in H$:

$$\mathcal{L}_{D_T}(h) \leq \frac{1}{T}\sum_{t=1}^{T} L(h(x_t), y_t) + 2\mathfrak{R}_T(H_L) + O(1/\sqrt{T}).$$

Now, our generalization bound for $\mathcal{L}_{D_{T+1}}(h)$ includes only the additive term $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1})$, but by definition of the discrepancy, for any $\epsilon > 0$, there exists $h \in H$ such that the inequality $|\mathcal{L}_{D_{T+1}}(h) - \mathcal{L}_{D_T}(h)| < \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + \epsilon$ holds.

Next, we present PAC learning bounds for empirical risk minimization. Let h_T^* be a best-in class hypothesis in H, that is one with the best expected loss. By a similar reasoning as in theorem 1, we can show that with probability $1 - \frac{\delta}{2}$ we have

$$\frac{1}{T}\sum_{t=1}^{T} L(h_T^*(x_t), y_t) \leq \mathcal{L}_{D_{T+1}}(h_T^*) + 2\mathfrak{R}_T(H_L) + \frac{1}{T}\sum_{t=1}^{T}\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + 2M\sqrt{\frac{\log\frac{2}{\delta}}{2T}}.$$

Let h_T be a hypothesis returned by empirical risk minimization (ERM). Combining this inequality with the bound of theorem 1 while using the definition of h_T and using the union bound, we obtain that with probability $1 - \delta$ the following holds:

$$\mathcal{L}_{D_{T+1}}(h_T) - \mathcal{L}_{D_{T+1}}(h_T^*) \leq 4\mathfrak{R}_T(H_L) + \frac{2}{T}\sum_{t=1}^{T}\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + 2M\sqrt{\frac{\log\frac{2}{\delta}}{2T}}. \quad (7)$$

This learning bound indicates a trade-off: larger values of the sample size T guarantee smaller first and third terms; however, as T increases, the average discrepancy term is likely to grow as well, thereby making learning increasingly challenging. This suggests an algorithm similar to empirical risk minimization but limited to the last m examples instead of the whole sample with $m < T$. This algorithm was previously used in [10] for the study of the tracking scenario. We will use it here to prove several theoretical guarantees in the PAC learning model.

Proposition 1. *Let $\Delta \geq 0$. Assume that $(D_t)_{t \geq 0}$ is a sequence of distributions such that $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{t+1}) \leq \Delta$ for all $t \geq 0$. Fix $m \geq 1$ and let h_T denote the hypothesis returned by the algorithm $\mathcal{A}$ that minimizes $\sum_{t=T-m}^{T} L(h(x_t), y_t)$ after receiving $T > m$ examples. Then, for any $\delta > 0$, with probability at least $1-\delta$, the following learning bound holds:*

$$\mathcal{L}_{D_{T+1}}(h_T) - \inf_{h \in H} \mathcal{L}_{D_{T+1}}(h) \leq 4\mathfrak{R}_m(H_L) + (m+1)\Delta + 2M\sqrt{\frac{\log \frac{2}{\delta}}{2m}}. \tag{8}$$

Proof. The proof is straightforward. Notice that the algorithm discards the first $T-m$ examples and considers exactly m instances. Thus, as in inequality 7, we have:

$$\mathcal{L}_{D_{T+1}}(h_T) - \mathcal{L}_{D_{T+1}}(h_T^*) \leq 4\mathfrak{R}_m(H_L) + \frac{2}{m}\sum_{t=T-m}^{T} \mathrm{disc}(D_t, D_{T+1}) + 2M\sqrt{\frac{\log \frac{2}{\delta}}{2m}}.$$

Now, we can use the triangle inequality to bound $\mathrm{disc}(D_t, D_{T+1})$ by $(T+1-m)\Delta$. Thus, the sum of the discrepancy terms can be bounded by $(m+1)\Delta$. □

To obtain the best learning guarantee, we can select m to minimize the bound just presented. This requires the expression of the Rademacher complexity in terms of m. The following is the result obtained when using a VC-dimension upper bound of $O(\sqrt{d/m})$ for the Rademacher complexity.

Corollary 1. *Fix $\Delta > 0$. Let H be a hypothesis set with VC-dimension d such that for all $m \geq 1$, $\mathfrak{R}_m(H_L) \leq \frac{C}{4}\sqrt{\frac{d}{m}}$ for some constant $C > 0$. Assume that $(D_t)_{t>0}$ is a sequence of distributions such that $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{t+1}) \leq \Delta$ for all $t \geq 0$. Then, there exists an algorithm $\mathcal{A}$ such that for any $\delta > 0$, the hypothesis h_T it returns after receiving $T > \left[\frac{C+C'}{2}\right]^{\frac{2}{3}} (\frac{d}{\Delta^2})^{\frac{1}{3}}$ instances, where $C' = 2M\sqrt{\frac{\log(\frac{2}{\delta})}{2d}}$, satisfies the following with probability at least $1-\delta$:*

$$\mathcal{L}_{D_{T+1}}(h_T) - \inf_{h \in H} \mathcal{L}_{D_{T+1}}(h) \leq 3\left[\frac{C+C'}{2}\right]^{2/3} (d\Delta)^{1/3} + \Delta. \tag{9}$$

Proof: Fix $\delta > 0$. Replacing $\mathfrak{R}_m(H_L)$ by the upper bound $\frac{C}{4}\sqrt{\frac{d}{m}}$ in (8) yields

$$\mathcal{L}_{D_{T+1}}(h_T) - \inf_{h \in H} \mathcal{L}_{D_{T+1}}(h) \leq (C+C')\sqrt{\frac{d}{m}} + (m+1)\Delta.$$

Choosing $m = (\frac{C+C'}{2})^{\frac{2}{3}}(\frac{d}{\Delta^2})^{\frac{1}{3}}$ to minimize the right-hand side gives exactly (9). □

When H has finite VC-dimension d, it is known that $\mathfrak{R}_m(H_L)$ can be bounded by $C\sqrt{d/m}$ for some constant $C > 0$, by using a chaining argument [18, 19, 20]. Thus, the assumption of the corollary holds for many loss functions L, when H has finite VC-dimension.

4 Drifting Tracking Scenario

In this section, we present a simpler proof of the bounds given by [9] for the agnostic case demonstrating that using the discrepancy as a measure of the divergence between distributions leads to tighter and more informative bounds than using the L_1 distance.

Proposition 2. *Let $\Delta > 0$ and let $(D_t)_{t\geq 0}$ be a sequence of distributions such that $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{t+1}) \leq \Delta$ for all $t \geq 0$. Let $m > 1$ and let h_T be as in proposition 1. Then,*

$$\mathop{\mathrm{E}}_{D}[\widehat{M}_{T+1}] - \inf_h \mathcal{L}_{D_{T+1}}(h) \leq 4\mathfrak{R}_m(H_L) + 2M\sqrt{\frac{\pi}{m}} + (m+1)\Delta. \quad (10)$$

Proof. Let $D = \bigotimes_{t=1}^{T+1} D_t$ and $D' = \bigotimes_{t=1}^{T} D_t$. By Fubini's theorem we can write:

$$\mathop{\mathrm{E}}_{D}[\widehat{M}_{T+1}] - \inf_h \mathcal{L}_{D_{T+1}}(h) = \mathop{\mathrm{E}}_{D'}\Big[\mathcal{L}_{D_{T+1}}(h_T) - \inf_h \mathcal{L}_{D_{T+1}}(h)\Big]. \quad (11)$$

Now, let $\phi^{-1}(\delta) = 4\mathfrak{R}_m(H_L) + (m+1)\Delta + 2M\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$, then, by (8), for $\beta > 4\mathfrak{R}_m(h) + (m+1)\Delta$, the following holds:

$$\Pr_{D'}[\mathcal{L}_{D_{T+1}}(h_T) - \inf_h \mathcal{L}_{D_{T+1}}(h) > \beta] < \phi(\beta).$$

Thus, the expectation on the right-hand side of (11) can be bounded as follows:

$$\mathop{\mathrm{E}}_{D'}\Big[\mathcal{L}_{D_{T+1}}(h_T) - \inf_h \mathcal{L}_{D_{T+1}}(h)\Big] \leq 4\mathfrak{R}_m(H_L) + (m+1)\Delta + \int_{4\mathfrak{R}_m(H_L)+(m+1)\Delta}^{\infty} \phi(\beta)d\beta.$$

The last integral can be rewritten as $2M\int_0^2 \frac{d\delta}{\sqrt{m\log\frac{2}{\delta}}} = 2M\sqrt{\frac{\pi}{m}}$ using the change of variable $\delta = \phi(\beta)$. This concludes the proof. □

The following corollary can be shown using the same proof as that of corollary 1.

Corollary 2. *Fix $\Delta > 0$. Let H be a hypothesis set with VC-dimension d such that for all $m > 1$, $4\mathfrak{R}_m(H_L) \leq C\sqrt{\frac{d}{m}}$. Let $(D_t)_{t>0}$ be a sequence of distributions over $\mathcal{X} \times \mathcal{Y}$ such that $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{t+1}) \leq \Delta$. Let $C' = 2M\sqrt{\frac{\pi}{d}}$ and $K = 3\left[\frac{C+C'}{2}\right]^{2/3}$. Then, for $T > \left[\frac{C+C'}{2}\right]^{\frac{2}{3}}(\frac{d}{\Delta^2})^{\frac{1}{3}}$, the following inequality holds:*

$$\mathop{\mathrm{E}}_{D}[\widehat{M}_{T+1}] - \inf_h \mathcal{L}_{D_{T+1}}(h) < K(d\Delta)^{1/3} + \Delta.$$

In terms of definition [5], this corollary shows that algorithm $\mathcal{A}$ $(\Delta, K(d\Delta)^{1/3} + \Delta)$-tracks H. This result is similar to a result of [9] which states that given $\epsilon > 0$ if $\Delta = O(d\epsilon^3)$ then $\mathcal{A}$ (Δ, ϵ)-tracks H. However, in [9], Δ is an upper bound on the L_1 distance and not the discrepancy. Our result provides thus a tighter and more general guarantee than that of [9], the latter because this result is applicable to any loss function and not only the zero-one loss, the former because our bound is based on the Rademacher complexity instead of the VC-dimension and more importantly because it is based on the discrepancy, which is a finer measure of the divergence between distributions than the L_1 distance. Indeed, for any $t \in [1, T]$,

$$\begin{aligned}
\operatorname{disc}_{\mathcal{Y}}(D_t, D_{t+1}) &= \sup_{h \in H} \left|\mathcal{L}_{D_t}(h) - \mathcal{L}_{D_{t+1}}(h)\right| \\
&= \sup_{h \in H} \Big|\sum_{x,y} (D_t(x,y) - D_{t+1}(x,y)) L(h(x), y)\Big|\Big| \\
&\le M \sup_{h \in H} \sum_{x,y} |D_t(x,y) - D_{t+1}(x,y)| = M L_1(D_t, D_{t+1}).
\end{aligned}$$

Furthermore, when the target function f is in H, then the $\mathcal{Y}$-discrepancies can be bounded by the discrepancies $\operatorname{disc}(D_t, D_{T+1})$, which, unlike the L_1 distance, can be accurately estimated from finite samples.

It is important to emphasize that even though our analysis was based on a particular algorithm, that of "truncated" empirical risk minimization, the bounds obtained here cannot be improved upon in the general scenario of drifting distributions, as shown by [10] in the case of binary classification.

We now illustrate the difference between the guarantees we present and those based on the L_1 distance by presenting a simple example for the zero-one loss where the L_1 distance can be made arbitrarily close to 2 while the discrepancy is 0. In that case, our bounds state that the learning problem is as favorable as in the absence of any drifting, while a learning bound with the L_1 distance would be uninformative. Consider

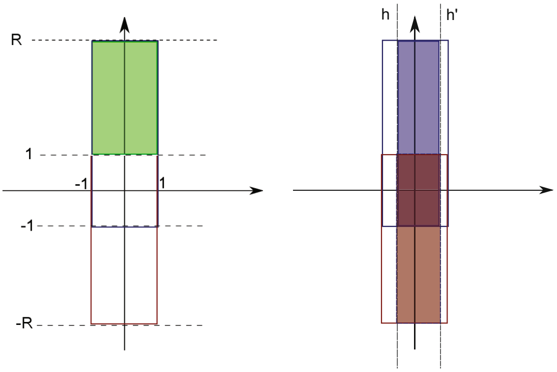


Fig. 1. Figure depicting the difference between the L_1 distance and the discrepancy. In the left figure, the L_1 distance is given by twice the area of the green rectangle. In the right figure, $P(h(x) \neq h'(x))$ is equal to the area of the blue rectangle and $Q(h(x) \neq h'(x))$ is the area of the red rectangle. The two areas are equal, thus $\operatorname{disc}(P, Q) = 0$.

measures P and Q in $\mathbb{R}^2$. Where P is uniform in the rectangle R_1 defined by the vertices $(-1, R)$, $(1, R)$, $(1, -1)$, $(-1, -1)$ and Q is uniform in the rectangle R_2 spanned by $(-1, -R)$, $(1, -R)$, $(-1, 1)$, $(1, 1)$. The measures are depicted in figure 1. The L_1 distance of these probability measures is given by twice the difference of measure in the green rectangle, i.e, $|P - Q| = 2\frac{(R-1)}{R+1}$ this distance goes to 2 as $R \to \infty$. On the other hand consider the zero-one loss and the hypothesis set consisting of threshold functions on the first coordinate, i.e. $h(x, y) = 1$ iff $h < x$. For any two hypotheses $h < h'$ the area of disagreement of this two hypotheses is given by the stripe $S = \{x \colon h < x < h'\}$. But it is trivial to see that $P(S) = P(S \cap R_1) = (h - h')/2$, but also $Q(S) = Q(S \cap R_2) = (h - h')/2$, since this is true for any pair of hypotheses we conclude that $\operatorname{disc}(P, Q) = 0$. This example shows that the learning bounds we presented can be dramatically more favorable than those given in the past using the L_1 distance.

Although this may be viewed as a trivial illustrative example, the discrepancy and the L_1 distance can greatly differ in more complex but realistic cases.

5 On-line to Batch Conversion

In this section, we present learning guarantees for drifting distributions in terms of the regret of an on-line learning algorithm $\mathcal{A}$. The algorithm processes a sample $(x_t)_{t\geq 1}$ sequentially by receiving a sample point $x_t \in \mathcal{X}$, generating a hypothesis h_t, and incurring a loss $L(h(x_t), y_t)$, with $y_t \in \mathcal{Y}$. We denote by R_T the regret of algorithm $\mathcal{A}$ after processing $T \geq 1$ sample points:

$$R_T = \sum_{t=1}^{T} L(h(x_t), y_t) - \inf_{h \in H} \sum_{t=1}^{T} L(h(x_t), y_t).$$

The standard setting of on-line learning assumes an adversarial scenario with no distributional assumption. Nevertheless, when the data is generated according to some distribution, the hypotheses returned by an on-line algorithm $\mathcal{A}$ can be combined to define a hypothesis with strong learning guarantees in the distributional setting when the regret R_T is in $O(\sqrt{T})$ (which is attainable by several regret minimization algorithms) [21, 22]. Here, we extend these results to the drifting scenario and the case of a convex combination of the hypotheses generated by the algorithm. The following lemma will be needed for the proof of our main result.

Lemma 1. *Let $\mathcal{S} = (x_t, y_t)_{t=1}^T$ be a sample drawn from the distribution $D = \bigotimes D_t$ and let $(h_t)_{t=1}^T$ be the sequence of hypotheses returned by an on-line algorithm sequentially processing $\mathcal{S}$. Let $\mathbf{w} = (w_1, \ldots, w_t)^\top$ be a vector of non-negative weights verifying $\sum_{t=1}^T w_t = 1$. If the loss function L is bounded by M then, for any $\delta > 0$, with probability at least $1 - \delta$, each of the following inequalities hold:*

$$\sum_{t=1}^{T} w_t \mathcal{L}_{D_{T+1}}(h_t) \leq \sum_{t=1}^{T} w_t L(h_t(x_t), y_t) + \bar{\Delta}(\mathbf{w}, T) + M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{1}{\delta}}$$

$$\sum_{t=1}^{T} w_t L(h_t(x_t), y_t) \leq \sum_{t=1}^{T} w_t \mathcal{L}_{D_{T+1}}(h_t) + \bar{\Delta}(\mathbf{w}, T) + M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{1}{\delta}},$$

where $\bar{\Delta}(\mathbf{w}, T)$ denotes the average discrepancy $\sum_{t=1}^{T} w_t \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1})$.

Proof. Consider the random process: $Z_t = w_t L(h_t(x_t), y_t) - w_t \mathcal{L}(h_t)$ and let $\mathcal{F}_t$ denote the filtration associated to the sample process. We have: $|Z_t| \leq M w_t$ and

$$\mathop{\mathrm{E}}_{D}[Z_t|\mathcal{F}_{t-1}] = \mathop{\mathrm{E}}_{D}[w_t L(h_t(x_t), y_t)|\mathcal{F}_{t-1}] - \mathop{\mathrm{E}}_{D_t}[w_t L(h_t(x_t), y_t)] = 0$$

The second equality holds because h_t is determined at time $t-1$ and x_t, y_t are independent of $\mathcal{F}_{t-1}$. Thus, by Azuma-Hoeffding's inequality, for any $\delta > 0$, with probability at least $1 - \delta$ the following holds:

$$\sum_{t=1}^{T} w_t \mathcal{L}_{D_t}(h_t) \leq \sum_{t=1}^{T} w_t L(h(x_t), y_t) + M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{1}{\delta}}. \tag{12}$$

By definition of the discrepancy, the following inequality holds for any $t \in [1, T]$:

$$\mathcal{L}_{D_{T+1}}(h_t) \leq \mathcal{L}_{D_t}(h_t) + \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}).$$

Summing up these inequalities and using (12) to bound $\sum_{t=1}^{T} w_t \mathcal{L}_{D_t}(h_t)$ proves the first statement. The second statement can be proven in a similar way. □

The following theorem is the main result of this section.

Theorem 2. *Assume that L is bounded by M and convex with respect to its first argument. Let $h_1, \ldots, h_T$ be the hypotheses returned by $\mathcal{A}$ when sequentially processing $(x_t, y_t)_{t=1}^{T}$ and let h be the hypothesis defined by $h = \sum_{t=1}^{T} w_t h_t$, where $w_1, \ldots, w_T$ are arbitrary non-negative weights verifying $\sum_{t=1}^{T} w_t = 1$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, h satisfies each of the following learning guarantees:*

$$\mathcal{L}_{D_{T+1}}(h) \leq \sum_{t=1}^{T} w_t L(h_t(x_t), y_t) + \bar{\Delta}(\mathbf{w}, T) + M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{1}{\delta}}$$

$$\mathcal{L}_{D_{T+1}}(h) \leq \inf_{h \in H} \mathcal{L}(h) + \frac{R_T}{T} + \bar{\Delta}(\mathbf{w}, T) + M\|\mathbf{w} - \mathbf{u}_0\|_1 + 2M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{2}{\delta}},$$

where $\mathbf{w} = (w_1, \ldots, w_T)^\top$, $\bar{\Delta}(\mathbf{w}, T) = \sum_{t=1}^{T} w_t \mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1})$, and $\mathbf{u}_0 \in \mathbb{R}^T$ is the vector with all its components equal to $1/T$.

Observe that when all weights are all equal to $\frac{1}{T}$, the result we obtain is similar to the learning guarantee obtained in theorem 1 when the Rademacher complexity of H_L is $O(\frac{1}{\sqrt{T}})$. Also, if the learning scenario is i.i.d., then the first sum of the bound vanishes and it can be seen straightforwardly that to minimize the RHS of the inequality we need to set $w_t = \frac{1}{T}$, which results in the known i.i.d. guarantees for on-line to batch conversion [21, 22].

Proof. Since L is convex with respect to its first argument, by Jensen's inequality, we have $\mathcal{L}_{D_{T+1}}(\sum_{t=1}^{T} w_t h_t) \leq \sum_{t=1}^{T} w_t \mathcal{L}_{D_{T+1}}(h_t)$. Thus, by Lemma 1, for any $\delta > 0$, the following holds with probability at least $1 - \delta$:

$$\mathcal{L}_{D_{T+1}}\left(\sum_{t=1}^{T} w_t h_t\right) \leq \sum_{t=1}^{T} w_t L(h_t(x_t), y_t) + \bar{\Delta}(\mathbf{w}, T) + M\|\mathbf{w}\|_2 \sqrt{2 \log \frac{1}{\delta}}. \tag{13}$$

This proves the first statement of the theorem. To prove the second claim, we will bound the empirical error in terms of the regret. For any $h^* \in H$, we can write using $\inf_{h \in H} \frac{1}{T} \sum_{t=1}^T L(h(x_t), y_t) \leq \frac{1}{T} \sum_{t=1}^T L(h^*(x_t), y_t)$:

$$\begin{aligned}
&\sum_{t=1}^T w_t L(h_t(x_t), y_t) - \sum_{t=1}^T w_t L(h^*(x_t), y_t) \\
&= \sum_{t=1}^T \Big(w_t - \frac{1}{T}\Big) [L(h_t(x_t), y_t) - L(h^*(x_t), y_t)] + \frac{1}{T} \sum_{t=1}^T [L(h_t(x_t), y_t) - L(h^*(x_t), y_t)] \\
&\leq M \|\mathbf{w} - \mathbf{u}_0\|_1 + \frac{1}{T} \sum_{t=1}^T L(h_t(x_t), y_t) - \inf_h \frac{1}{T} \sum_{t=1}^T L(h(x_t), y_t) \\
&\leq M \|\mathbf{w} - \mathbf{u}_0\|_1 + \frac{R_T}{T}.
\end{aligned}$$

Now, by definition of the infimum, for any $\epsilon > 0$, there exists $h^* \in H$ such that $\mathcal{L}_{D_{T+1}}(h^*) \leq \inf_{h \in H} \mathcal{L}_{D_{T+1}}(h) + \epsilon$. For that choice of h^*, in view of (13), with probability at least $1 - \delta/2$, the following holds:

$$\mathcal{L}_{D_{T+1}}(h) \leq \sum_{t=1}^T w_t L(h^*(x_t), y_t) + M \|\mathbf{w}_{-}\mathbf{u}_0\|_1 + \frac{R_T}{T} + \bar{\Delta}(\mathbf{w}, T) + M \|\mathbf{w}\|_2 \sqrt{2 \log \frac{2}{\delta}}.$$

By the second statement of Lemma 1, for any $\delta > 0$, with probability at least $1 - \delta/2$,

$$\sum_{t=1}^T w_t L(h^*(x_t), y_t) \leq \mathcal{L}_{D_{T+1}}(h^*) + \bar{\Delta}(\mathbf{w}, T) + M \|\mathbf{w}\|_2 \sqrt{2 \log \frac{2}{\delta}}.$$

Combining these last two inequalities, by the union bound, with probability at least $1 - \delta$, the following holds with $B(\mathbf{w}, \delta) = M \|\mathbf{w}_{-}\mathbf{u}_0\|_1 + \frac{R_T}{T} + 2M \|\mathbf{w}\|_2 \sqrt{2 \log \frac{2}{\delta}}$:

$$\begin{aligned}
\mathcal{L}_{D_{T+1}}(h) &\leq \mathcal{L}_{D_{T+1}}(h^*) + 2\bar{\Delta}(\mathbf{w}, T) + B(\mathbf{w}, \delta) \\
&\leq \inf_{h \in H} \mathcal{L}_{D_{T+1}}(h) + \epsilon + 2\bar{\Delta}(\mathbf{w}, T) + B(\mathbf{w}, \delta).
\end{aligned}$$

The last inequality holds for all $\epsilon > 0$, therefore also for $\epsilon = 0$ by taking the limit. □

6 Algorithm

The results of the previous section suggest a natural algorithm based on the values of the discrepancy between distributions. Let $(h_t)_{t=1}^T$ be the sequence of hypotheses generated by an on-line algorithm. Theorem 2 provides a learning guarantee for any convex combination of these hypotheses. The convex combination based on the weight vector $\mathbf{w}$ minimizing the bound of Theorem 2 benefits from the most favorable guarantee. This leads to an algorithm for determining $\mathbf{w}$ based on the following convex optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \lambda\|\mathbf{w}\|_2^2 + \sum_{t=1}^{T} w_t \left(\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) + L(h_t(x_t), y_t)\right) \\ \text{subject to:} \quad & \Big(\sum_{t=1}^{T} w_t = 1\Big) \wedge (\forall t \in [1, T], w_t \geq 0), \end{aligned} \tag{14}$$

where $\lambda \geq 0$ is a regularization parameter. This is a standard QP problem that can be efficiently solved using a variety of techniques and available software.

In practice, the discrepancy values $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1})$ are not available since they require labeled samples. But, in the deterministic scenario where the labeling function f is in H, we have $\mathrm{disc}_{\mathcal{Y}}(D_t, D_{T+1}) \leq \mathrm{disc}(D_t, D_{T+1})$. Thus, the discrepancy values $\mathrm{disc}(D_t, D_{T+1})$ can be used instead in our learning bounds and in the optimization (14). This also holds approximately when f is not in H but is close to some $h \in H$.

As shown in [14], given two (unlabeled) samples of size n from D_t and D_{T+1}, the discrepancy $\mathrm{disc}(D_t, D_{T+1})$ can be estimated within $O(1/\sqrt{n})$, when $\mathfrak{R}_n(H_L) = O(1/\sqrt{n})$. In many realistic settings, for tasks such as spam filtering, the distribution D_t does not change within a day. This gives us the opportunity to collect an independent *unlabeled* sample of size n from each distribution D_t. If we choose $n \gg T$, by the union bound, with high probability, all of our estimated discrepancies will be within $O(1/\sqrt{T})$ of their exact counterparts $\mathrm{disc}(D_t, D_{T+1})$.

Additionally, in many cases, the distributions D_t remain unchanged over some longer periods (cycles) which may be known to us. This in fact typically holds for some tasks such as spam filtering, political sentiment analysis, some financial market prediction problems, and other problems. For example, in the absence of any major political event such as a debate, speech, or a prominent measure, we can expect the political sentiment to remain stable. In such scenarios, it should be even easier to collect an unlabeled sample from each distribution. More crucially, we do not need then to estimate the discrepancy for all $t \in [1, T]$ but only once for each cycle.

6.1 Experiments

Here, we report the results of preliminary experiments demonstrating the performance of our algorithm. We tested our algorithm on synthetic data in a regression setting. The testing and training data were created as follows: instances were sampled from a two-dimensional Gaussian random variables $\mathcal{N}(\boldsymbol{\mu}_t, 1)$. The objective function at each time was given by $y_t = \mathbf{w}_t \cdot \mathbf{x}_t$. The weight vectors $\mathbf{w}_t$ and mean vectors $\boldsymbol{\mu}_t$ were selected as follows: $\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \mathbf{U}$ and $\mathbf{w}_t = R_\theta \mathbf{w}_{t-1}$, where $\mathbf{U}$ is the uniform random variable over $[-.1, +.1]^2$ and R_θ a rotation of magnitude θ distributed uniformly over $(-1, 1)$. We used the Widrow-Hoff algorithm [23] as our base on-line algorithm to determine h_t. After receiving T examples, we tested our final hypothesis on 100 points taken from the same Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{T+1}, 1)$. We ran the experiment 50 times for different amounts of sample points and took the average performance of our classifier. For these experiments, we are considering the ideal situation where the discrepancy values are given.

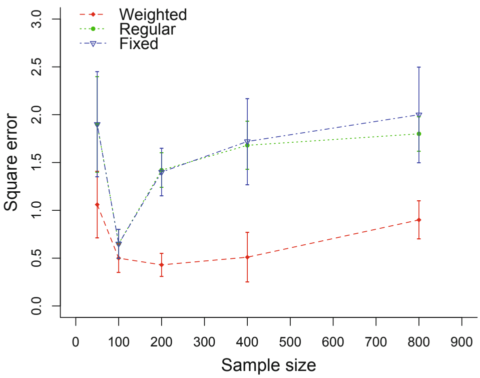


Fig. 2. Comparison of the performance of three algorithms as a function of the sample size T. `Weighted` stands for the algorithm described in this paper, `Regular` for an algorithm that averages over all the hypotheses, and `Fixed` for the algorithm that averages only over the last 100 hypotheses.

We compared the performance of our algorithm with that of the algorithm that (uniformly) averages all of the hypotheses and with that of the algorithm that averages only the last 100 hypotheses generated by the perceptron algorithm. Figure 2 shows the results of our experiments in the first setting. Observe that the error increases with the sample size. While the analysis of Section 3 could provide an explanation of this phenomenon in the case of the uniform averaging algorithm, in principle, it does not explain why the error also increases in the case of our algorithm. The answer to this can be found in the setting of the experiment. Notice that the Gaussians considered are moving their center and that the squared loss grows proportional to the radius of the smallest sphere containing the sample. Thus, as the number of points increases, so does the maximum value of the loss function in the test set. Finally, keep in mind that the accuracy of our algorithm can drastically change of course depending on the choice of the online algorithm used.

7 Conclusion

We presented a theoretical analysis of the problem of learning with drifting distributions in the batch setting. Our learning guarantees improve upon previous ones based on the L_1 distance, in some cases substantially, and our proofs are simpler and concise. These bounds benefit from the notion of discrepancy which seems to be the natural measure of the divergence between distributions in a drifting scenario. This work motivates a number of related studies, in particular a discrepancy-based analysis of the scenario introduced by [13] and further improvements of the algorithm we presented, in particular by exploiting the specific on-line learning algorithm used.

References

[1] Valiant, L.G.: A theory of the learnable. ACM Press, New York (1984)

[2] Vapnik, V.N.: Statistical Learning Theory. J. Wiley & Sons (1998)

[3] Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge University Press (2006)

[4] Herbster, M., Warmuth, M.: Tracking the best expert. Machine Learning 32(2), 151–178 (1998)

[5] Herbster, M., Warmuth, M.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)

[6] Cavallanti, G., Cesa-Bianchi, N., Gentile, C.: Tracking the best hyperplane with a simple budget perceptron. Machine Learning 69(2/3), 143–167 (2007)

[7] Helmbold, D.P., Long, P.M.: Tracking drifting concepts by minimizing disagreements. Machine Learning 14(1), 27–46 (1994)

[8] Bartlett, P.L.: Learning with a slowly changing distribution. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT 1992, pp. 243–252. ACM, New York (1992)

[9] Long, P.M.: The complexity of learning according to two models of a drifting environment. Machine Learning 37, 337–354 (1999)

[10] Barve, R.D., Long, P.M.: On the complexity of learning from drifting distributions. Information and Computation 138(2), 101–123 (1997)

[11] Freund, Y., Mansour, Y.: Learning under Persistent Drift. In: Ben-David, S. (ed.) EuroCOLT 1997. LNCS, vol. 1208, pp. 109–118. Springer, Heidelberg (1997)

[12] Bartlett, P.L., Ben-David, S., Kulkarni, S.: Learning changing concepts by exploiting the structure of change. Machine Learning 41, 153–174 (2000)

[13] Crammer, K., Even-Dar, E., Mansour, Y., Vaughan, J.W.: Regret minimization with concept drift. In: COLT, pp. 168–180 (2010)

[14] Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: Learning bounds and algorithms. In: Proceedings of COLT. Omnipress, Montréal (2009)

[15] Valiant, P.: Testing symmetric properties of distributions. SIAM J. Comput. 40(6), 1927–1968 (2011)

[16] Rakhlin, A., Sridharan, K., Tewari, A.: Online learning: Random averages, combinatorial parameters, and learnability (2010)

[17] Cortes, C., Mohri, M.: Domain Adaptation in Regression. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 308–323. Springer, Heidelberg (2011)

[18] Dudley, R.M.: A course on empirical processes. Lecture Notes in Math., vol. 1097, pp. 2–142 (1984)

[19] Pollard, D.: Convergence of Stochastic Processess. Springer, New York (1984)

[20] Talagrand, M.: The Generic Chaining. Springer, New York (2005)

[21] Littlestone, N.: From on-line to batch learning. In: Proceedings of the Second Annual Workshop on Computational Learning Theory, pp. 269–284. Morgan Kaufmann Publishers Inc. (1989)

[22] Cesa-Bianchi, N., Conconi, A., Gentile, C.: On the generalization ability of on-line learning algorithms. In: NIPS, pp. 359–366 (2001)

[23] Widrow, B., Hoff, M.E.: Adaptive switching circuits. Neurocomputing: Foundations of Research (1988)

On the Hardness of Domain Adaptation and the Utility of Unlabeled Target Samples

Shai Ben-David and Ruth Urner

School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
CANADA
{shai,rurner}@cs.uwaterloo.ca

Abstract. The Domain Adaptation problem in machine learning occurs when the test and training data generating distributions differ. We consider the covariate shift setting, where the labeling function is the same in both domains. Many works have proposed algorithms for Domain Adaptation in this setting. However, there are only very few generalization guarantees for these algorithms. We show that, without strong prior knowledge about the training task, such guarantees are actually unachievable (unless the training samples are prohibitively large). The contributions of this paper are two-fold: On the one hand we show that Domain Adaptation in this setup is hard. Even under very strong assumptions about the relationship between source and target distribution and, on top of that, a realizability assumption for the target task with respect to a small class, the required total sample sizes grow unboundedly with the domain size. On the other hand, we present settings where we achieve almost matching upper bounds on the sum of the sizes of the two samples. Moreover, the (necessarily large) samples can be mostly unlabeled (target) samples, which are often much cheaper to obtain than labels. The size of the labeled (source) sample shrinks back to standard dependence on the VC-dimension of the concept class. This implies that unlabeled target-generated data is provably beneficial for DA learning.

Keywords: Statistical Learning Theory, Domain Adaptation, Sample Complexity, Unlabeled Data.

1 Introduction

Much of the theoretical analysis of machine learning focuses on a model where the training and test data are generated by the *same* underlying distribution. While this may sometimes be a good approximation of reality, in many practical tasks this assumption cannot be justified - it is often the case that the data used for training the learner is generated by a different probability distribution than the one generating the test (target) data. The data generating distribution might change over time, or one may wish to apply classifiers trained on some available training data to other domains for which there is no labeled training

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 139–153, 2012.

data available. E.g., in natural language processing one might only have labeled documents of a certain type available, say legal documents, but needs to build a classifier to label documents of a different type, say medical documents This scenario is referred to as *Domain Adaption* (DA). Domain Adaptation occurs in many practical situations and is frequently addressed in experimental research.

Obviously, the success of Domain Adaptation learning depends on the relationship between the process generating the training data to the one that generates the target (test) data. Much of the Domain Adaptation research addresses learning under the *covariate shift* assumption, where both the training data generating distribution and the test data generating distribution share the same labeling function (and differ only in their marginals). Several algorithmic paradigms have been proposed for Domain Adaptation learning in this setup. Discrepancy Minimization [1], Importance Weighting [2], [3], Density Ratio Estimation [4], and more. However, there are very few satisfying theoretical guarantees on the success of these paradigms.

In this work, we show that, in contrast to the impression that one may get from these proposed methods, without strong prior knowledge about the training task, guarantees on the success of Domain Adaptation learning under covariate shift are impossible (unless the training samples are prohibitively large).

To allow a more concrete description of our results, we now outline the modeling assumptions that underly our work.

The Input Data Available to the Learner: In this work, we consider a model in which the learner has access to source-generated random labeled samples and to unlabeled samples generated by the distribution governing the target domain. The focus of this paper is analyzing the sample complexity of DA learning with respect to each of these types of samples.

The Relationship between the Source and Target Data-Generating Distributions: Besides the covariate shift assumption, we consider several measures for the relationship between the marginals of the source and target distributions. Mainly, we employ a bound on the ratio of weights of sets in a prescribed collection of domain subsets.

The Learner's Prior Knowledge about the Task: Prior knowledge about a learning task is necessary for any guarantees of success (this is the "no-free-lunch" principle). Since the goal of the learner is to come up with a low-error predictor for the target task, prior knowledge about that task is required to guarantee its success. However, interestingly, the few papers that do provide error bounds guarantees for DA learning also assume that the learner has rather strong prior knowledge about the source task. In particular, both [5] and [1] assumed that the learner has knowledge of a hypothesis class, that bridges the discrepancies between source and target tasks, a class that has good approximation error with respect to *both* tasks.

The other studies cited above, that address DA learning in the covariate shift setup and do not assume such prior knowledge, also lack satisfying error bound guarantees. Is this just a coincidence? To answer this question, we focus on DA learning when the prior knowledge available to the learner is about just the target task. We assume that the learner has prior knowledge of a class that contains a function with zero classification error on the target distribution, but we do not make assumptions on the behavior of that class with respect to the source distribution. It turns out that under such conditions DA learning cannot be guaranteed to succeed as long as the training sample sizes are not excessively large.

Summary of Our Results: The first contribution of this paper is a rather strong negative result. We show, in Section 3, that even under strong assumptions, namely, covariate shift, a bound on the point-wise weight-ratio between the marginal distributions and realizability of the target distribution by a class of VC-dimension 1, the number of needed training examples (the sum of the number source-generated labeled examples and the number of target-generated unlabeled examples) may need to grow to infinity with the size of the domain set. In other words, even when learning from task-generated labeled examples is trivial, learning based on a sample generated by a closely-related source distribution and unlabeled data yields very high sample complexity. As an aside, our lower bound in that section employs a reduction from a novel probabilistic task that may find further applications in machine learning theory. We also prove a similar sample complexity lower bound, for the case of a Euclidean domain, assuming that the labeling function is known to be Lipschitz. The bound is exponential in the Euclidean dimension of the domain. These lower bounds apply regardless of any choice of a learning algorithm.

Our lower bounds almost match an upper bound by [6] on the size of the labeled source-generated training for DA learning, achieved by a simple nearest neighbor algorithm. However, in many practical DA learning tasks, unlabeled examples, even from the target domain, are easier to come by than labeled source examples. A natural followup question concerns the possibility of trading source-generated labeled examples against target-generated unlabeled data.

In Section 4, we present two scenarios in which we show that DA learning is indeed possible on the basis of (large) unlabeled samples together with a labeled sample whose size is basically determined by the VC-dimension of the concept class (as well as the discrepancy between the two marginal distributions and the usual accuracy and confidence parameters, ϵ and δ, but not depending on the domain size). In the first scenario we assume that the learner has the prior knowledge of a concept class relative to which the target distribution is realizable with margins. In the second scenario we assume finiteness of the domain, (but no margin- or Lipschitz-assumptions on the involved labeling functions).

Combining the lower and upper bounds described above, we conclude that when the size of the source-generated labeled sample is small (e.g., independent of the domain size) then, without strong prior knowledge about the source distribution, successful DA is possible only by adaptive algorithms (algorithms that make use of target generated data, such as ours). In other words, in that setup, unlabeled target-generated data is provably necessary and beneficial for DA learning.

Related Work. Here, we discuss some of the learning paradigms that have been proposed to facilitate successful DA under the covariate shift assumption.

Much of the work on Domain Adaptation addresses DA under the covariate shift assumption, where the conditional (label) distributions of the target and source data are identical. Under this assumption, source and target data distribution only differ in their marginals. Therefore, a natural approach for covariate shift DA learning is to reweigh the training sample to make it as similar as possible to a sample generated by the target distribution (e.g., [7] and [8]). Clearly, such reweighing, when implemented precisely, turns the distribution over source-generated training samples into a distribution close to that over target-generated samples, thus overcoming the source-target discrepancy. Similar ideas underly the methods of Discrepancy Minimization [1], Importance Weighting [2], [3] and Density Ratio estimation [4].

However, our sample complexity lower bound, Theorem 1, implies that in order to obtain such reliable reweighing the learner needs access to huge samples, of sizes that go to infinity with the size of the underlying domain.

A weight ratio assumption has previously been considered by [2]. They propose a Domain Adaptation paradigm with provable success rates, assuming the learner can access the values of the point-wise weight ratio. They also acknowledge the excessive strength of an assumption that bounds the point-wise weight ratio and discuss some relaxations of this. The apparent contradiction between their sample complexity upper bounds and our lower bounds is due the the sample complexity of estimating the weight ratio (that their analysis assumes is given to the learner). To address the sample complexity of estimating the weight-ratio function, [2] refer to [9]. However, the sample complexity analysis of [9] assumes that all the points of the labeled source sample S occur also in the unlabeled target sample T. When S and T are sampled independently, as is the case in the covariate shift DA learning setting, the size of T required to guarantee hitting every member in S grows unboundedly with the size of the support of the target distribution.

Distribution independent error bounds for Domain Adaptation learning were shown in an analysis of the problem with respect to a given "bridging" hypotheses class H – a class that is assumed to provide good approximation to *both* the source and the target distributions. [5] propose to measure the relatedness of the two distributions by two parameters that depend on the class H; the discrepancy between the marginal distributions by the so-called $d_{\mathcal{A}}$ distance (as introduced by [10]), and a notion of a "joint approximation error" of the class with respect to source and target. The paper provides upper bounds, in terms of these parameters, on the error of the simplest conservative Domain Adaptation algorithm—the empirical risk minimization (ERM) over the training data. A follow-up paper, [1], extends the $d_{\mathcal{A}}$ distance to real-valued function classes and loss functions other than the 0-1 loss. In addition, they propose a non-conservative learning paradigm— a certain reweighing procedure aimed to minimize the discrepancy between the source and target input samples. This is further extended to regression problems in [11].

Lower bounds for DA learning under covatiate shift are also presented in [12]. They prove worst case lower bounds on the sample complexity of reweighing paradigms for DA learning in the setup of a bridging hypothesis class. Both that paper and our lower bound consider the covariate shift setup and, on top of that, assume that the marginals of the training and test data are "similar". However, the notion of the distributions' similarity in [12] is weaker - having small $d_{\mathcal{A}}$ distance. The lower bound in that paper takes advantage of the looseness of the discrepancy $d_{\mathcal{A}}$ and constructs a scenario in which, although the marginals look similar w.r.t. that distance, the target distribution is supported on regions that have zero weight in the training distribution. It is no surprise that under such circumstances DA may fail (the training sample misses significant chunks of the target distribution). In our work, we consider the strong assumption that the point-wise density ratio between the two distributions is bounded from below by 0.5 (implying that no region that is significant w.r.t. the target is missed by the source distribution). Just the same, we show that any DA algorithm may fail, even if it just has to decide between the all-zero and the all-one predictors for the target. The failure of DA in this a setting is quite surprising and requires a novel proof technique.

Another aspect of the current paper - theoretical analysis of the utility of target generated unlabeled samples for DA learning - has also been addressed in [6]. However, they show such a benefit for the restricted setting of *proper* DA learning (where the learner is required to output a classifier from a predefined class), whereas we prove that unlabeled samples are also beneficial in the general DA learning framework, when the learner is allowed to output arbitrary predictors.

2 Preliminaries

2.1 Definitions

We consider the following formal setup for Domain Adaptation learning: We let $\mathcal{X}$ denote the domain set and let $l : \mathcal{X} \to \{0,1\}$ denote the labeling function of our learning task. There are two distributions over $\mathcal{X}$, the *source distribution* P_S and the *target distribution* P_T. Given a function $h : \mathcal{X} \to \{0,1\}$, we let $\mathrm{Err}^l_S(h) = \Pr_{x \sim P_S}(h(x) \neq l(x))$ denote the *source error* and $\mathrm{Err}^l_T(h) = \Pr_{x \sim P_T}(h(x) \neq l(x))$ denote the *target error* of h with respect to l.

A Domain Adaptation learner takes as input a labeled i.i.d. sample S drawn according to P_S and labeled by l and an unlabeled i.i.d. sample T drawn according to P_T and aims to output a label predictor $h : \mathcal{X} \to \{0,1\}$ with low target error. Formally, a *Domain Adaptation (DA) learner* is a function

$$A : \bigcup_{m=1}^{\infty} \bigcup_{n=1}^{\infty} (\mathcal{X} \times \{0,1\})^m \times \mathcal{X}^n \to \{0,1\}^{\mathcal{X}} .$$

Clearly, guarantees on the success of Domain Adaptation learners are impossible without assumptions on the relatedness of source and target distribution. Therefore, we investigate the success of DA learners for specified classes of pairs of source and target distributions (in standard PAC learning theory, the learner is required to succeed for all such pairs, where source and target are exactly the same). We now set our framework to measure the success of DA-learners:

Definition 1 (DA-learnablity). *Let $\mathcal{W}$ be a class of triples (P_S, P_T, l) of source and target distributions over some domain $\mathcal{X}$ and a labeling function, and let $\mathcal{A}$ be a DA learner. We say that $\mathcal{A}$ (ϵ, δ, m, n)*-solves DA for the class *$\mathcal{W}$, if, for all triples $(P_S, P_T, l) \in \mathcal{W}$, when given access to a sample S of size m, generated i.i.d. by P_S and labeled by l, and an unlabeled sample T of size n, generated i.i.d by P_T, with probability at least $1 - \delta$ (over the choice of the samples S and T) $\mathcal{A}$ outputs a function h with $\mathrm{Err}_T^l(h) \leq \epsilon$.*

2.2 Relatedness Assumptions

One basic observation about DA learning is that it may become impossible when the source and target distributions are supported on disjoint domain regions. To guard against such scenarios, it is common to assume that there is some non-zero lower bound to the pointwise density ratio between the two distributions. However, this is often an unrealistic assumption. To overcome this drawback, we propose the following relaxation of that assumption.

Definition 2 (Weight ratio). *Let $\mathcal{B} \subseteq 2^{\mathcal{X}}$ be a collection of subsets of the domain $\mathcal{X}$ measurable with respect to both P_S and P_T. We define the* weight ratio *of the source distribution and the target distribution with respect to $\mathcal{B}$ as*

$$C_{\mathcal{B}}(P_S, P_T) = \inf_{\substack{b \in \mathcal{B}(\mathcal{X}) \\ P_T(b) \neq 0}} \frac{P_S(b)}{P_T(b)},$$

We denote the weight ratio with respect to the collection of all sets that are P_S- and P_T-measurable by $C(P_S, P_T)$.

These measures become relevant for Domain Adaptation when bounded away from zero.

Note that in the case of discrete distributions $C(P_S, P_T)$ is equal to the pointwise weight ratio $C(P_S, P_T) = C_{\{\{x\}: x \in \mathcal{X}\}}(P_S, P_T)$. For every $\mathcal{B} \subseteq 2^{\mathcal{X}}$ we have $C(P_S, P_T) \leq C_{\mathcal{B}}(P_S, P_T)$, thus bounding the pointwise weight ratio away from 0 is the strongest restriction. Our lower bounds hold with a bound on the pointwise weight ratio. For the positive results, we will employ the weight ratio for some limited collections of subsets of the domain space. Note that if $\mathcal{B}$ is a set of finite VC-dimension, and we take the infimum in Definition 2 only over sets b with $P_T(b) \geq \eta$, for some $\eta > 0$, then we can estimate the weight ratio from finite samples (see Theorem 3.4 and the subsequent discussion of [10]).

In previous work on analysis of DA, the following distance has been employed:

Definition 3 ($d_{H\Delta H}$-Distance). *Let $\mathcal{X}$ be some domain, P_S and P_T distributions over $\mathcal{X}$ and H a hypothesis class. Then the $d_{H\Delta H}$-distance is defined as*

$$d_{H\Delta H}(P_S, P_T) = \sup_{A \in H\Delta H} |P_T(A) - P_S(A)|$$

where $H\Delta H = \{h_1 \Delta h_2 \mid h_1, h_2 \in H\}$, and $h_1 \Delta h_2 = \{x \in \mathcal{X} \mid h_1(x) \neq h_2(x)\}$.

2.3 Prior Knowledge Assumptions

We investigate Domain Adaptation under a realizability assumption for the target task. While realizability is a strong assumption, that renders standard learning easy, our lower bound shows that Domain Adaptation remains a challenging task even under this condition. Formally, for a hypothesis class $H \subseteq \{0,1\}^{\mathcal{X}}$ we let $\mathrm{opt}^l_T(H) = \min\{\mathrm{Err}^l_T(h) | h \in H\}$ denote the target approximation error of the class. We say that a class H *realizes* the target distribution with labeling function l if $\mathrm{opt}^l_T(H) = 0$.

If the domain is a euclidean space, $\mathcal{X} \subseteq \mathbb{R}^d$ for some dimension d, we denote the ball of radius r around some domain point x by $B_r(x)$ and formalize the margin assumption as follows:

Definition 4. *Let $\mathcal{X} \subseteq \mathbb{R}^d$, P a distribution over $\mathcal{X}$ and $h : \mathcal{X} \to \{0,1\}$ a classifier. We say that h is a* γ-margin classifier *with respect to P if for all $x \in \mathcal{X}$ whenever $P(B_\gamma(x)) > 0$ then $h(y) = h(z)$ holds for all $y, z \in B_\gamma(x)$.*

We say that a class H *realizes* the distribution *with margin* γ if the optimal (zero-error) classifier is a γ-margin classifier. Note that h being a γ-margin classifier with respect to P is equivalent to h satisfying the Lipschitz-property with Lipschitz constant $1/2\gamma$ on the support of P. We will call the property in Definition 4 *Lipschitzness* when making an assumption about the labeling function and a *margin assumption* when referring to the optimal classifier.

3 Lower Bounds for Realizable Domain Adaptation

The lower bound in this section shows that no small amount of labeled source and unlabeled target data suffices for DA under covariate shift. We show that even in the case where the learner knows that the target is realizable by the class $H_{1,0}$ that contains only the all-1 and the all-0 labeling functions, a class of VC-dimension 1, the sizes of the source sample and the target sample need to be (roughly) as large as $\sqrt{|\mathcal{X}|}$ for Domain Adaptation to be possible.

Theorem 1. *For every finite domain $\mathcal{X}$, for every ϵ and δ with $\epsilon + \delta < 1/2$, no algorithm can (ϵ, δ, s, t)-solve the DA problem for the class $\mathcal{W}$ of triples (P_S, P_T, l) with $C(P_S, P_T) \geq 1/2$, $d_{H_{1,0}\Delta H_{1,0}}(P_S, P_T) = 0$ and $\mathrm{opt}^l_T(H_{1,0}) = 0$ if $s + t < \sqrt{(1 - 2(\epsilon + \delta))|\mathcal{X}|} - 2$.*

This hardness result is quite surprising since it applies to a setting in which DA learning is seemingly as easy as it can get; the prior knowledge about the target task is so strong that one labeled target example would suffice for finding a zero error classifier. Furthermore, the source and target distributions share the same deterministic labeling function, and the marginals of the two distributions are similar from both the $d_{H\Delta H}$-distance and the weight ratio perspectives (namely the source probability of any domain point is at least half its target probability).

Several conclusions can be drawn from this lower bound:

1. If one assumes target realizability by a small hypothesis class but does not assume that there is such a class that has small approximation error with respect to both the source and the target, the DA sample complexity cannot be bounded as a function of only on the VC-dimension of the class that realizes the target distribution. This is in sharp contrast to the sample complexity of learning without discrepancy between the training and test data.

2. It is necessary to have some data generated by the target distribution available, if the number of labeled examples is only allowed to depend on the VC-dimension of the class.

3. Since under the covariate shift assumption having access to the ratio between a the source and target probability of domain points allows successful DA learning, our result implies that the sample sizes needed to obtain useful approximations of that ratio, as required, e.g., for importance weighting techniques, are prohibitively high.

A lower bound in terms of Lipschitzness. Theorem 2 implies a lower bound for the size of the sample for infinite domains under the additional assumption that the labeling function satisfies the Lipschitz property for some Lipschitz constant λ. Again, this lower bound holds under the assumption that the target is realizable by a two-function-class.

Theorem 2. *Let $\mathcal{X} = [0,1]^d$, $\epsilon > 0$ and $\delta > 0$ be such that $\epsilon + \delta < 1/2$, let $\lambda > 1$ and let $\mathcal{W}_\lambda$ be the set of triples (P_S, P_T, l) of distributions over $\mathcal{X}$ with $\mathrm{opt}_T^l(H_{1,0}) = 0$, $C(P_S, P_T) \geq 1/2$, $d_{H_{1,0}\Delta H_{1,0}}(P_S, P_T) = 0$ and λ-Lipschitz labeling functions l. Then no DA-learner can (s,t,ϵ,δ)-solve the DA-problem for the class $\mathcal{W}_\lambda$ unless $s + t \geq \sqrt{(\lambda+1)^d(1 - 2(\epsilon+\delta))} - 2$.*

Proof. Let $\mathcal{G} \subseteq \mathcal{X}$ be the points of a grid in $[0,1]^d$ with distance $1/\lambda$. Then we have $|\mathcal{G}| = (\lambda+1)^d$. Then the class $\mathcal{W}_\lambda$ contains all triples (P_S, P_T, l), where the support of P_S and P_T is $\mathcal{G}$, $\mathrm{opt}_T^l(H_{1,0}) = 0$, $C(P_S, P_T) \geq 1/2$, $d_{H_{1,0}\Delta H_{1,0}}(P_S, P_T) = 0$ and arbitrary labeling functions $l : \mathcal{G} \to \{0,1\}$, as every such function is λ-Lipschitz. As $\mathcal{G}$ is finite, the bound follows from Theorem 1.

3.1 Proof of Theorem 1

We obtain our lower bound by reducing the following problem to DA:

The Left/Right Problem. We consider the problem of distinguishing two distributions from finite samples. The Left/Right Problem was introduced in [13]:

Input: Three finite samples, L, R and M of points from some domain set $\mathcal{X}$.

Output: Assuming that L is an an *i.i.d.* sample from some distribution P over $\mathcal{X}$, that R is an an *i.i.d.* sample from some distribution Q over $\mathcal{X}$, and that M is an *i.i.d.* sample generated by one of these two probability distributions, was M generated by P or by Q ?

We first derive a lower bound on the sample size needed to solve the Left/Right problem in Lemma 1. Then we reduce the Left/Right problem to Domain Adaptation under target realizability, thereby obtaining a lower bound on the sample size needed to solve DA. Intuitively, one can not answer the Left/Right-question if the sample M intersects neither the sample L nor the sample R. This yields a lower bound for the Left/Right problem in the order of the square-root of the domain size. The idea of the reduction to Domain Adaptation is to define a source distribution that is a balanced mixture of P and Q with a labeling function that gives label 1 to points from L (generated by P) and label 0 to points from R (generated by Q). The sample M can then be considered an unlabeled sample from a target distribution that is equal to either P or Q. Thus, predicting label 0 or 1 correctly corresponds to deciding whether M was generated by P or by Q. Thereby, we obtain a lower bound for Domain Adaptation for the sum of the sizes of the labeled source sample and the unlabeled target sample, in the order of the squareroot of the domain size.

Lower bound for the Left/Right problem: We say that a (randomized) algorithm (δ, l, r, m)-solves the Left/Right problem if, given samples L, R and M of sizes l, r and m respectively, it gives the correct answer with probability at least $1-\delta$. We will now show that for any sample sizes l, r and m and for any $\gamma < 1/2$, there exists a finite domain $\mathcal{X} = \{1, 2, \ldots, n\}$ and a finite class $\mathcal{W}_n^{uni}$ of triples of distributions over $\mathcal{X}$ such that no algorithm can (γ, l, r, m)-solve the Left/Right problem for this class. In our class, both the distribution generating L and the distribution generating R are uniform over half of the points in $\mathcal{X}$, but their supports are disjoint. Formally, we construct the class as follows: $\mathcal{W}_n^{uni} = \{(U_A, U_B, U_C) : A \cup B = \{1, \ldots n\}, A \cap B = \emptyset,\ |A| = |B|, \text{ and } C = A \text{ or } C = B\}$, where, for a finite set Y, U_Y denotes the uniform distribution over Y. With this we obtain:

Lemma 1. *For any given sample sizes l for L, r for R and m for M and any $0 < \gamma < 1/2$, if $k = \max\{l, r\} + m$, then for $n > (k+1)^2/(1-2\gamma)$ no algorithm has probability of success greater than $1-\gamma$ over the class $\mathcal{W}_n^{uni}$.*

Proof. We employ a method introduced in [14] in the context of deriving a lower bound on the sample size for a related problem. The proof of this Lemma has been moved to the appendix for a more focused presentation.

Reducing the Left/Right problem to Domain Adaptation learning: In order to reduce the Left/Right problem to Domain Adaptation, we define a class of DA problems that corresponds to the class of triples $\mathcal{W}_n^{uni}$, for which we have proven a lower bound on the sample sizes needed for solving the Left/Right problem. For a number n, let $\mathcal{W}_n^{DA}$ be the class of triples (P_S, P_T, l), where P_S is uniform over some finite set $\mathcal{X}$ of size n, P_T is uniform over some subset U of $\mathcal{X}$ of size $n/2$ and l assigns points in U to 1 and points in $\mathcal{X} \setminus U$ to 0 or vice versa. Note that we have $C(P_S, P_T) = 1/2$ and $d_{H_{1,0} \Delta H_{1,0}}(P_S, P_T) = 0$ for all (P_S, P_T, l) in $\mathcal{W}_n^{DA}$. Further, for the class $H_{1,0}$ that contains only the constant 1 function and the constant 0 function, we have $\text{opt}_T^l(H_{1,0}) = 0$ for all elements of $\mathcal{W}_n^{DA}$.

Lemma 2. *The Left/Right problem reduces to Domain Adaptation. More precisely, given a number n and an algorithm $\mathcal{A}$ that, given the promise that the target task is realizable by the class $H_{1,0}$, can (ϵ, δ, s, t)-solve DA for a class $\mathcal{W}$*

that includes $\mathcal{W}_n^{DA}$, we can construct an algorithm that $(\epsilon + \delta, s, s, t+1)$*-solves the Left/Right problem on* $\mathcal{W}_n^{uni}$*.*

Proof. Assume we are given samples $L = \{l_1, l_2, \ldots, l_s\}$ and $R = \{r_1, r_2, \ldots, r_s\}$ of size s and a sample M of size $t+1$ for the Left/Right problem coming from a triple (U_A, U_B, U_C) of distributions in $\mathcal{W}_n^{uni}$. We construct an input to Domain Adaptation by setting the unlabeled target sample $T = M \setminus \{p\}$ where p is a point from M chosen uniformly at random and construct the labeled source sample S as follows: We select s elements from $L \times \{0\} \cup R \times \{1\}$ by successively flipping an unbiased coin, and depending on the output choosing the next element from $L \times \{0\}$ or $R \times \{1\}$.

These sets can now be considered as an input to Domain Adaptation generated from a source distribution $P_S = U_{A \cup B}$ that is uniform over $A \cup B$. The target distribution P_T of this Domain Adaptation instance has marginal equal to U_A or to U_B (depending on whether M was a sample from U_A or from U_B). The labeling function of this Domain Adaptation instance is $l(x) = 0$ if $x \in A$ and $l(x) = 1$ if $x \in B$. Observe that we have $C(P_S, P_T) = 1/2$, $\mathrm{opt}_T^l(H_{1,0}) = 0$, and $(P_S, P_T, l) \in \mathcal{W}_n^{DA}$. Assume that h is the output of $\mathcal{A}$ on input S and T. The algorithm for the Left/Right problem then outputs U_A if $h(p) = 0$ and U_B if $h(p) = 1$ and the claim follows as we have $\mathrm{Err}_h(P_S) \leq \epsilon$ with confidence $1 - \delta$.

Lemma 1 and Lemma 2 show that no algorithm can solve the DA problem for $\mathcal{W}_n^{DA}$, even under the assumption of realizability by $H_{1,0}$, if the sample sizes of the source and target sample satisfy $|S| + |T| < \sqrt{(1 - 2(\epsilon + \delta))|\mathcal{X}|}$. This completes the proof of Theorem 1.

4 The Use of Unlabeled Data

In many learning scenarios unlabeled data is abundantly available while labeled data is hard to obtain. Thus, it is natural to investigate, whether the amount of data that is necessary for Domain Adaptation can be covered by unlabeled target data rather than labeled source data. We start by presenting a Domain Adaptation algorithm for the case where the labeling function satisfies the Lipschitz property and the target is realizable with a margin. Note that [6] also provides a DA algorithm for labeling functions that satisfy the Lipschitz property. They show that a Nearest Neighbor algorithm solves the Domain Adaptation problem successfully, which provides an upper bound corresponding to the lower bound in Theorem 2 (without using unlabeled data). Here, we show that one can actually replace the labeled source sample by an unlabeled target sample if the learner has knowledge of a class that realizes the target distribution. Hereby, the size of the labeled sample required for success goes from the Nearest Neighbor sample complexity down to the (much smaller) sample complexity of standard learning under a realizability assumption. For this, we need the notion of an ϵ-net:

Definition 5. *Let $\mathcal{X}$ be some domain, $\mathcal{W} \subseteq 2^{\mathcal{X}}$ a collection of subsets of $\mathcal{X}$ and P a distribution over $\mathcal{X}$. An* ϵ-net *for $\mathcal{W}$ with respect to P is a subset $N \subseteq \mathcal{X}$ that intersects every member of $\mathcal{W}$ that has P-weight at least ϵ.*

We relate ϵ-nets for a source distribution to ϵ-nets for a target distribution:

Lemma 3. *Let $\mathcal{X}$ be some domain, $\mathcal{W} \subseteq 2^{\mathcal{X}}$ a collection of subsets of $\mathcal{X}$, and P_S and P_T a source and a target distribution over $\mathcal{X}$ with $C := C_{\mathcal{W}}(P_S, P_T) \geq 0$. Then every $(C\epsilon)$-net for $\mathcal{W}$ with respect to P_S is an ϵ-net for $\mathcal{W}$ w.r.t. P_T.*

Proof. Let $N \subseteq \mathcal{X}$ be an $(C\epsilon)$-net for $\mathcal{W}$ with respect to P_S. Consider a $U \in \mathcal{W}$ that has target-weight at least ϵ, *i.e.* $P_T(U) \geq \epsilon$. Then we have $P_S(U) \geq CP_T(U) \geq C\epsilon$. As N is an $(C\epsilon)$-net for $\mathcal{W}$ with respect to P_S, we have $N \cap U \neq \emptyset$.

4.1 Realizability with a Margin

We propose the following adaptive Domain Adaptation procedure:

```
Algorithm $\mathcal{A}$

Input  An i.i.d. sample $S$ from $P_S$ labeled by $l$, an unlabeled i.i.d. sample
    $T$ from $P_T$ and a margin parameter $\gamma$.
Step 1  Partition the domain $[0,1]^d$ into a collection $\mathcal{B}$ of boxes (axis-
    aligned rectangles) with sidelength $(\gamma/\sqrt{d})$.
Step 2  Obtain sample $S'$ by removing every point in $S$, which is sitting
    in a box that is not hit by $T$.
Step 3  Output an ERM classifier from $H$ for the sample $S'$.
```

The following theorem provides upper bounds on the sizes of the labeled and the unlabeled sample that suffice for algorithm $\mathcal{A}$ to succeed. Note that the complexity of the labeled sample is comparable to the size of a labeled sample required in standard learning. It depends only on the VC-dimension, the accuracy parameters and the weight ratio between source and target.

Theorem 3. *Let $\mathcal{X} = [0,1]^d$, $\gamma > 0$ a margin parameter, H be a hypothesis class of finite VC dimension and $\mathcal{W}$ be a class of triples (P_S, P_T, l) of source distribution, target distribution and labeling function with*

- *$C_{\mathcal{I}}(P_S, P_T) > 0$ for the class $\mathcal{I} = (H \Delta H) \sqcap \mathcal{B}$, where $\mathcal{B}$ is a partition of $[0,1]^d$ into boxes of sidelength $\gamma/\sqrt{d}$*
- *P_T is realizable by H with margin γ*
- *the labeling function l is a γ-margin classifier with respect to P_T.*

Then there is a constant $c > 1$ such that, for all $\epsilon > 0$, $\delta > 0$, and all $(P_S, P_T, l) \in \mathcal{W}$, when given an i.i.d. sample S from P_S, labeled by l of size

$$|S| \geq c\left(\frac{\mathrm{VC}(H) + \log(1/\delta)}{C_{\mathcal{I}}(P_S, P_T)(1-\epsilon)\epsilon}\log\left(\frac{\mathrm{VC}(H)}{C_{\mathcal{I}}(P_S, P_T)(1-\epsilon)\epsilon}\right)\right)$$

and an i.i.d. sample T from P_T of size $|T| \geq \frac{2(\sqrt{d}/\gamma)^d \ln(3(\sqrt{d}/\gamma)^d/\delta)}{\epsilon}$ then algorithm $\mathcal{A}$ outputs a classifier h with $\mathrm{Err}^l_T(h) \leq \epsilon$ with probability at least $1 - \delta$.

Proof. Let $\epsilon > 0$ and $\delta > 0$ be given and set $C = C_{\mathcal{I}}(P_S, P_T)$. We set $\epsilon' = \epsilon/2$ and $\delta' = \delta/3$ and divide the space $\mathcal{X}$ up into *heavy* and *light* boxes from $\mathcal{B}$, by defining a box $b \in \mathcal{B}$ to be light if $P_T(b) \leq \epsilon'/|\mathcal{B}| = \epsilon'/(\sqrt{d}/\gamma)^d$ and heavy otherwise. We let $\mathcal{X}^l$ denote the union of the light boxes and $\mathcal{X}^h$ the union of the heavy boxes. Further, we let P_S^h and P_T^h denote the restrictions of the source and target distributions to $\mathcal{X}^h$, *i.e.* we have $P_S^h(U) = P_S(U)/P_S(\mathcal{X}^h)$ and $P_T^h(U) = P_T(U)/P_T(\mathcal{X}^h)$ for all $U \subseteq \mathcal{X}^h$ and $P_S^h(U) = P_T^h(U) = 0$ for all $U \nsubseteq \mathcal{X}^h$. As $|\mathcal{B}| = (\sqrt{d}/\gamma)^d$, we have $P_T(\mathcal{X}^h) \geq 1 - \epsilon'$ and thus, $P_S(\mathcal{X}^h) \geq C(1 - \epsilon')$.

We will show that

Claim 1. With probability at least $1 - \delta'$ an *i.i.d.* P_T-sample T of size as stated in the Theorem hits every heavy box.

Claim 2. With probability at least $1 - 2\delta'$ the intersection of S and $\mathcal{X}^h$, where S is an *i.i.d.* P_S-sample of size as stated in the Theorem is an ϵ'-net for $H\Delta H$ with respect to P_T^h.

To see that these imply the claim of the theorem, let $S^h = S \cap \mathcal{X}^h$ denote the intersection of the source sample and the union of heavy boxes. By Claim 1, T hits every heavy box with high probability, thus $S^h \subseteq S'$, where S' is the intersection of S with boxes that are hit by T (see the description of the algorithm $\mathcal{A}$). Therefore, if S^h is an ϵ'-net for $H\Delta H$ with respect to P_T^h (as guaranteed by Claim 2) then so is S'. Hence, with probability at least $1 - 3\delta' = 1 - \delta$ the set S' is an ϵ'-net for $H\Delta H$ with respect to P_T^h. Now note that an ϵ'-net for $H\Delta H$ with respect to P_T^h is an ϵ-net with respect to P_T as every set of P_T-weight at least ϵ has P_T^h weight at least ϵ', by definition of $\mathcal{X}^h$ and P_T^h.

Finally, we need to show that S' being an ϵ-net for the set $H\Delta H$ of symmetric differences with respect to the target class, suffices for the ERM-classifier from the target class to have target error at most ϵ. Let $h_T^* \in H$ denote the γ-margin classifier of zero target error. Note that every box in $\mathcal{B}$ is labeled homogeneously with label 1 or label 0 by the labeling function l as l is a γ-margin classifier as well. Let $s \in S'$ be a sample point and $b_s \in \mathcal{B}$ be the box that contains s. As h_T^* is a γ-margin classifier and $P_T(b_s) > 0$ (b_s was hit by T by the definition of S'), b_s is labeled homogeneously by h_T^* as well and as h_T^* has zero target error this label has to correspond to the labeling by l. Thus $h_T^*(s) = l(s)$ for all $s \in S'$, which means that the empirical error with respect to S' of h_T^* is zero.

Now consider a classifier h_ϵ with $\mathrm{Err}_T^l(h_\epsilon) \geq \epsilon$. Let $s \in S'$ be a sample point in $h_T^* \Delta h_\epsilon$ (which exists as S' is an ϵ-net). As $s \in h_T^* \Delta h_\epsilon$, we have $h_\epsilon(s) \neq h_T^*(s) = l(s)$ and thus, h_ϵ as an empirical error larger than zero, which implies that no classifier of error larger than ϵ can be chosen by ERM on input S'.

Proof of Claim 1: Let b be a heavy box, thus $P_T(b) \geq \epsilon'/|\mathcal{B}|$. Then, when drawing an *i.i.d.* sample T from P_T, the probability of not hitting b is at most $(1 - (\epsilon'/|\mathcal{B}|))^{|T|}$. Now the union bound implies that the probability that there is at a box in $\mathcal{B}^h$ that does not get hit by the sample T is bounded by

$$|\mathcal{B}^h|(1 - (\epsilon'/|\mathcal{B}|))^{|T|} \leq |\mathcal{B}|(1 - (\epsilon'/|\mathcal{B}|))^{|T|} \leq |\mathcal{B}|\mathrm{e}^{-\epsilon'|T|/|\mathcal{B}|}.$$

Thus if $|T| \geq \frac{|\mathcal{B}|\ln(|\mathcal{B}|/\delta')}{\epsilon'} = \frac{2(\sqrt{d}/\gamma)^d \ln(3(\sqrt{d}/\gamma)^d/\delta)}{\epsilon}$ the sample T hits every heavy box with probability at least $1 - \delta'$.

Proof of Claim 2: Let $S^h := S \cap \mathcal{X}^h$. Note that, as S is an *i.i.d.* P_S sample, we can consider S^h to be an *i.i.d.* P_S^h sample. We have the following bound on the weight ratio between P_S^h and P_T^h:

$$\begin{aligned} C_{\mathcal{I}}(P_S^h, P_T^h) &= \inf_{p \in \mathcal{I}, P_T^h(p) > 0} \frac{P_S^h(p)}{P_T^h(p)} = \inf_{p \in \mathcal{I}, P_T^h(p) > 0} \frac{P_S(p)}{P_T(p)} \frac{P_T(\mathcal{X}^h)}{P_S(\mathcal{X}^h)} \\ &\geq C \frac{P_T(\mathcal{X}^h)}{P_S(\mathcal{X}^h)} \geq C(1-\epsilon'), \end{aligned}$$

where the last inequality holds as $P_T(\mathcal{X}^h) \geq (1-\epsilon')$ and $P_S(\mathcal{X}^h) \leq 1$. Note that every element in $H \Delta H$ can be partitioned in to elements from $\mathcal{I}$, therefore we obtain the same bound on the weight ratio for the symmetric differences of H: $C_{H\Delta H}(P_S^h, P_T^h) \geq C(1-\epsilon')$.

It is well known that there is a constant $c > 1$ such that, conditioned on S^h having size at least $M := c\left(\frac{\mathrm{VC}(H\Delta H)+\log(1/\delta')}{C(1-\epsilon')\epsilon'} \log(\frac{\mathrm{VC}(H\Delta H)}{C(1-\epsilon')\epsilon'})\right)$, with probability at least $1-\delta'$ it is a $C(1-\epsilon')\epsilon'$-net with respect to P_S^h and thus an ϵ'-net with respect to P_T^h by Lemma 3 (see, e.g. Corollary 3.8 in [15]).

Thus, it remains to show that with probability at least $1-\delta'$ we have $|S^h| \geq M$. As we have $P_S(\mathcal{X}^h) \geq C(1-\epsilon')$, we can view the sampling of the points of S and checking whether they hit $\mathcal{X}^h$ as a Bernoulli variable with mean $\mu = P_S(\mathcal{X}^h) \geq C(1-\epsilon')$. Thus, by Hoeffding's inequality we have that for all $t > 0$ $\Pr(\mu|S| - |S^h| \geq t|S|) \leq \mathrm{e}^{-2t^2|S|}$. If we set $C' = C(1-\epsilon')$, assume $|S| \geq \frac{2M}{C'}$ and set $t = C'/2$, we obtain $\Pr(|S^h| < M) \leq \Pr(\mu|S| - |S^h| \geq \frac{C'}{2}|S|) \leq \mathrm{e}^{-\frac{C'^2|S|}{2}}$.

Now $|S| \geq \frac{2M}{C'} > \frac{2(\mathrm{VC}(H\Delta H)+\log(1/\delta'))}{C^2(1-\epsilon')^2\epsilon'}$ implies that $\mathrm{e}^{-\frac{C'^2|S|}{2}} \leq \delta'$, thus we have shown that S^h is an ϵ'-net of $H\Delta H$ with probability at least $(1-\delta')^2 \geq 1-2\delta'$.

Imitating the proof of Claim 1 in [16] one can show that $\mathrm{VC}(H\Delta H) \leq 2\mathrm{VC}(H)+1$. This completes the proof.

4.2 Finite Domain

The procedure $\mathcal{A}$ from the previous section can be modified to work on any finite domain with arbitrary labeling functions and hypothesis classes of finite VC-dimension (under the target-realizability assumption). For the modification, we delete Step 2 and instead of Step 3 the algorithm removes every point from the labeled source sample S which is not hit by the unlabeled target sample T. This does not change the size of the source sample S needed for a guarantee of success, but the size of the target sample now depends on the size of the domain instead of the labeling function's Lipschitzness. The proof of the following result is a simple modification of the proof of Theorem 3 and is left to the reader.

Theorem 4. *Let $\mathcal{X}$ be some domain, H be a hypothesis class of finite VC dimension and $\mathcal{W} = \{(P_S, P_T, l) \mid C(P_S, P_T) > 0, \mathrm{opt}_T^l(H) = 0\}$ be a class of pairs of source and target distributions with bounded weight ratio where the target is realizable by H. Then there is a constant $c > 1$ such that, for all $\epsilon > 0$, $\delta > 0$, and all $(P_S, P_T, l) \in \mathcal{W}$, when given an i.i.d. sample S from P_S, labeled by l of size $|S| \geq c\left(\frac{\mathrm{VC}(H)+\log(1/\delta)}{C(P_S,P_T)^2(1-\epsilon)^2\epsilon} \log(\frac{\mathrm{VC}(H)}{C(P_S,P_T)^2(1-\epsilon)^2\epsilon})\right)$ and an i.i.d. sample*

T from P_T of size $|T| \geq \frac{2|\mathcal{X}|\ln(3|\mathcal{X}|/\delta)}{\epsilon}$ then algorithm $\mathcal{A}$ outputs a classifier h with $\mathrm{Err}^l_T(h) \leq \epsilon$ with probability at least $1 - \delta$.

Acknowledgements. We thank Shai Shalev-Shwartz for insightful discussions on this work and Shalev Ben-David for deriving the bound of Lemma 5.

References

[1] Mansour, Y., Mohri, M., Rostamizadeh, A.: Domain adaptation: Learning bounds and algorithms. In: COLT (2009)

[2] Cortes, C., Mansour, Y., Mohri, M.: Learning bounds for importance weighting. In: Lafferty, J., Williams, C.K.I., Shawe-Taylor, J., Zemel, R., Culotta, A. (eds.) Advances in Neural Information Processing Systems 23, pp. 442–450 (2010)

[3] Sugiyama, M., Krauledat, M., Müller, K.R.: Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research 8, 985–1005 (2007)

[4] Tsuboi, Y., Kashima, H., Hido, S., Bickel, S., Sugiyama, M.: Direct density ratio estimation for large-scale covariate shift adaptation. Journal of Information Processing 17, 138–155 (2009)

[5] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Machine Learning 79(1-2), 151–175 (2010)

[6] Ben-David, S., Shalev-Shwartz, S., Urner, R.: Domain adaptation–can quantity compensate for quality? In: ISAIM (2012)

[7] Huang, J., Gretton, A., Schölkopf, B., Smola, A.J., Borgwardt, K.M.: Correcting sample selection bias by unlabeled data. In: NIPS. MIT Press (2007)

[8] Sugiyama, M., Müller, K.: Generalization error estimation under covariate shift. In: Workshop on Information-Based Induction Sciences (2005)

[9] Cortes, C., Mohri, M., Riley, M., Rostamizadeh, A.: Sample Selection Bias Correction Theory. In: Freund, Y., Györfi, L., Turán, G., Zeugmann, T. (eds.) ALT 2008. LNCS (LNAI), vol. 5254, pp. 38–53. Springer, Heidelberg (2008)

[10] Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: VLDB, pp. 180–191 (2004)

[11] Cortes, C., Mohri, M.: Domain Adaptation in Regression. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 308–323. Springer, Heidelberg (2011)

[12] Ben-David, S., Lu, T., Luu, T., Pál, D.: Impossibility theorems for domain adaptation. In: AISTATS, vol. 9, pp. 129–136 (2010)

[13] Kelly, B.G., Tularak, T., Wagner, A.B., Viswanath, P.: Universal hypothesis testing in the learning-limited regime. In: IEEE International Symposium on Information Theory (ISIT) (2010)

[14] Batu, T., Fortnow, L., Rubinfeld, R., Smith, W.D., White, P.: Testing closeness of discrete distributions. CoRR abs/1009.5397 (2010)

[15] Haussler, D., Welzl, E.: Epsilon-nets and simplex range queries. In: Proceedings of the Second Annual Symposium on Computational Geometry, SCG 1986, pp. 61–71. ACM, New York (1986)

[16] Ben-David, S., Litman, A.: Combinatorial variability of vapnik-chervonenkis classes with applications to sample compression schemes. Discrete Applied Mathematics 86(1), 3–25 (1998)

[17] Ben-David, S.: Private communication (2011)

A Full Proof of Lemma 1

We employ a method introduced in [14] in the context of deriving a lower bound on the sample size for a related problem. The authors show that, when testing so-called permutation invariant properties, *i.e.* if the property does not change with permuting the underlying domain, it suffices to consider algorithms that take only a *fingerprint* of the sample as input (see precise Definition below). Note that the Left/Right problem is permutation-invariant, since, whether M is a sample from P or from Q does not depend on a permutation of $\mathcal{X}$.

Definition 6. *Let L, R, M be three multi-sets of sizes at most n each sampled from distributions P or Q over some domain $\mathcal{X}$ as in the definition of the Left/Right problem. We define the* fingerprint *of this triple of multi sets as the set $\{C_{i,j,k} \mid 1 \leq i, j, k \leq n\}$ where $C_{i,j,k}$ is the number of elements of $\mathcal{X}$, that appear exactly i times in L, j times in R and k times in M.*

The following lemma allows us to restrict our attention to fingerprints of an instance of the Left/Right problem as input.

Lemma 4 (Batu et al., [14]). *If there exists an algorithm $\mathcal{A}$ for testing some permutation-invariant property of distributions, then there exists an algorithm for that same task that gets as input only the fingerprints of the samples that $\mathcal{A}$ takes and enjoys the same guarantee on its probability of success.*

Proof (Proof sketch). This lemma is proven by showing how to reconstruct the samples from a fingerprint for some fixed permutation of the distribution. To see this, note that each element of $\mathcal{X}$ contributes to at most one of the $C_{i,j,k}$. Thus, an algorithm can reconstruct a permuted sample from the fingerprint and then feed this sample as input to $\mathcal{A}$. As the property is permutation-invariant, this can not change the (distribution over the) output(s).

The following lemma gives a lower bound on the sample size needed to see repetitions in a sample from a uniform distribution over a finite domain.

Lemma 5 (Shalev Ben-David [17]). *Let $\mathcal{X}$ be a finite domain of size n. For every $0 < \delta < 1$, with probability acceding $(1-\delta)$, an i.i.d. sample of size at most $\sqrt{\delta n} - \delta$ uniformly drawn over $\mathcal{X}$, contains no repeated elements.*

Proof of Lemma 1: Set $\delta = 1 - 2\gamma$. By Lemma 5, with probability exceeding $(1-\delta)$ the input to the Left/Right problem over $\mathcal{W}_n^{uni}$ has no repeated elements and the three input samples are disjoint. Consequently, with probability exceeding $(1-\delta)$, the fingerprint F of the input has $C_{1,0,0} = l$, $C_{0,1,0} = r$, $C_{0,0,1} = m$ and $C_{i,j,k} = 0$ for all other combinations of i, j and k independently of whether the sample M was generated by U_A or by U_B.

Let $\mathcal{A}$ be some algorithm. Let $p \in [0,1]$ be the probability that $\mathcal{A}$ outputs U_A on input F. Now, if $p \leq 1/2$ we have that $\mathcal{A}$ errs with probability larger than $(1-\delta)/2 = \gamma$ for all triples where C is equal to B. Otherwise it errs with probability larger than $(1-\delta)/2 = \gamma$ on all triples where C is equal to A. Thus, no algorithm can (γ, l, r, m)-solve the Left/Right problem for the class $\mathcal{W}_n^{uni}$.

Efficient Protocols for Distributed Classification and Optimization

Hal Daumé III[1], Jeff M. Phillips[2], Avishek Saha[2], and Suresh Venkatasubramanian[2]

[1] University of Maryland, CP, MD 20742, USA
hal@umiacs.umd.edu
[2] University of Utah, SLC, UT 84112, USA
{jeffp,avishek,suresh}@cs.utah.edu

Abstract. A recent paper [1] proposes a general model for distributed learning that bounds the communication required for learning classifiers with ε error on linearly separable data adversarially distributed across nodes. In this work, we develop key improvements and extensions to this basic model. Our first result is a two-party *multiplicative-weight-update* based protocol that uses $O(d^2 \log 1/\varepsilon)$ words of communication to classify distributed data in arbitrary dimension d, ε-optimally. This extends to classification over k nodes with $O(kd^2 \log 1/\varepsilon)$ words of communication. Our proposed protocol is simple to implement and is considerably more efficient than baselines compared, as demonstrated by our empirical results.

In addition, we show how to solve fixed-dimensional and high-dimensional linear programming with small communication in a distributed setting where constraints may be distributed across nodes. Our techniques make use of a novel connection from multipass streaming, as well as adapting the multiplicative-weight-update framework more generally to a distributed setting.

1 Introduction

In recent years, distributed learning (learning from data spread across multiple locations) has witnessed a lot of research interest [2]. One of the major challenges in distributed learning is to minimize communication overhead between different parties, each possessing a disjoint subset of the data. Recent work [1] has proposed a distributed learning model that seeks to minimize communication (in a series of rounds) by carefully choosing the most informative data points at each node in each round. The authors present a number of general sampling based results as well as a specific two-way protocol that provides a logarithmic error bound on communication for the family of linear classifiers in $\mathbb{R}^2$. Most of their results pertain to two players but they propose basic (and as we will see, inefficient) extensions for multi-player scenarios. A distinguishing feature of this model is that it is *adversarial*. Except linear separability, no distributional or other assumptions are made on the data or how it is distributed across nodes.

In this paper, we develop this model substantially with new algorithmic ideas for solving learning problems. First, we extend the results on linear classification to arbitrary dimensions, in the process presenting a more general algorithm that does not rely on explicit geometric constructions. This approach exploits the multiplicative weight

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 154–168, 2012.

update (MWU) framework (specifically its use in boosting) and retains desirable theoretical guarantees – *data-size-independent* communication between nodes in order to classify data – while being simple to implement. Moreover, it easily extends to k-players, scaling only linearly in k, improving earlier results in two dimensions by a factor of k.

Motivated by the insight that MWU can be used to solve distributed learning problems, we propose approximate solutions for distributed semidefinite programming. In addition, we show how a *generic* multipass streaming algorithm for a problem can be made distributed, and apply this framework to solving linear programming in a distributed setting both exactly and approximately. Together, these results indicate that general optimization problems can be solved efficiently in our model. Exploiting the strong link between learning and optimization will then open the door to deploying many other learning tasks in the distributed setting with minimal communication.

Related Work. Existing work in distributed learning mainly focuses on either inferring an accurate global classifier from multiple distributed sub-classifiers learned individually (at respective nodes) or on improving the efficiency of the overall learning protocol. The first line of work consists of techniques like *parameter mixing* [3, 4] or *averaging* [5] and classifier *voting* [6]. These approaches do admit convergence results but lack any useful bounds on the communication. Voting, on the other hand, has been shown [1] to yield suboptimal results on adversarially partitioned datasets. The goal of the second line of work is to make distributed algorithms scale to large datasets [7]; many of these works [8, 9] focus on MapReduce. [10] proposed a MapReduce based improved parallel stochastic gradient descent and more recently [11] improved the time complexity of γ-margin parallel algorithms from $\Omega(1/\gamma^2)$ to $O(1/\gamma)$.

Surprisingly absent in the above lines of work is the direct study of how to use communication sparingly in learning. And as [1] and this work demonstrates, intelligent interaction between nodes, communicating key data subsets not just its classification, can greatly reduce the necessary communication over existing approaches. On large distributed systems, communication has become a major bottleneck for many real-world problems; it accounts for a large percentage of total energy costs, and is the main reason that MapReduce algorithms are designed to minimize rounds (of communication). This strongly motivates the need to incorporate the study of this aspect of an algorithm directly, as presented and modeled in this paper.

Independently of this work[1], research by [12] considers very similar models to those of [1]. They also consider adversarially distributed data among k parties and provide algorithms to learn while minimizing the total communication between the parties. Like [1] the work of [12] presents both agnostic and non-agnostic results for generic settings, and shows improvements over sampling bounds in several specific settings including the d-dimensional linear classifier problem we consider here (also drawing inspiration from boosting). In addition, their work provides total communication bounds for decision lists and for proper and non-proper learning of parity functions. They also extend the model so as to preserve differential and distributional privacy while conserving total communication, as a resource, during the learning process.

[1] Preliminary versions of [12] and this work [13] were coordinated to be placed on the arXiv on the same day.

In contrast, this work identifies optimization as a key primitive underlying many learning tasks, and focuses on solving the underlying optimization problems as a way to provide general communication-friendly distributed learning methods. We introduce techniques that rely on multiplicative weight updates and multi-pass streaming algorithms. Our main contributions include translating these techniques into this distributed setting and using them to solve LPs (and SDPs) in addition to solving for d-dimensional linear separators.

2 Background

Here we revisit the basic model [1].

Model. We assume that there are k parties $P_1, P_2, \dots P_k$. Each party P_i possesses a dataset D_i that no other party has access to, and each D_i may have both positive and negative examples. The goal is to classify the full dataset $D = \cup_i D_i$ correctly. We assume that there exists a perfect classifier h^* from a family of classifiers $\mathcal{H}$ with associated range space $(D, \mathcal{H})$ and bounded VC-dimension v. We are willing to allow ε-classification error on D so that up to $\varepsilon|D|$ points in total are misclassified.

Each *word* of data (e.g., a single point or vector in $\mathbb{R}^d$ counts as $O(d)$ words) passed between any pair of parties is counted towards the total communication; this measure in words allows us to examine the cost of extending to d-dimensions, and allows us to consider communication in forms other than example points, but does not hinder us with precision issues required when counting bits. For instance, a protocol that broadcasts a message of M words (say M/d points in $\mathbb{R}^d$) from one node to the other $k-1$ players costs $O(kM)$ communication. The goal is to design a protocol with as little communication as possible. We assume an *adversarial* model of data distribution; in this setting we prepare for the worst, and allow some *adversary* to determine which player gets which subset of D.

Sampling Bounds. Given D and a family of classifiers with bounded VC-dimension v, a random sample from D of size

$$s_{\varepsilon,v} = O(\min\{(v/\varepsilon)\log(v/\varepsilon), v/\varepsilon^2\}) \tag{1}$$

has at most ε-classification error on D with constant probability [14], as long as there exists a perfect classifier. Throughout this paper we will assume that a perfect classifier exists. This constant probability of success can be amplified to $1-\delta$ with an extra $O(\log(1/\delta))$ factor of samples.

Randomly Partitioned Distributions. Assume that for all $i \in [1,k]$, each party P_i has a dataset D_i drawn from the same distribution. That is, all datasets D_i are identically distributed. This case is much simpler than what the remainder of this paper will consider. Using (1), each D_i can be viewed as a sample from the full set $D = \cup_i D_i$, and with *no* communication each party P_i can faithfully estimate a classifier with error $O((v/|D_i|)\log(v|D_i|))$ [1].

Henceforth we will focus on *adversarially* distributed data.

One-Way Protocols. Consider a restricted setting where protocols are only able to send data from parties P_i (for $i \geq 2$) to P_1; a restricted form of *one-way communication*. We can again use (1) so that all parties P_i send a sample S_i of size $s_{\varepsilon,\nu}$ to P_1, and then P_1 constructs a global classifier on $\cup_{i=2}^{k} S_i$ with ε-classification error $\cup_{i=1}^{k} D_i$; this requires $O(dks_{\varepsilon,\nu})$ words of communication for points in $\mathbb{R}^d$.

For specific classifiers [1] we can do better. For thresholds and intervals one can learn a *zero*-error distributed classifier using constant amount of one-way communication. The same can be achieved for axis-aligned rectangles with $O(kd^2)$ words of communication. However, those authors show that hyperplanes in $\mathbb{R}^d$, for $d \geq 2$, require at least $\Omega(k/\varepsilon)$ one-way bits of communication to learn an ε-error distributed classifier.

Two-Way Protocols. Hereafter, we consider two-way protocols where any two players can communicate back and forth. It has been shown [1] that, in $\mathbb{R}^2$, a protocol can learn linear classifiers with at most ε-classification error using at most $O(k^2 \log 1/\varepsilon)$ communication. This protocol is deterministic and relies on a complicated pruning argument, whereby in each round, either an acceptable classifier is found, or a constant fraction more of some party's data is ensured to be classified correctly.

3 Improved Random Sampling for *k*-Players

Our first contribution is an improved two-way k-player sampling-based protocol using *two-way* communication and the sampling result in (1). We designate party P_1 as a coordinator. P_1 gathers the size of each player's dataset D_i, simulates sampling from each player completely at random, and then reports back to each player the number of samples to be drawn by it, in $O(k)$ communication. Then each other party P_i selects $s_{\varepsilon,\nu}|D_i|/|D|$ random points (in expectation), and sends them to the coordinator. The union of this set satisfies the conditions of the result from (1) over $D = \cup_i D_i$ and yields the following result.

Theorem 1. *For any hypothesis family with VC-dimension ν for points in $\mathbb{R}^d$, there exists a two-way k-player protocol using $O(kd + d\min\{(\nu/\varepsilon)\log(\nu/\varepsilon), \nu/\varepsilon^2\})$ total words of communication that achieves ε-classification error, with constant probability.*

Using two-way communication, this type of result can be made even more general. Consider the case where each P_i's dataset arrives in a continuous stream; this is known as a *distributed data stream* [15]. Then applying results of [16], we can continually maintain a sufficient random sample at the coordinator of size s_ε (using an generalization of reservoir sampling) communicating $O((k + s_{\varepsilon,\nu})d\log|D|)$ words.

Theorem 2. *Let each of k parties have a stream of data points D_i where $D = \cup_i D_i$. For any hypothesis family with VC-dimension ν for points in $\mathbb{R}^d$, there exists a two-way k-player protocol using $O((k + \min\{(\nu/\varepsilon)\log(\nu/\varepsilon), \nu/\varepsilon^2\})\ d\log|D|)$ total words of communication that maintains ε-classification error, with constant probability.*

4 A Two-Party Protocol

In this section, we consider only two parties and refer to them as A and B. $A's$ data is labeled D_A and B's data is labeled D_B ($|D_B| = n$). Our protocol, summarized in Algorithm 1, is called WEIGHTEDSAMPLING. In each round, A sends a classifier h_A to B

and B responds back with a set of points R_B, constructed by sampling from a weighting on its points. After T rounds (for $T = O(\log(1/\varepsilon))$), we will show that by voting on the result from the set of T classifiers h_A will misclassify at most $\varepsilon|D_B|$ points from D_B while being perfect on D_A, and hence $\varepsilon|D_B| < \varepsilon|D_B \cup D_A| = \varepsilon|D|$, yielding a ε-optimal classifier as desired.

Algorithm 1. WeightedSampling

```
Input: $D_A, D_B$, parameters: $0 < \varepsilon < 1$
Output: $h_{AB}$ (classifier with $\varepsilon$-error on $D_A \cup D_B$)
Init: $R_B = \{\}$; $w_i^0 = 1\ \forall x_i \in D_B$;
for t = 1 ... $T = 5\log_2(1/\varepsilon)$ do
    ——— A's move ———
    $D_A = D_A \cup R_B$; $h_A^t := Learn(D_A)$; send $h_A^t$ to $B$;
    ——— B's move ———
    $R_B$ := Mwu ($D_B$, $h_A^t$, 0.75, 0.2); send $R_B$ to $A$;
end for
$h_{AB} = \mathsf{Majority}(h_A^1, h_A^2, \ldots, h_A^T)$;
```

R_B can construct its points in two ways: a random sample and a deterministic sample. We will focus on the randomized version since it is more practical, although it has slightly worse bounds in the two-party case. Then we will also mention and analyze the deterministic version.

It remains to describe how B's points are weighted and updated, which dictates how B constructs the sample sent to A. Initially, they are all given a weight $w_1 = 1$. Then the re-weighting strategy (described in Algorithm 2) is an instance of the multiplicative weight update framework; with each new classifier h_A from A, party B increases all weights of misclassified points by a $(1+\rho)$ factor, and does not change the weight for correctly classified points. We will show $\rho = 0.75$ is sufficient. Intuitively, this ensures that consistently misclassified points eventually get weighted high enough that they are very likely to be chosen as examples to be communicated in future rounds. The deterministic variant simply replaces Line 7 of Algorithm 2 with the weighted variant [17] of the deterministic construction of R_B [18]; see details below.

Note that this is roughly similar in spirit to the heuristic protocol [1] that exchanged support points and was called IterativeSupports, which we will experimentally compare against. But the protocol proposed here is less rigid, and as we will demonstrate next, this allows for a much less nuanced analysis.

4.1 Analysis

Our analysis is based on the multiplicative weight update framework (and closely resembles boosting). First, we state a key structural lemma. Thereafter, we use this lemma for our main result. For ease of readability, we defer all proofs to the appendix.

As mentioned above (see (1)), after collecting a random sample S_ε of size $s_{\varepsilon,d} = O(\min\{(d/\varepsilon)\log(d/\varepsilon), d/\varepsilon^2\})$ drawn over the entire dataset $D \subset \mathbb{R}^d$, a linear classifier learned on S_ε is sufficient to provide ε-classification error on all of D with constant probability. There exist deterministic constructions for these samples S_ε still of size $s_{\varepsilon,\nu}$ [18] (and sometimes slightly smaller [19]); although they provide at most ε-classification error with probability 1, they, in general, run in time exponential in ν. Note that the VC-dimension of linear classifiers in $\mathbb{R}^d$ is $O(d)$, and these results still holds when the points are weighted and the sample is drawn (respectively constructed [17]) and error measured with respect to this weighting distribution. Thus B could send $s_{\varepsilon,d}$ points to A, and we would be done; but this is too expensive. We restate this result with a constant c, so that at most a c fraction of the weights of points are mis-classified (later we show that $c = 0.2$ is sufficient with our framework). Specifically, setting $\varepsilon = c$ and rephrasing the above results yields the following lemma.

Algorithm 2. MWU (D_B, h_A^t, ρ, c)

```
Input: $h_A^t, D_B$, parameters: $0 < \rho < 1, 0 < c < 1$
Output: $R_B$ (a set of $s_{c,d}$ points)
for all ($x_i \in D_B$) do
    if($h_A^t(x_i) \neq y_i$) then $w_i^{t+1} = w_i^t(1+\rho)$;
    if($h_A^t(x_i) == y_i$) then $w_i^{t+1} = w_i^t$;
end for
randomly sample $R_B$ from $D_B$ (according to $w^{t+1}$);
```

Lemma 1. *Let B have a weighted set of points D_B with weight function $w : D_B \to \mathbb{R}^+$. For any constant $c > 0$, party B can send a set $S_{c,d}$ of size $O(d)$ (where the constant depends on c) such that any linear classifier that correctly classifies all points in $S_{c,d}$ will misclassify points in D_B with a total weight at most $c\sum_{x \in D_B} w(x)$. The set $S_{c,d}$ can be constructed deterministically, or a weighted random sample from (D_B, w) succeeds with constant probability.*

We first state the bound using the deterministic construction of the set $S_{c,d}$, and then extend it to the more practical (from an implementation perspective) random sampling result, but with a slightly worse communication bound.

Theorem 3. *The deterministic version of two-party two-way* WEIGHTEDSAMPLING *for linear separators in $\mathbb{R}^d$ misclassifies at most $\varepsilon|D|$ points after $T = O(\log(1/\varepsilon))$ rounds using $O(d^2 \log(1/\varepsilon))$ words of communication.*

In order to use random sampling, as suggested in Algorithm 2, we need to address the probability of failure of our protocol. More specifically, the set $S_{c,d}$ in Lemma 1 is of size $O(d \log(1/\delta'))$ and a linear classifier with no error on $S_{c,d}$ misclassifies points in D_B with weight at most $c\sum_{x \in D_B} w(x)$, with probability at least $1 - \delta'$. We want this probability of failure to be a constant δ over the entire course of the protocol. Setting $\delta' = \delta/T$, and applying the union bound implies that the probability of failure at any point in the protocol is at most $\sum_{i=1}^T \delta' = \sum_{i=1}^T \delta/T = \delta$. This increases the communication cost of each round to $O(d^2 \log(1/\delta')) = O(d^2 \log(\log(1/\varepsilon)/\delta)) = O(d^2 \log\log(1/\varepsilon))$ words, with a constant δ probability of failure. Thus, random sampling in WEIGHTEDSAMPLING requires a total of $O(d^2 \log(1/\varepsilon) \log\log(1/\varepsilon))$ words of communication. We formalize below.

Theorem 4. *The randomized two-party two-way protocol* WEIGHTEDSAMPLING *for linear separators in* $\mathbb{R}^d$ *misclassifies at most* $\varepsilon|D|$ *points, with constant probability, after* $T = O(\log(1/\varepsilon))$ *rounds using* $O(d^2 \log(1/\varepsilon) \log\log(1/\varepsilon))$ *words of communication.*

5 *k*-Party Protocol

In Section 3 we described a simple protocol (Theorem 1) to learn a classifier with ε-error jointly among k parties using $O(kd + d\min\{\nu/\varepsilon \log(\nu/\varepsilon), \nu/\varepsilon^2\})$ words of total communication. We now combine this with the two-party protocol from Section 4 to obtain a k-player protocol for learning a joint classifier with error ε.

We fix an arbitrary node (say P_1) as the coordinator for the k-player protocol of Theorem 1. Then P_1 runs a version of the two-player protocol (from Section 4) from A's perspective and where players $P_2, \dots, P_k$ serve jointly as the second player B. To do so, we follow the distributed sampling approach outlined in Theorem 1. Specifically, we fix a parameter c (set $c = 0.2$). Each other node reports the total weight $w(D_i)$ of their data to P_1, who then reports back to each node what fraction of the total data $w(D_i)/w(D)$ they own. Then each player sends the coordinator a random sample of size $s_{c,d}w(D_i)/w(D)$. Recall that we require $s_{c,d} = O(d \log\log(1/\varepsilon))$ in this case to account for probability of failure over all rounds. The union of these sets at P_1 satisfies the sampling condition in Lemma 1 for $\cup_{i=2}^k D_i$. P_1 computes a classifier on the union of its data and this joint sample and all previous joint samples, and sends the resulting classifier back to all the nodes. Sending this classifier to each party requires $O(kd)$ words of communication. The process repeats for $T = \log_2(1/\varepsilon)$ rounds.

Theorem 5. *The randomized k-party protocol for* ε*-error linear separators in* $\mathbb{R}^d$ *terminates in* $T = O(\log(1/\varepsilon))$ *rounds using* $O((kd + d^2 \log\log(1/\varepsilon)) \log(1/\varepsilon))$ *words of communication, and has a constant probability of failure.*

The random sampling algorithm required a sample of size $O(d \log\log(1/\varepsilon))$. However we can achieve a different communication trade-off using the deterministic construction where, in each round, each party P_i communicates a deterministically constructed set $S_{c,i}$ of size $O(d)$. The coordinator P_1 computes a classifier that correctly classifies points from all of these sets having at most $cw(D_i)$ weight of points misclassified in each D_i. The error is at most $cw(D_i)$ on each dataset D_i and so the error on all sets is at most $c\sum_{i=2}^k w(D_i) = cw(D)$. Again using $T = O(\log(1/\varepsilon))$ rounds we can achieve the following result.

Theorem 6. *The deterministic k-party protocol for* ε*-error linear separators in* $\mathbb{R}^d$ *terminates in* $T = O(\log(1/\varepsilon))$ *rounds using* $O(kd^2 \log(1/\varepsilon))$ *words of communication.*

6 Experiments

In this section, we compare WEIGHTEDSAMPLING with the following baselines for 2-party and k-party protocols.

- NAIVE: sends all data from $(k-1)$ nodes to a coordinator node and then learns at the coordinator.

- VOTING: trains classifiers at each individual node and sends over the $(k-1)$ classifiers to a coordinator node. For any datapoint, the coordinator node predicts the label by taking a vote over all k classifiers.
- RAND: each of the $(k-1)$ nodes sends a random sample of size $s_{\varepsilon,d}$ to a coordinator node and then a classifier is learned at the coordinator node using all of its own data and the samples received.
- RANDEMP: cheaper version of RAND that uses a random sample of size $9d$ from each party each round; this value was chosen to make this baseline technique as favorable as possible.
- MAXMARG: ITERATIVESUPPORTS that selects informative points heuristically [1]. We do not compare with MEDIAN [1] as it is not applicable beyond two dimensions.
- MWU: WEIGHTEDSAMPLING that randomly samples points based on the distribution of the weights and runs for $5\log(1/\varepsilon)$ number of rounds (ref. Section 4).
- MWUEMP: a cheaper version of MWU which is terminated early if the training error has reached $\varepsilon|D|$.

For all these methods, SVM (from libSVM [20] library), with a linear kernel, was used as the underlying classifier. We report training accuracy and communication cost. The training accuracy is computed over the combined dataset D with an ε value of 0.05 (where applicable). The communication cost (in words) of all methods are reported as ratios with reference to MWUEMP as the base method. All numbers reported are averaged over 10 runs of the experiments; standard deviations are reported where appropriate. For MWU and MWUEMP, we use $\rho = 0.75$.

Communication Cost Computation. Each example point incurs a cost of $d+1$ (d words to describe its position in $\mathbb{R}^d$ and 1 word to describe its sign). Similarly, each linear classifier requires $d+1$ words of communication (d words to describe its direction and 1 word to describe its offset). Note that given our cost computation, for some datasets the cost of RAND, RANDEMP and MWU can exceed the cost of NAIVE (see, for example, *Cancer*).

Datasets. Six datasets, three each for two-party and four-party case, have been generated synthetically from mixture of Gaussians. Each Gaussian has been carefully seeded to generate different data partitions. For *Synthetic1*, *Synthetic2*, *Synthetic4*, *Synthetic5*, each node contains 5000 data points (2500 positive and 2500 negative) whereas for *Synthetic3* and *Synthetic6*, each node contains 8500 data points (4250 positive and 4250 negative) and all of these datapoints lie in 50 dimensions. Additionally, we investigate the performance of our protocols on real-world datasets. We use *Cancer* and *Mushroom* from the LibSVM data repository [20] as these datasets are linearly or almost linearly separable. This shows that although our protocols were designed for noiseless data they work well on noisy datasets too. However, when applied on noisy data, we do not guarantee the accuracy bounds that were claimed for noiseless datasets.

In Tables 1-2, we highlight (in bold) the protocol that performs the best. By best we mean that the method has the cheapest communication cost as well an accuracy that is more that $(1-\varepsilon)$ times the optimal, i.e., 95% for $\varepsilon = 0.05$. As will be frequently seen for VOTING, the communication cost is the cheapest but the accuracy is far from the desired ε-error specified, and in such circumstances we do not deem VOTING as the best method.

Table 1. Mean accuracy (Acc) and communication cost (Cost) required for synthetic datasets

	Synthetic1		***Synthetic2***		***Synthetic3***	
	Acc	**Cost**	**Acc**	**Cost**	**Acc**	**Cost**
	2-party					
NAIVE	99.23 (0.0)	49.0	97.91 (0.0)	6.18	97.39 (0.0)	19.1
VOTING	**95.00 (0.0)**	**0.01**	60.64 (0.0)	0.01	74.55 (0.0)	0.01
RAND	99.02 (0.0)	29.4	97.72 (0.0)	3.71	97.16 (0.0)	6.74
RANDEMP	96.64 (0.1)	4.41	**95.13 (0.1)**	**0.56**	96.03 (0.1)	1.01
MAXMARG	96.39 (0.0)	4.26	93.76 (0.0)	6.18	73.62 (0.0)	19.1
MWU	98.66 (0.1)	49.5	97.59 (0.1)	6.24	97.11 (0.1)	11.3
MWUEMP	95.00 (0.0)	1.00	95.17 (0.1)	1.00	**95.25 (0.2)**	**1.00**
	Synthetic4		***Synthetic5***		***Synthetic6***	
	4-party					
NAIVE	99.26 (0.0)	100	97.97 (0.0)	12.7	97.47 (0.0)	54.8
VOTING	**95.00 (0.0)**	**0.01**	65.83 (0.0)	0.01	75.52 (0.0)	0.01
RAND	99.18 (0.0)	60.0	97.83 (0.0)	7.63	97.39 (0.0)	19.4
RANDEMP	97.33 (0.1)	9.00	96.61 (0.1)	1.15	96.67 (0.1)	2.90
MAXMARG	95.95 (0.0)	0.82	93.94 (0.0)	15.2	75.05 (0.0)	80.2
MWU	98.03 (0.2)	34.8	97.30 (0.1)	4.45	96.87 (0.1)	11.2
MWUEMP	95.11 (0.3)	1.00	**95.11 (0.2)**	**1.00**	**95.45 (0.2)**	**1.00**

6.1 Synthetic Results

Table 1 compares the performance metrics of the aforementioned protocols for *two*-parties. As can be seen, VOTING performs the best for *Synthetic1* and RANDEMP performs the best for *Synthetic2*. For *Synthetic3*, MWUEMP requires the least amount of communication to learn an ε-optimal distributed classifier. Note that, for *Synthetic2* and *Synthetic3*, both VOTING and MAXMARG fail to produce a ε-optimal ($\varepsilon = 0.05$) classifier. MAXMARG exhibits this behavior despite incurring a communication cost that is as high as NAIVE (i.e., the accumulated cost of the support points become the same as the cost of NAIVE at which point we stop the algorithm).

In Table 1, most of the two-party results carry over to the multiparty case. VOTING is the best for *Synthetic4* whereas MWUEMP is the best for *Synthetic5* and *Synthetic6*. As earlier, both VOTING and MAXMARG do not yield 0.05-optimal classifiers for *Synthetic5* and *Synthetic6*.

Figure 1 (for two-party using *Synthetic1*) shows the communication costs (in *log-scale*) with variations in the number of data points per node and the dimension of the data. Note that we do not report the numbers for MAXMARG since MAXMARG takes a long time to finish. However, for *Synthetic1* the numbers for MAXMARG are similar to those of RANDEMP and so their traces are similar. Note that in Figure 1, the cost of NAIVE increases as the number of dimensions increase. This is because the cost is multiplied by a factor of $(d+1)$, when expressed in words.

6.2 Real-World Results

Table 2 presents results for two and four-party protocols using real-world datasets. Other than two-party case for *Mushroom*, VOTING performs best in all other cases.

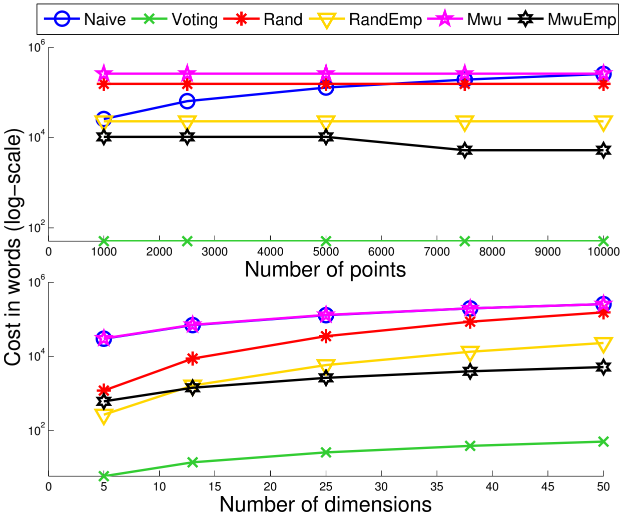


Fig. 1. Communication cost vs Size and Dimensionality for 2-party protocol

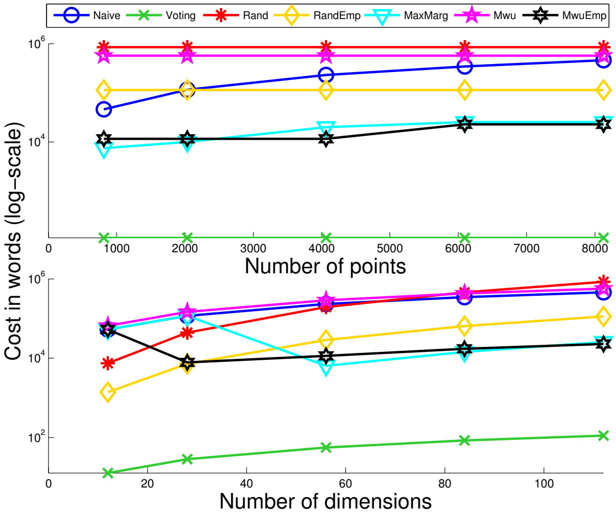


Fig. 2. Communication cost vs Size and Dimensionality for 2-party protocol

However, note that VOTING does not yield a 0.05-optimal distributed classifier for *Mushroom* using two-party protocol.

The results for communication cost (in *log-scale*) versus data size and communication cost (in *log-scale*) versus dimensionality are provided in Figure 2 for two-party protocol using the *Mushroom* dataset. MWUEMP (denoted by the black line) is comparable to MAXMARG and cheaper than all other baselines (except VOTING).

Table 2. Results for *Cancer* ($|D| = 683$, $d = 10$) and *Mushroom* ($|D| = 8124$, $d = 112$)

	Cancer		***Mushroom***		***Cancer***		***Mushroom***	
	Acc	**Cost**	**Acc**	**Cost**	**Acc**	**Cost**	**Acc**	**Cost**
	2-party				4-party			
Naive	97.07 (0.0)	3.34	100.00 (0.0)	20.01	97.07 (0.0)	1.00	100.00 (0.0)	28.61
Voting	**97.36 (0.0)**	**0.01**	88.38 (0.0)	0.00	**97.36 (0.0)**	**0.03**	**95.67 (0.0)**	**0.01**
Rand	97.16 (0.1)	4.52	100.00 (1.1)	36.97	97.19 (0.1)	12.81	100.00 (0.6)	105.70
RandEmp	96.90 (0.2)	0.88	100.00 (0.0)	4.97	96.99 (0.1)	2.50	99.99 (0.0)	14.20
MaxMarg	96.78 (0.0)	0.22	100.00 (0.0)	1.11	96.78 (0.0)	0.56	100.00 (0.0)	2.34
Mwu	97.36 (0.2)	49.51	100.00 (0.0)	24.88	97.00 (0.2)	48.46	100.00 (0.1)	24.65
MwuEmp	96.87 (0.4)	1.00	**99.73 (0.5)**	**1.00**	96.97 (0.3)	1.00	98.86 (0.4)	1.00

Remarks. The goal of our experiments was to show that our protocols perform well, particularly on difficult or adversarially partitioned datasets. For easy datasets, any baseline technique can perform well. Indeed, Voting performs the best on *Synthetic1* and *Synthetic4* and RandEmp performs better than others on *Synthetic2*. For the remaining three cases on synthetic datasets, MwuEmp outperforms the other baselines. On real world data, Voting usually performs well. However, as we have seen, for some datasets Voting and MaxMarg fail to yield an ε-optimal classifier. In particular for *Mushroom*, using the two-party protocol, the accuracy achieved by Voting is far from ε-optimal. These results show that there exists scenarios where Voting and MaxMarg perform particularly worse and thus are not safe strategies.

7 Distributed Optimization

Thus far, we have focused on solving the binary classification problem in a distributed setting. Classification however is merely one kind of learning task, and one might ask whether other problems can be addressed using the MWU framework we describe. A useful insight here is that many learning tasks can be formulated as optimization problems in which the data act as constraints. For example, a simple linear SVM formulation has for each labeled point (x,y) the constraint $y(\langle w,x\rangle + b) \geq 1$.

Thus, a natural way to study a general class of learning tasks via optimization is as follows. Each player i has a set of constraints $C_i = \{f_{ij}(x) \geq 0\}$, and the goal is to solve the optimization $\min g(x)$ subject to the union of constraints $\cup_i C_i$. As earlier, our goal is to solve the above with minimum communication.

7.1 Optimization via Multiplicative Weight Updates

A first observation is that the MWU framework described in previous sections applies to distributed optimization. Consider the problem of solving a general LP of the form $\min g^\top x$, subject to $Ax \geq b$, $x \in P$, where P is a set of "soft" constraints (for example, $x \geq 0$) and $Ax \geq b$ are the "hard" constraints. Let $z^* = \min g^\top x^*$ be the optimal value of the LP, obtained at x^*. Then the multiplicative weight update method can be used to obtain a solution $\tilde{x}$ such that $z^* = g^\top \tilde{x}$ and all (hard) constraints are satisfied approximately, i.e $\forall i$, $A_i\tilde{x} \geq b_i - \varepsilon$, where $A_i x \geq b_i$ is one row of the constraint matrix. We call such a solution a *soft-ε-approximation* (to distinguish it from a traditional approximation in which all constraints would be satisfied *exactly* and the objective would be approximately achieved).

The standard protocol works as follows [21]. We assume that the optimal z^* has been guessed (this can be determined by binary search), and define the set of "soft" constraints to be $\mathcal{P} = P \cup \{x \mid g^\top x = z^*\}$. Typically, it is easy to check for feasibility in $\mathcal{P}$. We define a *width* parameter $\rho = \max\{\max_{i\in[n],x\in\mathcal{P}} A_i x - b_i, 1\}$. Initialize $m_i(0) = 0$. Then we run $T = O((\rho^2/\varepsilon^2)\ln n)$ iterations (with $t = 1, 2, \ldots, T$) of the following: (1) Set $p_i(t) = \exp(-\varepsilon m_i(t-1)/2)$, (2) Find feasible $x(t)$ in $\mathcal{P} \cup \{x \mid \sum_i p_i A_i x \geq \sum_i p_i b_i\}$, (3) $m_i(t) = m_i(t-1) + A_i x(t) - b_i$. At the end, we return $\overline{x} = (1/t)\sum_t x(t)$ as our soft-ε-approximation for the LP.

We now describe a two-party distributed protocol for linear programming adapted from this scheme. The protocol is asymmetric. Player A finds feasible values of x and player B maintains the weights m_i. Specifically, player A constructs a feasible set $\mathcal{P}$ consisting of the original feasible set P and all of its own constraints. As above, B initializes a weight vector m to all zeros, and then sends over the *single* constraint $\sum_i p_i A_i x \geq \sum_i p_i b_i$ to A. Player A then finds a feasible x using this constraint as well as $\mathcal{P}$ (solving a linear program) and then sends the resulting x back to B, who updates its weight vector m. Each round of communication requires $O(d)$ words, and there are $O((\rho^2/\varepsilon^2)\ln n)$ rounds of communication. Notice that this is exponentially better than merely sending over all constraints.

Theorem 7. *There is a 2-player distributed protocol that uses $O((d\rho^2/\varepsilon^2)\ln n)$ words of communication to compute a soft-ε-approximation for a linear program.*

A similar result applies for SDP (based on an existing primal MWU-based SDP algorithm [21]) as well as other optimizations for which the MWU applies, such as rank minimization [22], etc.

7.2 Optimization via Multi-pass Streaming

We now present a different approach to distributed optimization. This approach introduces a novel reduction from *multipass streaming* to distributed communication. Given the extensive literature on streaming algorithms[23], this reduction is useful as a design strategy for algorithms in this model. Specifically, we show how fixed dimensional linear programming can be solved using this reduction.

A *streaming algorithm* [23] takes as input a sequence of items $x_1, \ldots x_n$. The algorithm is allowed working space that is *sublinear in n*, and is only allowed to look at each item once as it *streams* past. In *multipass* streaming, the algorithm may make more than one pass over the data, but is still limited to sublinear working space and a single look at each item in each pass. Lemma 2 shows that any (multipass) streaming algorithm can be used to build a multiparty distributed protocol.

Lemma 2. *A streaming algorithm for problem P that has s words of working storage and makes r passes over the data can be made into a k-player distributed algorithm for P that uses krs words of communication.*

Note that streaming algorithms often have $s = O(\mathrm{poly}\log n)$ and $r = O(\log n)$, yielding (sublinear) $O(k\ \mathrm{poly}\log n)$ words of communication.

We can apply this lemma to get a distributed algorithm for fixed-dimensional linear programming[2]. This relies on an existing result [24]:

Theorem 8 ([24]). *For n halfspaces in $\mathbb{R}^d$ (d constant), the lowest intersection point can be computed by a $O(1/\delta^{d-1})$-pass Las Vegas algorithm that uses $O((1/\delta^{O(1)})n^{\delta})$ space and runs in time $O((1/\delta^{O(1)})n^{1+\delta})$ with high probability, for any constant $\delta > 0$.*

Corollary 1. *There is a k-player algorithm for solving distributed linear programming that uses $O(k(1/\delta^{d+O(1)})n^{\delta})$ communication, for any constant $\delta > 0$.*

While the above streaming algorithm can be applied as a blackbox in Corollary 1, looking deeper into the streaming algorithm reveals room for improvement. As in the case of classification, suppose that we are permitted to violate an ε-fraction of the constraints. It turns out that the above streaming algorithm achieves its bounds by eliminating a fixed fraction of constraints in each space, and thus requires $\log_r n$ passes, where $r = n^{\Theta(\delta)}$. If we are allowed to violate an ε-fraction of constraints, we need only run the algorithm for $\log_r 1/\varepsilon$ passes, where r is now $O(1/\varepsilon^{\Theta(\delta)})$. This allows us to replace n in all terms by $1/\varepsilon$, resulting in an algorithm with communication *independent of n*.

Corollary 2. *There is a k-player algorithm for distributed linear programming that violates at most an ε-fraction of the constraints, and uses $O(k(1/\delta^{d+O(1)})(1/\varepsilon)^{\delta})$ communication, for any constant $\delta > 0$.*

8 Conclusion

In this work, we have proposed a simple and efficient protocol that learns an ε-optimal distributed classifier for hyperplanes in arbitrary dimensions. The protocol also gracefully extends to k-players. Our proposed technique WEIGHTEDSAMPLING relates to the MWU-based meta framework and we exploit this connection to extend WEIGHTEDSAMPLING for distributed convex optimization problems. This makes our protocol applicable to a wide variety of distributed learning problems that can be formulated as a convex optimization task over multiple distributed nodes.

References

[1] Daumé III, H., Phillips, J., Saha, A., Venkatasubramanian, S.: Protocols for learning classifiers on distributed data. In: AISTATS 2012 (2012)

[2] Bekkerman, R., Bilenko, M., Langford, J. (eds.): Scaling up Machine Learning: Parallel and Distributed Approaches. Cambridge University Press (2011)

[3] McDonald, R., Hall, K., Mann, G.: Distributed training strategies for the structured perceptron. In: NAACL HLT (2010)

[2] Fixed-dimensional linear programming is the case of linear programming where the dimension is treated as a constant.

[4] Mann, G., McDonald, R., Mohri, M., Silberman, N., Walker, D.: Efficient large-scale distributed training of conditional maximum entropy models. In: NIPS (2009)

[5] Collins, M.: Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In: EMNLP (2002)

[6] Bauer, E., Kohavi, R.: An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine Learning 36(1-2) (1999)

[7] Dekel, O., Gilad-Bachrach, R., Shamir, O., Xiao, L.: Optimal distributed online prediction using mini-batches. arXiv:1012.1367 (2010)

[8] Chu, C.T., Kim, S.K., Lin, Y.A., Yu, Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-reduce for machine learning on multicore. In: NIPS (2007)

[9] Teo, C.H., Vishwanthan, S., Smola, A.J., Le, Q.V.: Bundle methods for regularized risk minimization. J. Mach. Learn. Res. 11, 311–365 (2010)

[10] Zinkevich, M., Weimer, M., Smola, A., Li, L.: Parallelized stochastic gradient descent. In: NIPS (2010)

[11] Servedio, R.A., Long, P.: Algorithms and hardness results for parallel large margin learning. In: NIPS (2011)

[12] Balcan, M.F., Blum, A., Fine, S., Mansour, Y.: Distributed learning, communication complexity and privacy. In: COLT 2012, arXiv:1204.3514 (to appear, June 2012)

[13] Daumé III, H., Phillips, J.M., Saha, A., Venkatasubramanian, S.: Efficient protocols for distributed classification and optimization. arXiv:1204.3523

[14] Anthony, M., Bartlett, P.L.: Neural Network Learning: Theoretical Foundations, Cambridge (2009)

[15] Cormode, G., Muthukrishnan, S., Yi, K.: Algorithms for distributed functional monitoring. In: SODA (2008)

[16] Cormode, G., Muthukrishnan, S., Yi, K., Zhang, Q.: Optimal sampling from distributed streams. In: PODS (2010)

[17] Matousek, J.: Approximations and optimal geometric divide-and-conquer. In: STOC (1991)

[18] Chazelle, B.: The Discrepancy Method, Cambridge (2000)

[19] Matoušek, J.: Geometric Discrepancy. Springer (1999)

[20] Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM TIST 2(3) (2011)

[21] Arora, S., Hazan, E., Kale, S.: Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In: FOCS (2005)

[22] Meka, R., Jain, P., Caramanis, C., Dhillon, I.S.: Rank minimization via online learning. In: ICML (2008)

[23] Muthukrishnan, S.: Data streams: algorithms and applications. Foundations and trends in theoretical computer science. Now Publishers (2005)

[24] Chan, T.M., Chen, E.Y.: Multi-pass geometric algorithms. Disc. & Comp. Geom. 37(1), 79–102 (2007)

Appendix

Proof of Theorem 3

Proof. At the start of each round t, let ϕ_t be the potential function given by the sum of weights of all points in that round. Initially, $\phi_1 = \sum_{x_i \in D_B} w_i = n$ since by definition for each point $x_i \in D_B$ we have $w_i = 1$.

Then in each round, A constructs a classifier h^t_A at B to correctly classify the set of points that accounts for at least $1 - c$ fraction of the total weight by Lemma 1. All other misclassified points are upweighted by $(1+\rho)$. Hence, for round $(t+1)$ we have $\phi^{t+1} \leq \phi^t\left((1-c)+c(1+\rho)\right) = \phi^t\left(1+c\rho\right) = n\left(1+c\rho\right)^t$.

Let us consider the weight of the points in the set $S \subset D_B$ that have been misclassified by a majority of the T classifiers (after the protocol ends). This implies every point in S has been misclassified *at least* $T/2$ number of times and *at most* T number of times. So the minimum weight of points in S is $(1+\rho)^{T/2}$ and the maximum weight is $(1+\rho)^T$.

Let n_i be the number of points in S that has weight $(1+\rho)^i$ where $i \in [T/2, T]$. The potential function value of S after T rounds is $\phi_S^T = \sum_{i=T/2}^{T} n_i(1+\rho)^i$. Our claim is that $\sum_{i=1}^{T} n_i = |S| \leq \varepsilon n$. Each of these at most $|S|$ points have a weight of at least $(1+\rho)^{T/2}$. Hence we have

$$\phi_S^T = \sum_{i=T/2}^{T} n_i(1+\rho)^i \geq (1+\rho)^{T/2} \sum_{i=T/2}^{T} n_i = (1+\rho)^{T/2}|S|.$$

Relating these two inequalities we obtain the following,

$$|S|(1+\rho)^{T/2} \leq \phi_S^T \leq \phi^T \leq n\left(1+c\rho\right)^T.$$

Hence (using $T = 5\log_2(1/\varepsilon)$)

$$|S| \leq n\left(\frac{(1+c\rho)}{(1+\rho)^{1/2}}\right)^{5\log_2(1/\varepsilon)} = n(1/\varepsilon)^{5\log_2\left(\frac{(1+c\rho)}{(1+\rho)^{1/2}}\right)}. \tag{2}$$

Setting $c = 0.2$ and $\rho = 0.75$ we get $5\log_2\left((1+c\rho)/(1+\rho)^{1/2}\right) < -1$ and thus $|S| < n(1/\varepsilon)^{-1} < \varepsilon n$, as desired since $\varepsilon < 1$. Thus each round uses $O(d)$ points yielding a total communication of $O(d^2\log(1/\varepsilon))$ words.

Proof of Theorem 5

Proof. The correctness and bound of $T = O(\log(1/\varepsilon))$ rounds follows from Theorem 3, since, aside from the total weight gathering step, from party P_1's perspective it appears to run the protocol with some party B where B represents parties $P_2, P_3, \ldots, P_k$. The communication for P_1 to collect the samples from all parties is $O(kd + ds_{c,d}) = O(kd + d^2\log\log(1/\varepsilon))$. And it takes $O(dk)$ communication to return h_A to all $k-1$ other players. Hence the total communication over $T = O(\log(1/\varepsilon))$ rounds is $O((kd + d^2\log\log(1/\varepsilon))\log(1/\varepsilon))$ as claimed.

Proof of Lemma 2

Proof. First consider the case when $k = 2$. Consider a streaming algorithm S satisfying the conditions above. The simulation works by letting the first player A simulate the first half of S, and letting the second player B simulate the second half. Specifically, the first player A simulates the behavior of S on its input. When this simulation of S exhausts the input at A, A sends over the contents of the working store of S to B. B restarts S on its input using this working store as S's current state. When B has finished simulating S on its input, it sends the contents of the working storage back to A. This completes one pass of S, and used s words of communication. The process continues for r passes.

If there are k players $A_1, \ldots, A_k$ instead of two, then we fix an arbitrary ordering of the players. The first player simulates S on its input, and at completion passes the contents of the working store to the next one, and so on. Each pass now requires $O(ks)$ words of communication, and the result follows.

The Safe Bayesian:
Learning the Learning Rate via the Mixability Gap

Peter Grünwald

CWI, Amsterdam and Leiden University, The Netherlands
pdg@cwi.nl

Abstract. Standard Bayesian inference can behave suboptimally if the model is wrong. We present a modification of Bayesian inference which continues to achieve good rates with wrong models. Our method adapts the Bayesian learning rate to the data, picking the rate minimizing the cumulative loss of sequential prediction by posterior randomization. Our results can also be used to adapt the learning rate in a PAC-Bayesian context. The results are based on an extension of an inequality due to T. Zhang and others to dependent random variables.

1 Introduction

Problem 1: Bayes when the Model is Wrong Standard Bayesian inference may fail if the probability model $\mathcal{P}$ under consideration is "wrong yet useful". Grünwald and Langford (2007) (GL from now on) exhibit cases in which the posterior never concentrates, putting substantial weight on many "bad" distributions even in the limit of infinite sample size. As a result, predictions based on the posterior remain suboptimal forever. This problem can be addressed by equipping Bayes with a learning rate η as in (Zhang, 2006a). Standard Bayesian inference corresponds to $\eta = 1$; for small enough η, Bayesian inference will become well-behaved again and its predictions will become optimal in the limit. However, picking η too small may lead to an unnecessarily slow convergence rate. The appropriate choice for η depends on the true distribution, which is unknown, and it is unclear how to estimate it from data: GL show that marginalizing out η (as a Bayesian would prefer) does not solve the problem, and picking the η that maximizes the Bayesian marginal likelihood of the data $Z^n = Z_1, \ldots, Z_n$ does not help either (see also Example 3, Example 4 and Figure 1, this paper's **essential picture**).

Problem 2: PAC-Bayesian Learning Rates In statistical learning theory, one consider models Θ of predictors defined relative to some loss function LOSS, e.g. Θ may be a set of classifiers and LOSS may be the 0/1-loss. In *relative PAC-Bayesian bounds* (Audibert, 2004, Zhang, 2006b, Catoni, 2007) one proves frequentist convergence bounds of randomized predictors which depend on some user-specified "prior" distribution over Θ. The bounds are typically optimized by setting the randomized predictor equal to a pseudo-Bayesian posterior at some optimal learning rate η, which once again depends on the unknown true

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 169–183, 2012.

distribution. Algorithms for estimating η from the data have been proposed for special settings (Audibert, 2004), but so far, a general approach has been lacking.

The Safe Bayesian. We address both problems at once by picking the $\hat{\eta}$ that maximizes the "sequentially randomized" Bayesian marginal log-likelihood, which for priors with finite support can be reinterpreted as the $\hat{\eta}$ minimizing the cumulative loss of the Hedge(η) algorithm (Freund and Schapire, 1997). We then predict by the Cesàro average of the Bayesian posteriors at $\hat{\eta}$. We extend this *safe Bayesian* algorithm to the statistical learning case by defining pseudo-probabilities $p_\theta(y \mid x) \propto e^{-\text{LOSS}(y,\theta(x))}$ in the usual manner.

In our first result, Theorem 1, we show that for all η smaller than some "critical" η_{CRIT}, we can expect a small *mixability gap*, a notion reminiscent of Vovk's (1990, 2001) fundamental concept of *mixability* for individual sequence prediction. In our context a small mixability gap means that the expected cumulative log-loss one obtains by *randomizing* according to the posterior is close to the cumulative log-loss one obtains by *mixing* the posterior. If the posterior concentrates, then the mixability gap is small, and we may think of the $\hat{\eta}$ inferred by our algorithm as estimating the largest rate at which the posterior does concentrate. Our main result, Theorem 2 shows that, broadly speaking, the convergence rates achieved by the safe Bayesian algorithm are optimal for the underlying, unknown true distribution in several settings. Specifically, if the model is correct or convex, we perform essentially as well as standard Bayesian inference, which in this case is among the best methods available. Yet when the model is incorrect, in the setting of Grünwald and Langford (2007), unlike standard Bayes, the safe Bayesian posterior does learn to predict optimally, i.e. as well as the single distribution in the model that predicts best.

In Section 2 we introduce notation, concepts and present the safe Bayesian algorithm. Since the algorithm can be applied in a wide variety of contexts (standard Bayes, statistical learning, Hedge-like) this section is, unfortunately, long. In Section 3 we introduce η_{CRIT}, which allows us to give a second, detailed introduction to the results that are to follow. Section 4 gives our first result, relating randomized ("Gibbs") to standard Bayesian prediction and gives, in Figure 1, a crucial picture. Section 5 gives our main result, Theorem 2, showing that the Safe Bayesian algorithm performs comparably to an algorithm that knows the critical learning rate in advance. We also compare our results to Grünwald (2011) who already provided a procedure that adapts to η_{CRIT} in a much more restricted setting. In Appendix A we prove Theorem 1 and 2. The latter is built upon Theorem 3, an extension of a PAC-Bayesian style inequality which is of independent interest. Due to space constraints we omit the proof of Theorem 3, of the existence of $\mathcal{Q}$ in (4), and some additional discussion in Example 5. This additional material can be found on the author's web page.

2 Preliminaries; The Algorithm

Statistical Setting. We first present our algorithm in the statistical setting, and then show how it can be adjusted to decision-theoretic settings. Consider a "model" $\mathcal{P} = \{p_\theta \mid \theta \in \Theta\}$ of densities on a space $\mathcal{Z}$ relative to some fixed measure μ. The densities may be, but are not necessarily, probability densities or mass functions: we only require that for all $z \in \mathcal{Z}$, $p_\theta(z) \geq 0$, and $\int_{\mathcal{Z}} p_\theta d\mu < \infty$. We extend p_θ to sequences $z^n = z_1, \ldots, z_n$ of n outcomes by $p_\theta(z^n) = \prod_{i=1}^n p_\theta(z_i)$. There are no restrictions on the structure of Θ; thus $\mathcal{P}$ may very well be a 'nonparametric' set such as, say, the set of all Gaussian mixtures on $\mathcal{Z}$ with a countable number of components. Often we are interested in estimating a *conditional* probability density. In that case, $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, and $p_\theta(z)$ abbreviates $p_\theta(y \mid x)$, the conditional density of y given x, and our requirement becomes that for all $x \in \mathcal{X}$, $\int_{\mathcal{Y}} p_\theta(y \mid x) d\mu_x < \infty$ for some underlying measure μ_x. The abbreviation $z \equiv y \mid x$ is unusual, but in our case harmless, and greatly simplifies notation.

An *estimator* is a function $\breve{\nu} : \bigcup_{n=1}^\infty \mathcal{Z}^n \to \Theta$ where the function evaluated at z^n is denoted $\breve{\nu} | z^n$. If Z^n has a distribution P^*, $\breve{\nu}$ becomes a random variable and we omit the argument '$\mid Z^n$' if it is clear from the context. A *randomized* estimator is a function $\breve{W} : \bigcup_{n=1}^\infty \mathcal{Z}^n \to \text{dist}(\Theta)$, where $\text{dist}(\Theta)$ is the set of all distributions on Θ. We write $\breve{W} \mid Z^n$ for the estimate for data Z^n. Following Zhang (2006a,b), for any prior Π with density π relative to some underlying measure ρ, we define the *generalized Bayesian posterior*, denoted as $\Pi \mid Z^n, \eta$, as the distribution on Θ with density

$$\pi(\theta \mid z^n, \eta) := \frac{p_\theta^\eta(z^n)\pi(\theta)}{\int_\Theta p_\theta^\eta(z^n)\pi(\theta)\rho(d\theta)} = \frac{p_\theta^\eta(z^n)\pi(\theta)}{\mathbf{E}_{\theta\sim\Pi}[p_\theta^\eta(z^n)]}. \tag{1}$$

Algorithm 1. The Safe Bayesian Algorithm. In the DTOL and statistical learning interpretation, log-loss in the fifth-to-last line is replaced by the loss of interest $\ell_{\Pi|z^{i-1},\eta}(z_i)$. The definition of $\kappa_{\max}$ is explained below (8).

```
Input: data $z_1, \ldots, z_n$, model $\{p_\theta \mid \theta \in \Theta\}$, prior $\Pi$ on $\Theta$.
Output: Distribution on $\Theta$.
$\kappa_{\max} := \lceil \log_2(2\sqrt{n} \ln V) \rceil$ with $V$ as in (3),
$\mathcal{S}_n := \{1, 2^{-1}, 2^{-2}, 2^{-3}, \ldots, 2^{-\kappa_{\max}}\}$ ;
for all $\eta \in \mathcal{S}_n$ do
    $s_\eta := 0$ ;
    for $i = 1 \ldots n$ do
        Compute generalized Bayes posterior $\Pi(\cdot \mid z^{i-1}, \eta)$ with learning rate $\eta$;
        Calculate "posterior expected loss" of predicting actual next outcome:
        $r := E_{\theta\sim\Pi|z^{i-1},\eta}[-\ln p_\theta(z_i)]\ [= \boldsymbol{\ell_{\Pi|z^{i-1},\eta}(z_i)}]$ ; set $s_\eta := s_\eta + r$;
    end
end
Choose $\hat{\eta} = \arg\min_{\eta\in\mathcal{S}_n}\{s_\eta\}$ (if min achieved for several $\eta \in \mathcal{S}_n$, pick largest) ;
Output distribution $\breve{W}_{\text{SAFE}} \mid Z^n := \text{CES}(\Pi \mid \hat{\eta}; Z^n) = n^{-1}\sum_{i=1}^n \Pi(\cdot \mid z^i, \hat{\eta})$.
```

We can think of the generalized Bayesian posterior as a randomized estimator. For a randomized estimator $\breve{W}$ and a sample Z^n, we define the corresponding (randomized) Cesàro-averaged estimator as $\text{CES}(\breve{W}; Z^n) := n^{-1} \sum_{i=1}^{n} \breve{W} | \, Z^i$. We are now ready to present the safe Bayesian algorithm.

The algorithm implements a particular randomized estimator: it picks the $\hat{\eta}$ for which the cumulative log-loss of sequentially predicting by *randomizing* according to the posterior ("Gibbs sampling") is minimized (this is different from standard Bayesian prediction, which *mixes* rather than randomizes). It then outputs the corresponding Cesàro estimator. The use of randomization makes $\hat{\eta}$ very different from a standard 'empirical Bayes' estimate — see Example 4.

DTOL Setting. We consider a variation of the original decision-theoretic online (DTOL) setting (Freund and Schapire, 1997) along the lines of (Zhang, 2006a). Let $\mathcal{A}$ be a set of *actions*, where each $a \in \mathcal{A}$ is identified by its *loss* $\ell_a : \mathcal{Z} \rightarrow \mathbb{R}$. Thus the loss of action a on outcome z is $\ell_a(z)$. We let $\Theta \subset \mathcal{A}$ be a subset of actions whose losses $\ell_\theta(z_i)$ can be observed at each time point i. As in the original DTOL setting, the learner may not have access to z^{i-1} directly. We assume that the learner is allowed to *randomize*, i.e. for any distribution W in $\mathcal{A}$, all $z \in \mathcal{Z}$ we define

$$\ell_W(z) := \mathbf{E}_{a \sim W}[\ell_a(z)], \tag{2}$$

and we assume that for each such W, the learner is allowed to play an action $a_W \in \mathcal{A}$ with, for all $z \in \mathcal{Z}$, $\ell_{a_W}(z) \leq \ell_W(z)$. This is achieved either automatically (e.g. with convex loss functions defined on convex $\mathcal{A}$, such that for each W an a_W trivially exists) or by definition; e.g. in the PAC-Bayesian literature, it is usually assumed that the learner is allowed to play a randomized action W and is satisfied by evaluating its performance 'on average' (Catoni, 2007).

To apply our algorithm in the DTOL setting, we define pseudo-probabilities $p_a(z) := \exp(-\ell_a(z))$ in the usual manner, for each $a \in \mathcal{A}$, so that $-\ln p_a(z) = \ell_a(z)$, as already indicated in the fifth-to-last-line in Algorithm 1. Readers familiar with the HEDGE-algorithm (Freund and Schapire, 1997, Chaudhuri et al., 2009) will notice that the safe Bayesian algorithm really just runs Hedge at different learning rates η, picking the $\hat{\eta}$ that minimizes cumulative loss with hindsight, and then makes a Cesàro-averaged prediction of the n previous Hedge predictions with this loss. Note however that, while the algorithm employs an on-line learning method, our aim is to prove bounds on its batch behaviour after observing z^n (Theorem 2).

Statistical Learning Setting. A special case of the decision-theoretic setting is standard statistical learning in which $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, each θ is a function $\theta : \mathcal{X} \rightarrow \mathcal{Y}'$ and $\ell_\theta = \text{LOSS}(Y, \theta(X))$ where $\text{LOSS} : \mathcal{Y} \times \mathcal{Y}' \rightarrow \mathbb{R}$ is some loss function, e.g. the 0/1-loss in the classification setting with $\mathcal{Y} = \mathcal{Y}' = \{0, 1\}$, and $\text{LOSS} : \mathcal{Y} \times \mathcal{Y} \rightarrow \{0, 1\}$ given by $\text{LOSS}(y, \hat{y}) := |y - \hat{y}|$.

Condition on $\mathcal{P}/\Theta$. Throughout this paper we assume the model $\mathcal{P}$ satisfies the following condition. Let

$$U(\mathcal{P}) := \sup_{z \in \mathcal{Z}} \sup_{p, p' \in \mathcal{P}} \frac{p(z)}{p'(z)} \text{ and } V = V(\mathcal{P}) = 2U(\mathcal{P}). \tag{3}$$

If $\mathcal{P}$ is clear from the context we write V rather than $V(\mathcal{P})$ (the reason for distinguishing between V and U is just for notational convenience in stating our results later; it is due to the factor 2 in (4) below). We always assume that the ratio in (3) is well-defined for all $z \in \mathcal{Z}$ and that $1 < V < \infty$. We may think of $U(\mathcal{P})$ as the maximum ratio between the density of z (or $y \mid x$) assigned by different $p \in \mathcal{P}$. In the DTOL setting with bounded loss functions like 0/1-loss, this condition is automatically satisfied with $V \leq \exp(L_{\max})$ with $L_{\max} = \sup_{z \in \mathcal{Z}, \theta, \theta' \in \Theta} (\ell_\theta(z) - \ell_{\theta'}(z))$. Yet in general density estimation and unbounded loss settings, this is currently a serious restriction on our results.

3 The Critical η_{CRIT} — Extended Introduction

From now on, we assume the random variables Z and $Z_1, Z_2, \ldots, Z^n$ to be i.i.d. $\sim P^*$, i.e. each outcome Z_i has the same distribution as Z. We denote expectation under P^* by $\mathbf{E}^*$. Let $\mathcal{P}$ be the learner's model. Let $D(P \| q)$ denote the KL divergence between a distribution P and a distribution with density q (possibly defective, i.e. $\int_{\mathcal{Z}} q(z) d\mu \neq 1$). In the longer version of this paper we show that that the set of *best-approximating densities*,

$$\mathcal{Q} := \{q : \inf_{p \in \mathcal{P}} D(P^* \| p) = D(P^* \| q) \ , \ U(\mathcal{P} \cup \{q\}) \leq 2U(\mathcal{P})\}, \tag{4}$$

is not empty, although it may not be contained in $\mathcal{P}$ but only in its (appropriately defined) closure (U is as in (3)).

Our Goal. We focus on the statistical setting; for the DTOL setting, see Example 2 below. In case $\mathcal{P}$ is a standard probability model (all densities integrate to 1) and $\inf_{p \in \mathcal{P}} D(P^* \| p)$ is nonzero, we say that the model $\mathcal{P}$ is *misspecified* (or simply: "wrong"). Our goal is to show that even in this case, the safe Bayesian algorithm outputs an estimator $\breve{W}_{\text{SAFE}}$ that "converges" quickly to $\mathcal{Q}$, in a sense we now make precise. For any two (conditional) densities p and p', we define the *generalized KL (Kullback-Leibler) divergence* (already introduced in the original Kullback and Leibler (1951)!) relative to P^* as

$$D^*(p' \| p) := \mathbf{E}^*_Z[-\ln p(Z) + \ln p'(Z)] = D(P^* \| p) - D(P^* \| p'). \tag{5}$$

Note that, for a best-approximating q as in (4), $D^*(q \| p) \geq 0$ for all $p \in \mathcal{P}$. Theorem 2 below shows that for some $q \in \mathcal{Q}$, $\mathbf{E}_{\theta \sim \breve{W}_{\text{SAFE}} \mid Z^n}[D^*(q \| p_\theta)]$ converges to 0 in expectation as $n \to \infty$ at certain rates. Since trivially, for all $q, q' \in \mathcal{Q}$, all $p \in \mathcal{P}$, $D^*(q \| p) = D^*(q' \| p)$, this means that such convergence takes place simultaneously for *all* $q \in \mathcal{Q}$. Hence, from now on, for ease of exposition, we fix a particular such q and present all results in terms of that q. Since $D^*(q \| p_\theta) \geq 0$ for all $\theta \in \Theta$, this convergence implies that, at large n, $\breve{W}_{\text{SAFE}}$ puts nearly all its mass on Θ with small $D^*(q \| p_\theta)$; in this sense, Theorem 2 shows that $\breve{W}_{\text{SAFE}}$ *concentrates.* To make this precise, we must first define the *critical learning rate.*

The Critical Learning Rate. In the well-specified case, in which P^* has density p^* and we must have $q = p^* \in \mathcal{Q}$, we trivially have that, for $\eta = 1$, for all $p \in \mathcal{P}$:

$$A_\eta(q \| p) := \ \mathbf{E}^*_Z \left[\left(\frac{p(Z)}{q(Z)} \right)^\eta \right] \leq 1, \tag{6}$$

as is seen by writing out the expectation in full and substituting q by p^*. Classical theorems on two-part MDL inference for the well-specified case (Barron and Cover, 1991, Zhang, 2006a, Grünwald, 2007) invariably make use of (6) at some point in the proofs; so do classical results on Bayesian consistency (Doob, 1949), in which (6) is used to establish that $\{p(Z^n)/q(Z^n)\}_{n=1,2...}$ is a martingale. It can be shown (Li, 1999, Kleijn and van der Vaart, 2006) that (6) still holds for $\eta = 1$ if $\mathcal{P}$ is *convex* (Figure 1 in Section 4 will make clear that convexity plays a role here). This is the fundamental reason why standard MDL and Bayesian convergence bounds still hold in that setting. If (6) does not hold for $\eta = 1$ then MDL and Bayes may not converge — see Example 3 below. Luckily, for many types of $\mathcal{P}$, one can still show that (6) holds for some $\eta < 1$. In that case, the standard MDL and Bayesian convergence proofs still go through if the standard posterior is replaced by the η-generalized posterior, leading to results like (11) below. Thus it makes sense to define the *critical exponent* η_{CRIT} as the largest value of η such that, for all $p \in \mathcal{P}$, (6) holds. It is useful to extend the idea slightly so that, for $u \geq 0$, $\eta_{\text{CRIT}}(u)$ is the "critical exponent with slack u/n"; $\eta_{\text{CRIT}}(0)$ is just the critical value as defined before:

$$\eta_{\text{CRIT}}(u) := \sup \left\{ \eta \leq 1 \ : \ \text{for all } p \in \mathcal{P}, \quad \ln \mathbf{E}^*_Z \left[\left(\frac{p(Z)}{q(Z)} \right)^{\eta} \right] \leq \frac{u}{n} \right\}. \qquad (7)$$

This definition implicitly depends on q and n. Clearly $\eta_{\text{CRIT}}(u)$ is increasing in u. By differentiation to η as in (Grünwald, 2011) it follows that also for all $0 < \eta \leq \eta_{\text{CRIT}}(u)$, all $p \in \mathcal{P}$, $\ln \mathbf{E}^*_Z \left[\left(\frac{p(Z)}{q(Z)} \right)^{\eta} \right] \leq \frac{u}{n}$. In case $\mathcal{Q}$ is not a singleton, we define $\eta_{\text{CRIT}}(u)$ as (7) for the $q \in \mathcal{Q}$ that maximizes it for the given u.

How small can η_{CRIT} become? Let V be as in (3). (Grünwald, 2011, Lemma 1) shows that, for all $P^*, \mathcal{P}$, all $0 \leq u \leq n$,

$$\eta_{\text{CRIT}}(u) \geq \eta_{\text{MIN}}(u), \text{ where } \eta_{\text{MIN}}(u) := \tfrac{1}{2 \ln V} \sqrt{\tfrac{u}{n}}. \qquad (8)$$

To get good bounds on the behaviour of the Safe Bayesian algorithm as in Theorem 2 we need to be able to use an η close to $\eta_{\text{CRIT}}(u)$ for a value of $u \geq 0$ that optimizes the bound in Theorem 2. It can be seen that restricting u to be ≥ 1 does not seriously affect the bound, which explains why, in the definition of $\mathcal{S}_n$ in the safe Bayesian algorithm, we could safely restrict ourselves to η no smaller than $O(1/(2 \ln V \sqrt{n}))$. In favourable cases though, $\eta_{\text{CRIT}}(u)$ will be larger than $\eta_{\text{MIN}}(u)$. We shall now see that this leads to faster convergence rates.

Existing Results that we will Extend. We define the generalized Bayesian marginal distribution as $p_{\text{Bayes}}(z^n \mid \eta) := \mathbf{E}_{\theta \sim \Pi} [p^{\eta}_{\theta}(Z^n)]$ and the predictive distribution as $p_{\text{Bayes}}(z_i \mid z^{i-1}, \eta) := p_{\text{Bayes}}(z^i \mid \eta)/p_{\text{Bayes}}(z^{i-1} \mid \eta)$. For $\eta = 1$, these are the standard Bayesian marginal/predictive distributions. By the familiar Bayesian telescoping using (1), p_{Bayes} can be written as product of the generalized posterior predictive distributions:

$$p_{\text{Bayes}}(z^n \mid \eta) = \prod_{i=1}^{n} \tfrac{p_{\text{Bayes}}(z^i \mid \eta)}{p_{\text{Bayes}}(z^{i-1} \mid \eta)} = \prod_{i=1}^{n} \mathbf{E}_{\theta \sim \Pi \mid Z^{i-1}, \eta} \left[p^{\eta}_{\theta}(z_i) \right]. \qquad (9)$$

We also define the *Bayesian redundancy* as

$$\textsc{bayes-red}_n(\eta) := \tfrac{1}{\eta}\mathbf{E}^*_{Z^n}\left[-\ln \tfrac{p_{\text{Bayes}}(Z^n|\eta)}{q^\eta(Z^n)}\right] = \tfrac{1}{\eta}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n - \ln \tfrac{p_{\text{Bayes}}(Z_i|Z^{i-1}\eta)}{q^\eta(Z_i)}\right] \tag{10}$$

For $\eta = 1$, the Bayes redundancy is the expected codelength difference between coding (log-loss prediction) by the code induced by $p_{\text{Bayes}} \mid \eta$ and coding by the code induced by q. This quantity is indeed called the (relative) "redundancy" of the Bayesian mixture code in information theory, see e.g. (Takeuchi and Barron, 1998). We can also give a precise codelength interpretation for $\eta < 1$ via the 'entropification' construction as in Grünwald (2011), but because of space constraints will not do so here. We now informally summarize a central result in MDL and PAC-Bayesian inference in terms of BAYES-RED: for all $0 < \eta < \eta_{\text{CRIT}}(u)$, for some constant C_η depending on η, we have

$$\mathbf{E}^*_{Z^n}\mathbf{E}_{\theta\sim\Pi|Z^n,\eta}\left[D^*(q\|p_\theta)\right] \leq \tfrac{C_\eta}{n} \cdot \textsc{bayes-red}_n(\eta) + R_u, \tag{11}$$

where R_u is a remainder term that becomes negligible for small enough $u \geq 0$. In the remainder of this informal section, we assume that we have chosen u small enough and ignore this term, as well as other precise conditions needed for (11) to hold (R_u will return in the formal statement of our results). (11) is the generic formulation of the result. Variations of (11) are presented by, among others, Zhang (2006a,b), Barron and Cover (1991), Li (1999), Audibert (2004), Catoni (2007). The importance of (11) is that *in practical settings* $\textsc{bayes-red}_n(\eta)$ *grows sublinearly and then (11) implies that (a) the generalized posterior concentrates and (b) leads to asymptotically optimal approximations to q in KL divergence.*

Example 1. [**MDL formulation**] A simple rewriting of the redundancy as in Zhang (2006b) shows that

$$\textsc{bayes-red}_n(\eta) = \mathbf{E}^*_{Z^n}\left[\mathop{\mathbf{E}}_{\theta\sim\Pi|Z^n,\eta}\left[-\ln \frac{p_\theta(Z^n)}{q(Z^n)}\right] + \frac{1}{\eta}D(\,(\Pi \mid Z^n, \eta) \parallel (\Pi \mid \eta)\,)\right] \tag{12}$$

where $D(W\|V) = \int w(\theta)\log(w(\theta)/v(\theta))\rho(d\theta)$ denotes standard KL divergence between distributions with densities W and V respectively. To verify (12), simply replace D by its definition and simplify. If Π has countable support $\Theta' \subset \Theta$ then, irrespective of model correctness, using that for all θ_0, all z^n, $p_{\text{Bayes}}(z^n \mid \eta) = \sum \pi(\theta)p^\eta_\theta(z^n) \geq \pi(\theta_0)p^\eta_{\theta_0}(z^n)$, we have the familiar

$$\textsc{bayes-red}_n(\eta) \leq \mathbf{E}^*_{Z^n}\left[\min_{\theta\in\Theta'}\left\{-\ln\tfrac{p_\theta(Z^n)}{q(Z^n)} + \tfrac{-\ln\pi(\theta)}{\eta}\right\}\right]. \tag{13}$$

If $q = p_{\tilde{\theta}}$ for some $\tilde{\theta} \in \Theta'$, this becomes $\textsc{bayes-red}_n(\eta) \leq -\ln\pi(\tilde{\theta})/\eta$, showing that then prediction by p_{Bayes} stays within $O(1)$ of the best-approximating q.

Example 2. [**Statistical Learning**] Now $z = (x, y)$, $\ell_\theta(z) = \textsc{loss}(y, \theta(x))$, we define $\textsc{risk}(\theta)$ to be the expected loss of θ, i.e. $\textsc{risk}(\theta) := \mathbf{E}^*_Z[\ell_\theta(Z)]$, extended to $\textsc{risk}(W)$ as in (2). Let $\textsc{risk}_{\text{emp}}(W) = n^{-1}\sum_{i=1}^n \ell_W(Z_i)$ be the empirical risk of distribution W. Let $\tilde{\theta}$ be any optimal action within Θ, i.e. $\text{RISK}(\tilde{\theta}) = \min_{\theta \in \Theta} \text{RISK}(\theta)$. Then using $\ell_\theta = -\ln p_\theta$, (12) can be further rewritten as

$$\frac{1}{n}\text{BAYES-RED}_n(\eta) = \mathbf{E}^*_{Z^n}\left[\text{RISK}_{\text{emp}}(\Pi \mid Z^n, \eta)\right] - \text{RISK}(\tilde{\theta}) + \eta^{-1}\mathbf{E}^*_{Z^n}[D(\cdot\|\cdot)]$$

and (11) now expresses that , with $R := \mathbf{E}^*_{Z^n}\left[\text{RISK}_{\text{emp}}(\Pi \mid Z^n, \eta)\right] - \text{RISK}(\tilde{\theta})$,

$$\mathbf{E}^*_{Z^n}[\text{RISK}(\Pi \mid Z^n, \eta)] - \text{RISK}(\tilde{\theta}) \leq C \cdot R + \frac{C}{n\eta}\mathbf{E}^*_{Z^n}[D(\ (\Pi \mid Z^n, \eta) \parallel (\Pi \mid \eta)\ )],$$

a familiar equation from the PAC-Bayesian literature: the relative risk is bounded by the empirical risk difference plus a KL-divergence penalty term. Analogous results hold in probability rather than in expectation (in many of the PAC-Bayesian literature, only in-probability results are given; Zhang (2006a, 2006b) gives both in-probability and in-expectation results).

The bounds that can be obtained via (11) are often minimax optimal. For example, if the model is correct then $\eta_{\text{CRIT}}(0) = 1$, so we can take $u = 0$. For that case Barron and Cover (1991) already showed that with appropriate choices of prior $\text{BAYES-RED}_n(1)$, (or rather its upper bound (13)) is so small that (11) leads to the optimal convergence rates in a number of nonparametric settings; Zhang (2006a) extends this to parametric models $\mathcal{P}$. If we consider 0/1-loss and a countable set of classifiers Θ, then, as is well-known, the worst-case risk obtainable by any procedure is $O((-\ln \pi(\tilde{\theta})/n)^{1/2})$ and as shown by Grünwald (2011), this is indeed the bound we get from (11) if u is chosen appropriately. Many other examples can be found in (Zhang, 2006a,b).

The key point for us is that (11) only holds for $\eta < \eta_{\text{CRIT}}(u)$; but $\eta_{\text{CRIT}}(u)$ depends on the true distribution and it is not clear how to find it. Our Theorem 1 combined with Theorem 2 imply via Corollary 1 that the safe Bayesian algorithm $\breve{W}_{\text{SAFE}}$ performs at least as well as the Bayesian posterior randomized estimator $\Pi \mid \eta$ with $\eta = \eta_{\text{CRIT}}(u)/4$. Since $\text{BAYES-RED}_n(\eta)/n$ has a bounded nonnegative derivative (as is straightforward to show), this leads to bounds that are within a constant factor of the best bound that can be obtained for any $\eta \leq \eta_{\text{CRIT}}(u)$.

In fact, Theorem 2 only shows that $\breve{W}_{\text{SAFE}}$ satisfies (11) plus an additional penalty MIX-GAP_n, which measures how much is lost in terms of cumulative log-loss by randomizing rather than mixing. Theorem 1 below shows that, for $\eta \leq \eta_{\text{CRIT}}(u)/2$, this extra penalty is sufficiently small to get the desired bound.

4 First Result: Randomizing vs. Mixing

Define the *Gibbs redundancy* as

$$\text{GIBBS-RED}_n(\eta) = \mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}_{\theta \sim \Pi \mid Z^{i-1}, \eta}\left[-\ln \frac{p_\theta(Z_i)}{q(Z_i)}\right]\right].$$

and note that, by Jensen's inequality and (10), we always have $\text{BAYES-RED}_n(\eta) \leq \text{GIBBS-RED}_n(\eta)$. The following theorem shows that, if η is sufficiently subcritical, then the reverse essentially holds as well:

Theorem 1. *Let $\eta_{\text{CRIT}}(u)$ be defined as in (7). For $0 < \eta \leq \eta_{\text{CRIT}}/2$, we have:*

$$\text{GIBBS-RED}_n(\eta) \leq C_{2\eta} \text{BAYES-RED}_n(\eta) + (C_{2\eta} - 1)\tfrac{u}{\eta}, \tag{14}$$

for a constant $C_\eta \leq 2 + 2\eta \ln V$ (so $C_{2\eta} \leq 2 + 4\eta V$) with V as in (3).

The theorem thus expresses that, in terms of log-loss, if $\eta \leq \eta_{\text{CRIT}}(u)/2$ then sequential prediction by posterior randomization is not much worse in expectation than sequential prediction by the standard Bayes predictive distribution, i.e. by mixing rather than randomizing. To explore this further, we define the *mixability gap* of a randomized estimator $\breve{W}$ as

$$\text{MIX-GAP}_n(\eta, \breve{W}) := \mathbf{E}^*_{Z^n} \left[\sum_{i=1}^n \mathbf{E}_{\theta \sim \breve{W} | Z^{i-1}} [-\ln p_\theta(Z_i)] + \tfrac{1}{\eta} \ln p_{\text{Bayes}}(Z^n \mid \eta) \right]$$

The mixability gap for the Bayesian posterior can be rewritten as:

$$\text{MIX-GAP}(\eta,\ (\Pi|\eta)\ ) = \text{GIBBS-RED}_n(\eta) - \text{BAYES-RED}_n(\eta) \geq 0. \tag{15}$$

In the information-theoretic interpretation, MIX-GAP_n is the expected amount of additional bits (additional log-loss), normalized relative to η, incurred by predicting by randomizing according to the posterior rather than by p_{Bayes}, which first mixes using the posterior and then predicts using the resulting distribution. With these definitions, (14) can be rewritten as $(\text{MIX-GAP}_n(\eta, \Pi \mid \eta) + \text{BAYES-RED}_n(\eta))$ $\leq C_{2\eta}(\text{BAYES-RED}_n(\eta) + (C_{2\eta} - 1)u/\eta$, i.e.

$$\text{MIX-GAP}_n(\eta, \Pi \mid \eta) \leq (C_{2\eta} - 1) \left(\text{BAYES-RED}_n(\eta) + \tfrac{u}{\eta} \right). \tag{16}$$

Hence, for $\eta \leq \eta_{\text{CRIT}}(u)/2$, the excess loss of randomizing rather than mixing is of the same order as the excess loss of mixing rather than predicting with q.

Example 3. [**Bayesian misspecification**] For simplicity consider Π with countable support. As shown by Grünwald and Langford (2007), if the model is incorrect, in some cases with $\eta_{\text{CRIT}}(0) \ll 1$, the standard Bayesian posterior (based on $\eta = 1$) puts, P^*-almost surely, nearly all of its mass on a set of 'bad' distributions p', all with arbitrarily large $D^*(q \| p')$ at all large n. Yet (13) shows that the redundancy, and hence the cumulative log-loss risk of standard Bayesian prediction (with $\eta = 1$) must still be small (see Example 5); this is possible because Bayes then *mixes* various 'bad' but very different $p' \in \mathcal{P}$ into a single 'good' predictive distribution $p_{\text{Bayes}}(Z_i \mid Z^{i-1}, \eta) \notin \mathcal{P}$; see Figure 1. If this happens[1] for many i between 1 and n, then by definition $\text{MIX-GAP}_n(\eta, \Pi \mid \eta)$ becomes extremely large. Theorem 1 shows that, if we set $\eta \leq \eta_{\text{CRIT}}(u)/2$, then this will not happen: the posterior $\Pi \mid Z^n, \eta$ will concentrate in the sense that if we sample from it, we will tend to draw a distribution p with $D^*(q \| p)$ close to 0, for all $q \in \mathcal{Q}$. Even if $\mathcal{Q}$ is nonsingleton this is fundamentally different from choosing $\eta = 1 \gg \eta_{\text{CRIT}}(0)/2$, in which case the posterior puts almost all of its mass on distributions p' with $D^*(q \| p')$ large for all $q \in \mathcal{Q}$. Example 5 explains why posterior concentration is so important.

[1] In the GL examples, the set of distributions over which the posterior mixes changes with sample size i, but they always remain 'bad', yet the resulting predictive distribution always remains 'good', i.e. $D^*(q \| p_{\text{Bayes}}(Z_i \mid Z^{i-1}, \eta))$ remains small.

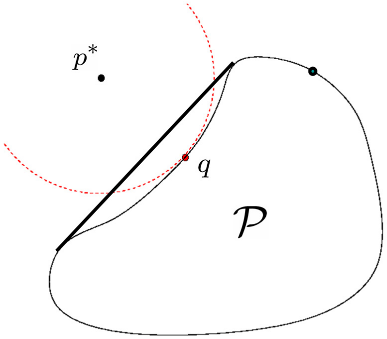


Fig. 1. Mixing vs. Randomizing: a mixture (e.g. the Bayes predictive distribution) that puts substantial mass on the two endpoints of the line segment, is closer to p^* than q, the best approximation of p^* within $\mathcal{P}$. This can only happen if q is not in the convex hull of $\mathcal{P}$. The picture is an idealization: in the GL examples the posterior mixes not just two but many 'bad' (i.e., far from p^*) distributions, $\mathcal{P}$ is not 2-dimensional parametric and the geometry is not Euclidean but determined by the KL divergence.

5 Main Result and Its Applications

We have seen that the learner would like to infer a $\hat{\eta}$ from the data that works at least as well as the unknown $\eta_{\text{CRIT}}(u)/2$. The following theorem shows that the safe Bayesian algorithm achieves this. Let $\mathcal{S}_n$ be defined as in Algorithm 1, and let $\{\breve{W}_\eta\}$ for $\eta \in \mathcal{S}_n$ represent a set of randomized estimators for Θ, one for each η. We let $\hat{\eta}$ be the "maximum likelihood" estimate of η, i.e. $\hat{\eta} \mid Z^n = \arg\min_{\eta \in \mathcal{S}_n} \sum_{i=1}^n \mathbf{E}_{\theta \sim W|Z^{i-1},\eta}[-\ln p_\theta(Z_i)]$.

Theorem 2. *Let $C_{2\eta}$ be as in Theorem 1. For $\eta \in \mathcal{S}_n$ with $\eta \leq \eta_{\text{CRIT}}(u)/2$, we have:*

$$\begin{aligned}&\mathbf{E}^*_{Z^n}\mathbf{E}_{\theta\sim\text{CES}(\breve{W}_{\hat{\eta}|Z^n};Z^n)}\left[D^*(q\|p_\theta)\right] \leq \\ &\frac{C_{2\eta}}{n}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}_{\theta\sim\breve{W}_\eta|Z^{i-1}}\left[-\ln\frac{p_\theta(Z_i)}{q(Z_i)}\right]+\frac{u+O(\ln\ln n)}{\eta}\right] = \\ &\frac{C_{2\eta}}{n}\left(\text{MIX-GAP}_n(\eta,\breve{W}_\eta)+\text{BAYES-RED}_n(\eta)+\frac{u+O(\ln\ln n)}{\eta}\right).\end{aligned} \tag{17}$$

The theorem works for any $\breve{W}_\eta$, but to get good bounds the mixability gap of $\breve{W}_\eta$ must be small. Theorem 1 tells us that it will be small if we use $\breve{W}_\eta \mid Z^n := (\Pi \mid Z^n, \eta)$, i.e. we randomize according to the posterior. If we plug this choice into (17) and rewrite the right-hand side using Theorem 1 (see (16)) and the fact that $\text{BAYES-RED}_n(\eta)$ is decreasing in η and $\eta_{\text{CRIT}}(u) \geq \eta_{\text{MIN}}(u)$ as in (8), then:

Corollary 1. *The Safe Bayesian algorithm satisfies, for $\eta \leq \eta_{\text{CRIT}}(u)/4$:*

$$\mathbf{E}^*_{Z^n}\mathbf{E}^*_{\theta\sim\breve{W}_{\text{SAFE}}|Z^n}\left[D^*(q\|p_\theta)\right] \leq \frac{C^2_{2\eta}}{n}\left(\text{BAYES-RED}_n(\eta)+\frac{u+O(\ln\ln n)}{\eta}\right). \tag{18}$$

Note that $C_{2\eta}$ has become $C_{2\eta}^2$. We got rid of the requirement that $\eta \in \mathcal{S}_n$ by using that $\text{BAYES-RED}_n(\eta)$ is decreasing in η, so that (18) must hold for every η smaller than the largest $\eta_{\max} \in \mathcal{S}_n$ with $\eta_{\max} \leq \eta_{\text{CRIT}}(u)/2$. Note that $\eta_{\max}$ may be arbitrarily close to $\eta_{\text{CRIT}}(u)/4$ rather than $\eta_{\text{CRIT}}(u)/2$.

While the bound is thus in terms of $\eta \leq \eta_{\text{CRIT}}(u)/4$, it is conceivable that the algorithm chooses a larger $\hat{\eta}$, possibly even with $\hat{\eta} \gg \eta_{\text{CRIT}}(u)$. Thus, to be fully precise, we cannot claim that we "learn" the optimal learning rate, but only that we learn to behave as well as if we would know the optimal learning rate. The second line of (17) indicates that the randomized η-posterior in Algorithm 1 may in principle be replaced by other estimators that approximate $p_{\text{Bayes}} \mid \eta$ reasonably well, such as e.g. the Bayesian MAP estimator with prior $w(\theta)^{1/\eta}$.

Example 4. [**Safe vs. Empirical Bayes**] The Safe Bayesian algorithm chooses $\hat{\eta}$ that minimizes a cumulative log-loss and hence maximizes a likelihood[2]. Indeed, if we interchanged expectation and logarithm in the definition of r in Algorithm 1, then we would mix rather than randomize and by (9), $\hat{\eta}$ would become the *empirical Bayes estimate* of η. Now for $\eta > \eta_{\text{CRIT}}(u)$, we may be in the situation of Figure 1 where our Bayesian predictive distribution achieves small cumulative log loss by mixing, at many sample sizes, bad distributions into a good one. Empirical Bayes will tend to pick such an unwanted η, and indeed, it was already shown in GL that it does not solve the Bayesian inconsistencies noted there.

Example 5. [**Bayesian Misspecification, Cont.**] The examples considered by GL are based on P^* and countable $\mathcal{P}$ such that $0 < \eta_{\text{CRIT}}(0) \ll 1$ and a prior $\pi(q) > 0$ on the best-approximating q. Using (13), Corollary 1 thus bounds the convergence rate of $\breve{W}_{\text{SAFE}}$ as $O((\ln \ln n)/n)$, only a factor $\ln \ln n$ worse compared to (11), which is the best known bound when $\eta_{\text{CRIT}}(0)$ is known in advance. Thus, the inconsistency for $\eta = 1$ goes away. Now one may wonder why one should not just, given sample Z^n, directly use the standard Bayes predictive distribution $p_{\text{Bayes}}(Z_{n+1} \mid Z^n, \eta)$ with $\eta = 1$ to make predictions about Z_{n+1}? By (10) and (13), the cumulative expected log-loss risk of this procedure should be bounded by $-\ln \pi(q)$, indicating a risk smaller than $O(1/n)$ at most n. If one is only interested in log-loss, this can indeed be done, and there is indeed no good reason to take $\eta < 1$. But in many other cases, there is a very good reason to take a smaller $\hat{\eta}$ so that Corollary 1 holds. We give one reason below; two more reasons can be found in the longer version of this paper.. Note first that the corollary implies that, for large enough n, the posterior is concentrated (see above Example 1), and the phenomenon of Figure 1 cannot occur (Ex. 3). Now for 'nonmixable' loss functions (Vovk, 1990) one cannot mix the predictors θ in a Bayesian way. For example, the examples of GL also have an interpretation in terms of 0/1-loss: they show that, in the statistical learning setting with $p_\theta(y \mid x) \propto \exp(-\text{LOSS}(y, \theta(x)))$, predicting according to the MAP, Gibbs or Bayes classifier based on the posterior $\Pi \mid Z^n, \eta$ for $\eta = 1$ lead to predictions that *never* come close to $\tilde{L} = \inf_{\theta \in \Theta} \text{RISK}(\theta)$. Yet Corollary 1 implies

[2] In fact, $\hat{\eta}$ maximizes a "prequential" likelihood, and the algorithm (not its analysis) is a prime instance of Dawid's (1984) prequential approach to statistics.

(see Example 2) that prediction based on $\breve{W}_{\text{SAFE}}$ does converge to $\tilde{L}$ at rate $O((\ln \ln n)/n)$. But now, in contrast to the log-loss case, predicting with $p_{\text{Bayes}} \mid \eta$ for $\eta = 1$ is not an option, since — as explained at length by GL07 — its predictions, which are mixtures over p_θ as above, are mixtures of exponentiated rather than randomized classifiers and hence do not correspond to feasible actions; rather, they are *pseudo-predictions* in the sense of Vovk (2001). In that case, prediction by $\breve{W}_{\text{SAFE}}$, i.e. using the learning rate $\hat{\eta}$, is presumably the best one can do.

Example 6. [**Probabilistic Setting with Correct or Convex Model** $\mathcal{P}$] Corollary 1 implies that safe Bayesian estimation behaves essentially as well as standard Bayesian estimation if the model is correct, i.e. $\inf_{p \in \mathcal{P}} D(P^* \| p) = D(P^* \| q) = 0$ and $q = p^*$. Then $\eta_{\text{CRIT}}(0) = 1$ and we can take $u = 0$ and $C_{2\eta}^2/\eta = C_2^2/1 \leq (2 + 4 \ln V)^2$ in (18). Zhang (2006a) obtains the same risk bound as (18) for $\Pi \mid \eta$ for any $\eta < 1$, the only difference being that the factor $C_2^2/1$ on the right is replaced by something smaller, and that there is no $O(\ln \ln n/\eta n)$ term. The extra factor incurred by $\breve{W}_{\text{SAFE}}$ may be the inevitable price to pay for not knowing in advance that our model was, in fact, correct, using a procedure that still leads to good results if it is incorrect.

Related Work. Grünwald (2011) already proposed an adjustment to 2-part MDL and Bayesian MAP approaches to deal with misspecified (wrong) models. The learning rate was determined in a completely different manner, roughly by measuring how much one can additionally compress the data using a code based on the convex hull of $\mathcal{P}$ rather than $\mathcal{P}$. The resulting procedure is computationally much more demanding than the Safe Bayesian algorithm. Also, it can only be applied to countable $\mathcal{P}$ — a severe restriction — whereas the Safe Bayesian algorithm can be applied to arbitrary $\mathcal{P}$. Finally, the bounds in Grünwald (2011) —although qualitatively similar to the ones presented here — have much larger constant factors ($O(V)$ instead of $O(\ln V)$ with V as in (3) above).

Acknowledgements. Supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This paper benefited from discussions with A. Barron, S. de Rooij, T. van Erven, W. Koolen, A.P. Dawid and some insightful remarks by an anonymous referee.

A Proofs

Proof of Theorem 1. Apply Lemma 1 below, with $\mathcal{G} = \Theta$, $\nu(\theta) := \theta$ and for all $z^i \in \mathcal{Z}^i$, $f_{\nu(\theta)}(z_i \mid z^{i-1}) := p_\theta(z_i)/q(z_i)$, noting that the Lemma applies for $\eta \leq \eta_{\text{CRIT}}(u)/2$. Rewriting the left-hand side using the definition of GIBBS-RED, the statement is seen to imply Theorem 1.

To prepare for Lemma 1, let $Z, Z_1, Z_2, \ldots Z_n$ be i.i.d. random variables relative to a probability triple (Ω, Σ, P^*). Let $\mathcal{G}$ be a set and, for each $\nu \in \mathcal{G}$, let $f_\nu(\cdot \mid \cdot) : \mathcal{Z} \times \bigcup_{i=0}^{n-1} \mathcal{Z}^i \rightarrow \mathbb{R}^+$ be a measurable function such that for $i \leq n$, $P^*(f_\nu(Z_i \mid Z^{i-1}) > 0) = 1$. The notation $f_\nu(Z_i \mid Z^{i-1})$ is suggestive of our applications, in which f_ν represents a ratio of conditional densities. Define $f_\nu(Z^n) := \prod_{i=1}^n f(Z_i \mid Z^{i-1})$. Let $\mathit{\Pi}$ be a prior distribution on $\mathcal{G}$ and let $\mathit{\Pi} \mid Z^i, \eta$ be the generalized posterior defined as in (1), with p_θ^η replaced by f_ν^η.

Lemma 1. *Let $C_\eta = 2 + 2\eta \ln V$ and suppose for all $z^n \in \mathcal{Z}^n$, $f_\nu(z^i \mid z^{i-1}) \in [1/V, V]$. For all $\eta > 0$ such that for all $\nu \in \mathcal{G}$, $\ln \mathbf{E}^*_Z[f_\nu^{2\eta}(Z)] \leq u/n$, we have:*

$$\begin{aligned}\mathbf{E}^*_{Z^n}\left[\sum_{i=0}^{n-1} \mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i,\eta}[-\ln f_\nu(Z_{i+1} \mid Z^i)]\right] \leq \\ \tfrac{C_{2\eta}}{\eta}\mathbf{E}^*_{Z^n}\left[-\ln \mathbf{E}_{\nu\sim\mathit{\Pi}} f_\nu^\eta(Z^n)\right] + \tfrac{C_{2\eta}-1}{\eta}u.\end{aligned} \tag{19}$$

Proof.

$$\begin{aligned}
&\mathbf{E}^*_{Z^n}\left[\textstyle\sum_{i=0}^{n-1} \mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i}[-\ln f_\nu(Z_{i+1} \mid Z^i)]\right] = \\
&\eta^{-1}\textstyle\sum_{i=0}^{n-1} \mathbf{E}^*_{Z^i}\mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i}\left[\mathbf{E}^*_{\bar{Z}_{i+1}}[-\ln f_\nu^\eta(\bar{Z}_{i+1} \mid Z^i)]\right] \leq_{\text{(a)}} \\
&\eta^{-1}\textstyle\sum_{i=0}^{n-1} \mathbf{E}^*_{Z^i}\left[C_{2\eta}\left(-\ln \mathbf{E}^*_{\bar{Z}_{i+1}}\mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i}[f_\nu^\eta(\bar{Z}_{i+1} \mid Z^i)]\right) + (C_{2\eta}-1)\left(\frac{u}{n}\right)\right] \leq \\
&\eta^{-1}\textstyle\sum_{i=0}^{n-1} \mathbf{E}^*_{Z^i}\mathbf{E}^*_{Z_{i+1}}\left[C_{2\eta}\left(-\ln \mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i}[f_\nu^\eta(Z_{i+1} \mid Z^i)]\right) + (C_{2\eta}-1)\left(\frac{u}{n}\right)\right] = \\
&\tfrac{C_{2\eta}}{\eta}\mathbf{E}^*_{Z^n}\left[\textstyle\sum_{i=0}^{n-1} -\ln \mathbf{E}_{\nu\sim\mathit{\Pi}\mid Z^i}[f_\nu^\eta(Z_{i+1} \mid Z^i)]\right] + \tfrac{C_{2\eta}-1}{\eta}u =_{\text{(c)}} \\
&\tfrac{C_{2\eta}}{\eta}\mathbf{E}^*_{Z^n}\left[-\ln \mathbf{E}_{\nu\sim\mathit{\Pi}} f_\nu^\eta(Z^n)\right] + \tfrac{C_{2\eta}-1}{\eta}u.
\end{aligned}$$

(a) follows from Lemma 2 below, applied with T set to the random vector $T = (\nu, \bar{Z}_{i+1})$ and $g((\nu, \bar{Z}_{i+1})) \equiv f_\nu^\eta(\bar{Z}_{i+1} \mid Z^i)$. The next line is Jensen's inequality, and (c) is the telescoping of the Bayesian predictive distribution as in e.g. (9). All other equalities are immediate.

Lemma 2. *Let T be a random vector taking values in some set $\mathcal{T}$. For all measurable functions $g : \mathcal{T} \to [1/V, V]$, all $\eta' > 0, \epsilon \geq 0$ with $\ln \mathbf{E}[g(T)^{2\eta'}] \leq \epsilon$, all $0 < \eta \leq \eta'$: $\mathbf{E}[-\ln g(T)] \leq \frac{C_{2\eta}}{\eta}\left(-\ln \mathbf{E}[g(T)^\eta]\right) + \frac{C_{2\eta}-1}{\eta}\epsilon$.*

This lemma slightly extends a result by Barron and Li (1999). It is proved as Proposition 5 in Grünwald (2011) (in different context, but the modification to our setting is immediate).

Proof of Theorem 2. We apply Theorem 3 below in the form (23), with $\mathcal{G}$ set to $\mathcal{S}_n$ in Theorem 2, the deterministic estimator $\breve{\nu}$ set to $\hat{\eta}$, and with $f_{\hat{\nu}_{\mid Z^n}}(z_i \mid z^{i-1})$ set to $\exp(\mathbf{E}_{\theta\sim\breve{W}_{\hat{\eta}\mid Z^n}\mid z^{i-1}}[\eta \ln(p_\theta(z_i)/q(z_i))]$. Here η is just a fixed exponent and $\hat{\eta}$ is the meta-estimator in Theorem 2 indexing the learning rate at which the randomized estimator $\breve{W}_\eta$ of Theorem 2 is applied. Plugging these substitutions into (23) using that $Z_1, Z_2, \ldots$ are i.i.d., we get

$$\begin{aligned}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n -\frac{1}{\eta}\ln \mathbf{E}^*_{\bar{Z}_i}\left[f_{\hat{\nu}_{\mid Z^n}}(\bar{Z}_i \mid Z^{i-1})\right]\right] \leq \\ \mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}_{\theta\sim W_{\hat{\eta}\mid Z^n}\mid Z^{i-1}}\left[-\ln \frac{p_\theta(Z_i)}{q(Z_i)}\right] + \frac{-\ln \pi(\hat{\eta})}{\eta}\right].\end{aligned}$$

where we also divided both sides by η. We now move the inner expectation on the left-hand side outside of the logarithm by applying Lemma 2 with $T = \bar{Z}_i$, $g^\eta(\bar{Z}_i) = f_{\hat{\nu}|Z^n}(\bar{Z}_i \mid Z^{i-1})$, using our assumption $0 < \eta < \eta_{\text{CRIT}}(u)/2$, which gives

$$\begin{array}{c}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}^*_{\bar{Z}_i}\mathbf{E}^*_{\theta\sim W_{\hat{\eta}|Z^n}|Z^{i-1}}\left[-\ln\frac{p_\theta(\bar{Z}_i)}{q(\bar{Z}_i)}\right]\right] \leq \\ \frac{C_{2\eta}}{\eta}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n -\ln \mathbf{E}^*_{\bar{Z}_i}\left[f_{\hat{\nu}|Z^n}(\bar{Z}_i \mid Z^{i-1})\right] + \frac{u}{n}\right].\end{array} \tag{20}$$

Combining the previous two equations, dividing by n and recognizing the inner expectation in the left hand side of (20) to be equal to $\mathbf{E}^*_{\theta\sim\breve{W}_{\hat{\eta}|Z^n}|Z^{i-1}}D^*(q\|p_\theta)$ gives

$$\begin{array}{c}\frac{1}{n}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}^*_{\theta\sim\breve{W}_{\hat{\eta}|Z^n}|Z^{i-1}}[D^*(q\|p_{\breve{W}_\theta})]\right] \leq \\ \frac{C_{2\eta}}{n}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n \mathbf{E}_{\theta\sim\breve{W}_{\hat{\eta}|Z^n}|Z^{i-1}}\left[-\ln\frac{p_\theta(Z_i)}{q(Z_i)}\right] + \frac{u-\ln\pi(\hat{\eta}|Z^n)}{\eta}\right]\end{array} \tag{21}$$

The left side is equal to $\mathbf{E}^*_{Z^n}\left[D^*(q\|p_{\text{CES}(\breve{W}_{\hat{\eta}|Z^n};Z^n)})\right]$. We now take π to be the uniform prior on $\mathcal{S}_n$, so that for all $\eta \in \mathcal{S}_n$, $-\ln\pi(\eta) = \ln\|\mathcal{S}_n\| = \ln\|\kappa_{\max}+1\| = O(\ln\ln n)$. The result now follows from (21), noting that the right-hand side increases if we replace $\hat{\eta} \mid Z^n$ by $\eta \in \mathcal{S}_n$.

Towards Theorem 3. We extend an inequality which, in various guises and level of detail, was proven earlier by M. Seeger (2002), D. McAllester (2003), O. Catoni (2007), J.Y. Audibert (2004) (in the context of PAC-Bayesian inference; see Zhang (2006b) for references to additional relevant papers by these authors), and A. Barron (with T. Cover (1991) and with J. Li (1999)), and T. Zhang (2006a,b) in the context of MDL-type inference. Our version is a direct extension of Theorem 2.1. of Zhang (2006b). Let $Z_1, Z_2, \ldots$, P^*, f_ν and $\mathcal{G}$ be as above Lemma 1, except that now we do *not* require $Z_1, Z_2, \ldots$ to be i.i.d. All earlier guises of Zhang's result assumed that $Z_1, \ldots, Z_n$ are i.i.d. both according to the 'true' P^* and according to all 'densities' $f_\nu(Z_i \mid z^{i-1})$, which were not allowed to depend on z^{i-1}. Our application of the inequality to prove Theorem 2 requires us to extend it to non-i.i.d. models (represented below by $f_\nu(Z_i \mid z^{i-1})$ which vary with z^{i-1}). As a by-product, we also extend it to non-i.i.d. Z_i (in principle this should allow us to extend the in this paper to some non-i.i.d. misspecification settings as considered by Shalizi (2009)). The result compares the expectation of $Z_i \mid Z^{i-1}$ to its actually observed value, and then takes another expectation over the values that can actually be observed. To ease readability, we denote the Z_i in the inner expectation as $\bar{Z}_i$.

Theorem 3. [Extended Zhang's Inequality] *For arbitrary $\mathcal{G}$, let Π be a ("prior") distribution on $\mathcal{G}$ and let $\breve{W} : \bigcup_{n\geq 0}\mathcal{Z}^n \to \mathcal{G}$ be a randomized estimator. Then, with $D(\cdot\|\cdot)$ denoting KL divergence, we have:*

$$\begin{array}{c}\mathbf{E}^*_{Z^n}\mathbf{E}_{\nu\sim\breve{W}|Z^n}\left[\sum_{i=1}^n -\ln\mathbf{E}^*_{\bar{Z}_i|Z^{i-1}}[f_\nu(\bar{Z}_i \mid Z^{i-1})]\right] \leq \\ \mathbf{E}^*_{Z^n}\left[\mathbf{E}_{\nu\sim\breve{W}|Z^n}\left[\sum_{i=1}^n -\ln f_\nu(Z_i \mid Z^{i-1})\right] + D(\,(\breve{W}|Z^n)\,\|\Pi)\right].\end{array} \tag{22}$$

As a special case, suppose $\mathcal{G}$ is countable, π is a probability mass function on $\mathcal{G}$, and $\breve{\nu}$ is a deterministic estimator. Then

$$\begin{aligned}\mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n -\ln \mathbf{E}^*_{\bar{Z}_i|Z^{i-1}}[f_{\breve{\nu}|Z^n}(\bar{Z}_i \mid Z^{i-1})]\right] \leq \\ \mathbf{E}^*_{Z^n}\left[\sum_{i=1}^n[-\ln f_{\breve{\nu}|Z^n}(Z_i \mid Z^{i-1})] - \ln \pi(\breve{\nu})\right].\end{aligned} \tag{23}$$

The proof is in the longer version of this paper.

References

Audibert, J.Y.: PAC-Bayesian statistical learning theory. PhD thesis, Université Paris VI (2004)

Barron, A.R., Cover, T.M.: Minimum complexity density estimation. IEEE Transactions on Information Theory 37(4), 1034–1054 (1991)

Catoni, O.: PAC-Bayesian Supervised Classification. Lecture Notes IMS (2007)

Chaudhuri, K., Freund, Y., Hsu, D.: A parameter-free hedging algorithm. In: NIPS 2009, pp. 297–305 (2009)

Dawid, A.P.: Present position and potential developments: Some personal views, statistical theory, the prequential approach. J. R. Stat. Soc. Ser. A-G 147(2), 278–292 (1984)

Doob, J.L.: Application of the theory of martingales. In: Le Calcul de Probabilités et ses Applications. Colloques Internationaux du CNRS, pp. 23–27 (1949)

Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55, 119–139 (1997)

Grünwald, P.: The MDL Principle. MIT Press, Cambridge (2007)

Grünwald, P.: Safe learning: bridging the gap between Bayes, MDL and statistical learning theory via empirical convexity. In: Proc. COLT 2011, pp. 551–573 (2011)

Grünwald, P., Langford, J.: Suboptimal behavior of Bayes and MDL in classification under misspecification. Machine Learning 66(2-3), 119–149 (2007)

Kleijn, B., van der Vaart, A.: Misspecification in infinite-dimensional Bayesian statistics. Ann. Stat. 34(2) (2006)

Kullback, S., Leibler, R.A.: On information and sufficiency. Ann. Math. Stat. 22, 76–86 (1951)

Li, J.Q.: Estimation of Mixture Models. PhD thesis, Yale, New Haven, CT (1999)

McAllester, D.: PAC-Bayesian stochastic model selection. Mach. Learn. 51(1), 5–21 (2003)

Jordan, M.I., Bartlett, P.L., McAuliffe, J.D.: Convexity, classification and risk bounds. J. Am. Stat. Assoc. 101(473), 138–156 (2006)

Seeger, M.: PAC-Bayesian generalization error bounds for Gaussian process classification. J. Mach. Learn. Res. 3, 233–269 (2002)

Shalizi, C.: Dynamics of Bayesian updating with dependent data and misspecified models. Electronic Journal of Statistics 3, 1039–1074 (2009)

Takeuchi, J., Barron, A.R.: Robustly minimax codes for universal data compression. In: Proc. ISITA 1998, Japan (1998)

van der Vaart, A.: Asymptotic Statistics. Cambridge University Press (1998)

Vovk, V.: Competitive on-line statistics. Intern. Stat. Rev. 69, 213–248 (2001)

Vovk, V.: Aggregating strategies. In: Proc. COLT 1990, pp. 371–383 (1990)

Zhang, T.: From ϵ-entropy to KL entropy: analysis of minimum information complexity density estimation. Ann. Stat. 34(5), 2180–2210 (2006a)

Zhang, T.: Information theoretical upper and lower bounds for statistical estimation. IEEE T. Inform. Theory 52(4), 1307–1321 (2006b)

Data Stability in Clustering: A Closer Look

Lev Reyzin[*]

School of Computer Science,
Georgia Institute of Technology,
266 Ferst Drive,
Atlanta, GA 30332
lreyzin@cc.gatech.edu

Abstract. We consider the model introduced by Bilu and Linial [12], who study problems for which the optimal clustering does not change when distances are perturbed. They show that even when a problem is NP-hard, it is sometimes possible to obtain efficient algorithms for instances resilient to certain multiplicative perturbations, e.g. on the order of $O(\sqrt{n})$ for max-cut clustering. Awasthi et al. [6] consider center-based objectives, and Balcan and Liang [9] analyze the k-median and min-sum objectives, giving efficient algorithms for instances resilient to certain constant multiplicative perturbations.

Here, we are motivated by the question of to what extent these assumptions can be relaxed while allowing for efficient algorithms. We show there is little room to improve these results by giving NP-hardness lower bounds for both the k-median and min-sum objectives. On the other hand, we show that multiplicative resilience parameters, even only on the order of $\Theta(1)$, can be so strong as to make the clustering problem trivial, and we exploit these assumptions to present a simple one-pass streaming algorithm for the k-median objective. We also consider a model of additive perturbations and give a correspondence between additive and multiplicative notions of stability. Our results provide a close examination of the consequences of assuming, even constant, stability in data.

1 Introduction

Clustering is one of the most widely-used techniques in statistical data analysis. The need to partition, or cluster, data into meaningful categories naturally arises in virtually every domain where data is abundant. Unfortunately, most of the natural clustering objectives, including k-median, k-means, and min-sum, are NP-hard to optimize [17,19]. It is, therefore, unsurprising that many of the clustering algorithms used in practice come with few guarantees.

Motivated by overcoming the hardness results, Bilu and Linial [12] consider a perturbation **resilience assumption** that they argue is often implicitly made when choosing a clustering objective: that the optimum clustering to the desired objective Φ is preserved under multiplicative perturbations up to a factor $\alpha > 1$

[*] Supported by a Simons Postdoctoral Fellowship in Theoretical Computer Science.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 184–198, 2012.

to the distances between the points. They reason that if the optimum clustering to an objective Φ is not resilient, as in, if small perturbations to the distances can cause the optimum to change, then Φ may have been the wrong objective to be optimizing in the first place. Bilu and Linial [12] show that for max-cut clustering, instances resilient to perturbations of $\alpha = O(\sqrt{n})$ have efficient algorithms for recovering the optimum itself.

Continuing that line of research, Awasthi et al. [6] give a polynomial time algorithm that finds the optimum clustering for instances resilient to multiplicative perturbations of $\alpha = 3$ for center-based[1] clustering objectives when centers must come from the data (we call this the **proper** setting), and $\alpha = 2 + \sqrt{3}$ when when the centers do not need to (we call this the **Steiner** setting). Their method relies on a **stability** property implied by perturbation resilience (see Section 2). For the Steiner case, they also prove an NP-hardness lower bound of $\alpha = 3$. Subsequently, Balcan and Liang [9] consider the proper setting and improve the constant past $\alpha = 3$ by giving a new polynomial time algorithm for the k-median objective for $\alpha = 1 + \sqrt{2} \approx 2.4$ stable instances.

1.1 Our Results

Our work further delves into the proper setting, for which no lower bounds have previously been shown for the stability property. In Section 3 we show that even in the proper case, where the algorithm is restricted to choosing its centers from the data, for any $\epsilon > 0$, it is NP-hard to optimally cluster $(2-\epsilon)$-stable instances, both for the **k-median** and **min-sum** objectives (Theorems 1 and 2). To prove this for the min-sum objective, we define a new notion of stability that is implied by perturbation resilience, a notion that may be of independent interest.

Then in Section 4, we look at the implications of assuming resilience or stability in the data, even for a constant perturbation parameter α. We show that for even fairly small constants, the data begins to have very strong structural properties, as to make the clustering task fairly trivial. When α approaches ≈ 5.7, the data begins to show what is called **strict separation**, where each point is closer to points in its own cluster than to points in other clusters (Theorem 3). We show that with strict separation, optimally clustering in the very restrictive one-pass streaming model becomes possible (Theorem 4).

Finally, in Section 5, we look at whether the picture can be improved for clustering data that is stable under additive, rather than multiplicative, perturbations. One hope would be that **additive stability** is a more useful assumption, where a polynomial time algorithm for ϵ-stable instances might be possible. Unfortunately, this is not the case. We consider a natural additive model and show that severe lower bounds hold for the additive notion as well (Theorems 5 and 6). On the positive side, we show via reductions that algorithms for multiplicatively stable data also work for additively stable data for a different but related parameter.

[1] For center-based clustering objectives, the clustering is defined by a choice of centers, and the objective is a function of the distances of the points to their closest center.

Our results demonstrate that on the one hand, it is hard to improve the algorithms to work for low stability constants, and that on the other hand, higher stability constants can be quite strong, to the point of trivializing the problem. Furthermore, switching from a multiplicative to an additive stability assumption does not help to circumvent the hardness results, and perhaps makes matters worse. These results, taken together, narrow the range of interesting parameters for theoretical study and highlight the strong role that the choice of constant plays in stability assumptions.

One thing to note that there is some difference between the very related resilience and stability properties (see Section 2), stability being weaker and more general [6]. Some of our results apply to both notions, and some only to stability. This still leaves open the possibility of devising polynomial-time algorithms that, for a much smaller α, work on all the α-perturbation resilient instances, but not on all α-stable ones.

1.2 Previous Work

The classical approach in theoretical computer science to dealing with the worst-case NP-hardness of clustering has been to develop efficient approximation algorithms for the various clustering objectives [3,4,10,13,20,15], and significant efforts have been exerted to improve approximation ratios and to prove lower bounds. In particular, for metric k-median, the best known guarantee is a $(3+\epsilon)$-approximation [4], and the best known lower bound is $(1+1/e)$-hardness of approximation [17,19]. For metric min-sum, the best known result is a $O(\text{polylog}(n))$-approximation to the optimum [10].

In contrast, a more recent direction of research has been to characterize under what conditions we can find a desirable clustering efficiently. Perturbation resilience/stability are such conditions, but they are related to other stability notions in clustering. Ostrovsky et al. [23] demonstrate the effectiveness of Lloyd-type algorithms [21] on instances with the stability property that the cost of the optimal k-means solution is small compared to the cost of the optimal $(k-1)$-means solution, and their guarantees have later been improved by Awasthi et al. [5].

In a different line of work, Balcan et al. [8] consider what stability properties of a similarity function, with respect to the ground truth clustering, are sufficient to cluster well. In a related direction, Balcan et al. [7] argue that, for a given objective Φ, approximation algorithms are most useful when the clusterings they produce are structurally close to the optimum originally sought in choosing to optimize Φ in the first place. They then show that, for many objectives, if one makes this assumption explicit – that all c-approximations to the objective yield a clustering that is ϵ-close to the optimum – then one can recover an ϵ-close clustering in polynomial time, even for values of c below the hardness of approximation constant. The assumptions and algorithms of Balcan et al. [7] have subsequently been carefully analyzed by Schalekamp et al. [24].

Ackerman and Ben-David [1] also study various notions of stability, and among their results, introduce a notion where only the positions of cluster centers are perturbed. They show that instances stable in this manner will have polynomial algorithms for finding near-optimal clusterings.

2 Notation and Preliminaries

In a clustering instance, we are given a set S of n points in a finite metric space, and we denote $d : S \times S \to \mathbb{R}_{\geq 0}$ as the distance function. Φ denotes the objective function over a partition of S into k clusters which we want to optimize over the metric, i.e. Φ assigns a score to every clustering. The optimal clustering with respect to Φ is denoted as $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$.

The k**-median objective** requires S to be partitioned into k disjoint subsets $\{S_1, \ldots, S_k\}$ and each subset S_i to be assigned a center $s_i \in S$. The goal is to minimize Φ_{med}, measured by

$$\phi_{\mathrm{med}}(S_1, \ldots, S_k) \doteq \sum_{i=1}^{k} \sum_{p \in S_i} d(p, s_i).$$

The centers in the optimal clustering are denoted as $c_1, \ldots, c_k$. In an optimal solution, each point is assigned to its nearest center. For the **min-sum objective**, S is partitioned into k disjoint subsets, and the objective is to minimize $\Phi_{\mathrm{m-s}}$, measured by

$$\phi_{\mathrm{m-s}}(S_1, \ldots, S_k) \doteq \sum_{i=1}^{k} \sum_{p,q \in S_i} d(p, q).$$

Now, we define the perturbation resilience notion introduced by Bilu and Linial [12].

Definition 1. *For $\alpha > 1$, a clustering instance (S, d) is* ***α-perturbation resilient*** *to a given objective Φ if for any function $d' : S \times S \to \mathbb{R}_{\geq 0}$ such that $\forall p, q \in S$,*

$$d(p, q) \leq d'(p, q) \leq \alpha d(p, q),$$

there is a unique optimal clustering $\mathcal{C}'$ for Φ under d' and this clustering is equal to the optimal clustering $\mathcal{C}$ for Φ under d.

In this paper, we consider the k-median and min-sum objectives, and we thereby investigate the following definitions of stability, which are implied by perturbation resilience, as shown in Sections 3.1 and 3.2. The following definition is adapted from Awasthi et al. [6].

Definition 2. *A clustering instance (S, d) is* ***α-center stable*** *for the k-median objective if for any optimal cluster $C_i \in \mathcal{C}$ with center c_i, $C_j \in \mathcal{C}$ $(j \neq i)$ with center c_j, any point $p \in C_i$ satisfies $\alpha d(p, c_i) < d(p, c_j)$.*

Next, we define a new analogous notion of stability for the min-sum objective, and we show in Section 3.2 that for the min-sum objective, perturbation resilience implies min-sum stability. To help with exposition for the min-sum objective, we define the distance from a point p to a set of points A,

$$d(p,A) \doteq \sum_{q\in A} d(p,q).$$

Definition 3. *A clustering instance (S,d) is **α-min-sum stable** for the min-sum objective if for all optimal clusters $C_i, C_j \in \mathcal{C}$ $(j \neq i)$, any point $p \in C_i$ satisfies $\alpha d(p,C_i) < d(p,C_j)$.*

This is an especially useful generalization because algorithms working under the perturbation resilience assumption often also work for min-sum stability.

3 Lower Bounds

3.1 The k-Median Objective

Awasthi et al. [6] prove the following connection between perturbation resilience and stability. Both their algorithms and the algorithms of Balcan and Liang [9] crucially use this stability assumption.

Lemma 1. *Any clustering instance that is α-perturbation resilient for the k-median objective also satisfies the α-center stability.*

Awasthi et al. [6] proved that for $\alpha < 3 - \epsilon$, k-median clustering α-center stable instances is NP-hard when Steiner points are allowed in the data. Afterwards, Balcan and Liang [9] circumvented this lower bound and achieved a polynomial time algorithm for $\alpha = 1 + \sqrt{2}$ by assuming the algorithm must choose cluster centers from within the data.

In the theorem below, we prove a lower bound for the center stable property in this more restricted setting, showing there is little hope of progress even for data where each point is nearly twice closer to its own center than to any other.

Theorem 1. *For any $\epsilon > 0$, the problem of solving $(2-\epsilon)$-center stable k-median instances is NP-hard.*

Proof. We reduce from the perfect dominating set promise problem, which we prove to be NP-hard (see Appendix), where we are promised that the input graph $G = (V,E)$ is such that all of its smallest dominating sets D are perfect, and we are asked to find a dominating set of size at most d. The reduction is simple. We take an instance of the NP-hard problem PDS-PP on $G = (V,E)$ on n vertices and reduce it to an $\alpha = 2 - \epsilon$-center stable instance. Our distance metric as follows. Every vertex $v \in V$ becomes a point in the k-center instance. For any two vertices $(u,v) \in E$ we define $d(u,v) = 1/2$. When $(u,v) \notin E$, we set $d(u,v) = 1$. This trivially satisfies the triangle inequality for any graph G, as the sum of the distances along any two edges is at least 1. We set $k = d$.

We observe that a k-median solution of cost $(n-k)/2$ corresponds to a dominating set of size d in the PDS-PP instance, and is therefore NP-hard to find. We also observe that because all solutions of size $\leq d$ in the PDS-PP instance are perfect, each (non-center) point in the k-median solution has distance $1/2$ to exactly one (its own) center, and a distance of 1 to every other center. Hence, this instance is $\alpha = (2-\epsilon)$-center stable, completing the proof. □

3.2 The Min-Sum Objective

Analogously to Lemma 1, we can show that α-perturbation resilience implies our new notion of α-min-sum stability.

Lemma 2. *If a clustering instance is α-perturbation resilient, then it is also α-min-sum stable.*

Proof. Assume to the contrary that the instance is α-perturbation resilient but is not α-min-sum stable. Then, there exist clusters C_i, C_j in the optimal solution $\mathcal{C}$ and a point $p \in C_i$ such that $\alpha d(p, C_i) \geq d(p, C_j)$. We perturb d as follows. We define d' such that for all points $q \in C_i$, $d'(p,q) = \alpha d(p,q)$, and for the remaining distances, $d' = d$. Clearly d' is an α-perturbation of d.

We now note that $\mathcal{C}$ is not optimal under d'. Namely, we can create a cheaper solution $\mathcal{C}'$ that assigns point p to cluster C_j, and leaves the remaining clusters unchanged, which contradicts optimality of $\mathcal{C}$. This shows that $\mathcal{C}$ is not the optimum under d' which contradicts the instance being α-perturbation resilient. Therefore we can conclude that if a clustering instance is α-perturbation resilient, then must also be α-min-sum stable. □

Moreover, we show (see Appendix) that the min-sum algorithm of Balcan and Liang [9], which requires α to be bounded from below by $3\left(\frac{\max_{C \in \mathcal{C}} |C|}{\min_{C \in \mathcal{C}} |C|-1}\right)$, works with this more general condition. This further motivates following bound.

Theorem 2. *For any $\epsilon > 0$, the problem of finding an optimal min-sum k clustering in $(2-\epsilon)$-min-sum stable instances is NP-hard.*

Proof. Consider the **triangle partition problem**. Let graph $G = (V, E)$ and $|V| = n = 3k$, and let each vertex have maximum degree of $d = 4$. The problem of whether the vertices of G can be partitioned into sets $V_1, V_2, \ldots, V_k$ such that each V_i contains a triangle in G is NP-complete [16], even with the degree restriction [25].

We reduce the triangle partition problem to an $\alpha = (2-\epsilon)$-min-sum stable clustering instance. The metric is as follows. Every vertex $v \in V$ becomes a point in the min-sum instance. For any two vertices $(u, v) \in E$ we define $d(u, v) = 1/2$. When $(u, v) \notin E$, we set $d(u, v) = 1$. This satisfies the triangle inequality for any graph, as the sum of the distances along any two edges is at least 1.

Now we show that we can cluster this instance into k clusters such that the cost of the min-sum objective is exactly n iff the original instance is a YES instance of triangle partition. This follows from two facts.

1. A YES instance of triangle partition maps to a clustering into $k = n/3$ clusters of size 3 with pairwise distances 1/2, for a total cost of n
2. A cost of n is the best achievable because a balanced clustering with all minimum pairwise intra-cluster distances is optimal.

In the clustering from our reduction, each point has a sum-of-distances to its own cluster of 1. Now we examine the sum-of-distances of any point to other clusters. A point has two distances of 1/2 (edges) to its own cluster, and because $d = 4$, it can have at most two more distances of 1/2 (edges) into any other cluster, leaving the third distance to the other cluster to be 1, yielding a total cost of ≥ 2 into any other cluster. Hence, it is $\alpha = (2 - \epsilon)$-min-sum stable. □

We note that it is tempting to restrict the degree bound to 3 in order to further improve the lower bound. Unfortunately, the triangle partition problem on graphs of maximum degree 3 is polynomial-time solvable [25], and we cannot improve the factor of $2 - \epsilon$ by restricting to graphs of degree 3 in this reduction.

4 Strong Consequences of Stability

In Section 3, we showed that k-median clustering even $(2 - \epsilon)$-center stable instances is NP-hard. In this section we show that even for resilience to constant multiplicative perturbations of $\alpha > \frac{1}{2}(5+\sqrt{41}) \approx 5.7$, the data obtains a property referred to as **strict separation**, where all points are closer to all other points in their own cluster than to points in any other cluster; this property is known to be helpful in clustering [8]. Then we show that this property renders center-based clustering fairly trivial even in the difficult one-pass streaming model.

4.1 Strict Separation

We will make use of the following lemma, whose proof follows directly from the triangle inequality.

Lemma 3. *For any two points p and p' belonging to different centers c_i and c_j in the optimal clustering of an α-center stable instance, $d(c_i, p') > \frac{\alpha(\alpha-1)}{\alpha+1} d(c_i, p)$.*

Now we can prove the following theorem, which shows that even for relatively small multiplicative constants for α, center stable, and therefore perturbation resilient, instances exhibit strict separation.

Theorem 3. *Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the optimal clustering of a $\frac{1}{2}(5 + \sqrt{41})$-center stable instance. Let $p, p' \in C_i$ and $q \in C_j$ $(i \neq j)$, then $d(p, q) > d(p, p')$.*

Proof. Let $\{c_1, \ldots, c_k\}$ be the centers of clusters $\{C_1, \ldots, C_k\}$. Define

$$p_f \doteq \arg\max_{r \in C_i} d(p, r).$$

By Lemma 3 we have

$$d(c_i, q) > \frac{\alpha(\alpha-1)}{\alpha+1} d(c_i, p) \quad \text{and also} \quad d(c_i, q) > \frac{\alpha(\alpha-1)}{\alpha+1} d(c_i, p_f).$$

Adding the two gives us

$$\frac{\alpha(\alpha-1)}{\alpha+1} d(c_i, p) + \frac{\alpha(\alpha-1)}{\alpha+1} d(c_i, p_f) < 2d(c_i, q),$$

and by the triangle inequality, we get

$$\frac{\alpha(\alpha-1)}{\alpha+1} d(p, p_f) < 2d(c_i, q). \tag{1}$$

We also have

$$d(c_i, q) \leq d(p, c_i) + d(p, q). \tag{2}$$

Combining Equations 1 and 2, and by the definition of p_f, we have

$$\frac{\alpha(\alpha-1)}{\alpha+1} d(p, p_f) \quad < \quad 2d(p, c_i) + 2d(q, p) \quad \leq \quad 2d(p, p_f) + 2d(q, p).$$

From the RHS and LHS of the above, it follows that

$$d(p, q) \quad > \quad \left(\frac{\alpha(\alpha-1)}{2(\alpha+1)} - 1\right) d(p, p_f) \quad \geq \quad \left(\frac{\alpha(\alpha-1)}{2(\alpha+1)} - 1\right) d(p, p'). \tag{3}$$

Equation 3 follows from the definitions of p_f and p'. Finally, the statement of the Lemma follows by setting $\alpha \geq \frac{1}{2}(5 + \sqrt{41}) \approx 5.7$. □

4.2 Clustering in the Streaming Model

Here, we turn to the restrictive **one-pass streaming** model. In the natural streaming model for center-based objectives, the learner sees the data $p_1, p_2, \ldots$ in one pass, and must, using limited memory and time, implicitly cluster the data by retaining k points to use as centers.

The clustering is then the one induced by placing each point in the cluster to the closest center produced by the algorithm. We note that a streaming algorithm can be used for the general batch problem, as one can present the data to the algorithm in a streaming fashion.

Streaming models have been extensively studied in the context of clustering objectives [2,14,18,22], where the known approximation guarantees are weaker than in the standard offline model. We, however, show that an α-center stability assumption can make the problem of finding the optimum tractable for center-based objectives, in only one pass. We view this not so much as an advance in the state-of-the-art in clustering, but rather as an illustration of how powerful stability assumptions can be, even for constant parameter values.

For our result, we can use Theorem 3 to immediately give us the following.

Corollary 1. *Let $\mathcal{C} = \{C_1, \ldots, C_k\}$ be the optimal clustering of a $\frac{1}{2}(5 + \sqrt{41})$-center stable instance. Any algorithm that chooses centers $\{c'_1, \ldots, c'_k\}$ such that $c'_i \in C_i$ induces the partition $\mathcal{C}$ when points are assigned to their closest centers.*

This leads to an algorithm that easily and efficiently finds the optimal clustering.

Theorem 4. *For $\frac{1}{2}(5+\sqrt{41})$-center stable instances, we can recover the optimal clustering for the k-median objective, even in one pass in the streaming model.*

Proof. Consider Algorithm 1. It proceeds as follows: it keeps k centers, and whenever a new point comes in, it adds it as a center and removes some point that realizes the argmin distance among the current centers.

Algorithm 1. A streaming algorithm for $\frac{1}{2}(5 + \sqrt{41})$-center stable instances

```
let $p_1, p_2, \ldots$ be the stream of points
let $C$ be a set of candidate centers, initialized $C = \{p_1, \ldots, p_k\}$
while there is more data in stream do
    receive point $p_i$
    $C = C \cup p_i$
    let $p \in \arg\min_{\{p_j, p_k\} \in C} d(p_j, p_k)$
    $C = C \setminus p$
end while
return $C$ (thereby inducing a clustering $\mathcal{C}$)
```

The correctness of this algorithm follows from two observations:

1. A pair in any $k + 1$ points belong to the same cluster (pigeonhole principle).
2. 2 points in different clusters cannot realize the argmin distance (Theorem 3).

Hence, whenever a point is removed as a candidate center, it has a partner in the same optimal cluster that remains. Once we see a point from each cluster, by Corollary 1, we get the optimal partition. □

5 Additive Stability

So far, in this paper our notions of stability were defined with respect to multiplicative perturbations. Similarly, we can imagine an instance being resilient with respect to additive perturbations. Consider the following definition.

Definition 4. *Let $d : S \times S \to [0, 1]$, and let $0 < \beta \leq 1$. A clustering instance (S, d) is* ***additive** β**-perturbation*** *resilient to a given objective Φ if for any function $d' : S \times S \to R \geq 0$ such that $\forall p, q \in S$, $d(p, q) \leq d'(p, q) \leq d(p, q) + \beta$, there is a unique optimal clustering $\mathcal{C}'$ for Φ under d' and this clustering is equal to the optimal clustering $\mathcal{C}$ for Φ under d.*

We note that in the definition above, we require all pairwise distances between points to be at most 1. Otherwise, resilience to additive perturbations would be a very weak notion, as the distances in most instances could be scaled as to be resilient to arbitrary additive perturbations.

Especially in light of positive results for other additive stability notions [1,11], one possible hope is that our hardness results might only apply to the multiplicative case, and that we might be able to get polynomial time clustering algorithms for instances resilient to arbitrarily small additive perturbations. We show that this is unfortunately not the case – we introduce notions of additive stability, similar to Definitions 2 and 3, and for the k-median and min-sum objectives, we show correspondences between multiplicative and additive stability.

5.1 The k-Median Objective

Analogously to Definition 2, we can define a notion of additive β-center stability.

Definition 5. *Let $d : S \times S \to [0,1]$, and let $0 \leq \beta \leq 1$. A clustering instance (S,d) is **additive β-center stable** to the k-median objective if for any optimal cluster $C_i \in \mathcal{C}$ with center c_i, $C_j \in \mathcal{C}$ $(j \neq i)$ with center c_j, any point $p \in C_i$ satisfies $d(p,c_i) + \beta < d(p,c_j)$.*

We can now prove that perturbation resilience implies center stability.

Lemma 4. *Any clustering instance satisfying additive β-perturbation resilience for the k-median objective also satisfies additive β-center stability.*

Proof. The proof is similar to that of Lemmas 1 and 2 – see the Appendix. □

We now consider center stability, as in the multiplicative case. We first prove that additive center stability implies multiplicative center stability, and this gives us the property that any algorithm for $\left(\frac{1}{1-\beta}\right)$-center stable instances will work for additive β-center stable instances.

Lemma 5. *Any additive β-center stable clustering instance for the k-median objective is also (multiplicative) $\left(\frac{1}{1-\beta}\right)$-center stable.*

Proof. Let the optimal clustering be $C_1, \ldots, C_k$, with centers $c_1, \ldots, c_k$, of an additive β-center stabile clustering instance. Let $p \in C_i$ and let $i \neq j$. From the stability property,

$$d(p,c_j) > d(p,c_i) + \beta \geq \beta. \tag{4}$$

We also have $d(p,c_i) < d(p,c_j) - \beta$, from which we can see

$$\frac{1}{d(p,c_j) - \beta} < \frac{1}{d(p,c_i)}.$$

This gives us

$$\frac{d(p,c_j)}{d(p,c_i)} > \frac{d(p,c_j)}{d(p,c_j) - \beta} \geq \frac{1}{1-\beta}. \tag{5}$$

Equation 5 is derived as follows. The middle term, for $d(p, c_j) \geq \beta$ (which we have from Equation 4), is monotonically decreasing in $d(p, c_j)$. Using $d(p, c_j) \leq 1$ bounds it from below. Relating the LHS to the RHS of Equation 5 gives us the needed stability property. □

Now we prove a lower bound that shows that the task of clustering additive $(1/2 - \epsilon)$-center stable instances w.r.t. the k-median objective remains NP-hard.

Theorem 5. *For any $\epsilon > 0$, the problem of finding an optimal k-median clustering in additive $(1/2 - \epsilon)$-center stable instances is NP-hard.*

Proof. We use the reduction in Theorem 1, in which the metric satisfies the needed property that $d : S \times S \to [0, 1]$. We observe that the instances from the reduction are additive $(1/2 - \epsilon)$-center stable. Hence, an algorithm for solving k-median on a $(1/2 - \epsilon)$-center stable instance can decide whether a PDS-PP instance contains a dominating set of a given size, completing the proof. □

5.2 The Min-Sum Objective

Here we define additive min-sum stability and prove the analogous theorems for the min-sum objective.

Definition 6. *Let $d : S \times S \to [0, 1]$, and let $0 \leq \beta \leq 1$. A clustering instance is **additive β-min-sum stable** for the min-sum objective if for every point p in any optimal cluster C_i, it holds that $d(p, C_i) + \beta(|C_i| - 1) < d(p, C_j)$.*

Lemma 6. *If a clustering instance is additive β-perturbation resilient, then it is also additive β-min-sum stable.*

Proof. The proof appears in the Appendix. □

As we did for the k-median objective, we can also reduce additive stability to multiplicative stability for the min-sum objective.

Lemma 7. *Let $t = \frac{\max_{C \in \mathcal{C}} |C|}{\min_{C \in \mathcal{C}} |C| - 1}$. Any additive β-min-sum stabile clustering instance for the min-sum objective is also (multiplicative) $\left(\frac{1}{1 - \beta/t}\right)$-min-sum stable.*

Proof. Let the optimal clustering be $C_1, \ldots, C_k$ and let $p \in C_i$. Let $i \neq j$. From the stability property, we have

$$d(p, C_j) > d(p, C_i) + \beta(|C_i| - 1) \geq \beta(|C_i| - 1). \tag{6}$$

We also have

$$d(p, C_i) < d(p, C_j) - \beta(|C_i| - 1).$$

Taking reciprocals and multiplying by $d(p, C_j)$, we have

$$\frac{d(p,C_j)}{d(p,C_i)} > \frac{d(p,C_j)}{d(p,C_j)-\beta(|C_i|-1)} \geq \frac{|C_j|}{|C_j|-\beta(|C_i|-1)} \tag{7}$$

$$\geq \frac{\max_{C\in\mathcal{C}}|C|}{\max_{C\in\mathcal{C}}|C_j|-\beta(\min_{C\in\mathcal{C}}|C|-1)} \geq \frac{1}{1-\beta/t}. \tag{8}$$

The LHS of Equation 7 is derived as follows: the middle term, for $d(p,C_j) \geq \beta(|C_i|-1)$ (which we have from Equation 6), is monotonically decreasing in $d(p,C_j)$. Using $d(p,c_j) \leq |C_j|$ bounds it from below. Equation 8 gives us the needed property. □

Finally, as with the k-median objective, we show that additive min-sum stability exhibits similar lower bounds as in the multiplicative case.

Theorem 6. *For any $\epsilon > 0$, the problem of finding an optimal min-sum clustering in additive $(1/2-\epsilon)$-min-sum stable instances is NP-hard.*

Proof. We use the reduction in Theorem 2, in which the metric satisfies the property that $d : S \times S \to [0,1]$. The instances from the reduction are additive $(1/2-\epsilon)$-min-sum stable. Hence, an algorithm for clustering a $(1/2-\epsilon)$-min-sum stable instance can solve the triangle partition problem. □

6 Discussion

Our lower bounds, together with the structural properties implied by fairly small constants, illustrate the importance parameter settings play in stability assumptions. These results make us wonder the degree to which the assumptions studied herein hold in practice; empirical study of real datasets is warranted.

Another interesting direction is to relax the assumptions. Awasthi et al. [6] suggest considering stability under random, and not worst-case, perturbations. Balcan and Liang [9] also study a relaxed version of the assumption, where perturbations can change the optimal clustering, but not by much. It is open to what extent, and on what data, any of these approaches will yield practical improvements.

Acknowledgements. Maria-Florina Balcan and Yingyu Liang for helpful discussions; Avrim Blum and Santosh Vempala for feedback on the writing; the anonymous referees for useful pointers.

References

1. Ackerman, M., Ben-David, S.: Clusterability: A theoretical study. Journal of Machine Learning Research - Proceedings Track 5, 1–8 (2009)
2. Ailon, N., Jaiswal, R., Monteleoni, C.: Streaming k-means approximation. In: NIPS (2009)
3. Arora, S., Raghavan, P., Rao, S.: Approximation schemes for euclidean-medians and related problems. In: STOC, pp. 106–113 (1998)

4. Arya, V., Garg, N., Khandekar, R., Meyerson, A., Munagala, K., Pandit, V.: Local search heuristics for k-median and facility location problems. SIAM J. Comput. 33(3), 544–562 (2004)
5. Awasthi, P., Blum, A., Sheffet, O.: Stability yields a ptas for k-median and k-means clustering. In: FOCS, pp. 309–318 (2010)
6. Awasthi, P., Blum, A., Sheffet, O.: Center-based clustering under perturbation stability. Inf. Process. Lett. 112(1-2), 49–54 (2012)
7. Balcan, M.F., Blum, A., Gupta, A.: Approximate clustering without the approximation. In: SODA (2009)
8. Balcan, M.F., Blum, A., Vempala, S.: A discriminative framework for clustering via similarity functions. In: STOC, pp. 671–680 (2008)
9. Balcan, M.F., Liang, Y.: Clustering under Perturbation Resilience. In: Czumaj, A., Mehlhorn, K., Pitts, A., Wattenhofer, R. (eds.) ICALP 2012, Part I. LNCS, vol. 7391, pp. 63–74. Springer, Heidelberg (2012)
10. Bartal, Y., Charikar, M., Raz, D.: Approximating min-sum -clustering in metric spaces. In: STOC, pp. 11–20 (2001)
11. Ben-David, S.: Alternative Measures of Computational Complexity with Applications to Agnostic Learning. In: Cai, J.-Y., Cooper, S.B., Li, A. (eds.) TAMC 2006. LNCS, vol. 3959, pp. 231–235. Springer, Heidelberg (2006)
12. Bilu, Y., Linial, N.: Are stable instances easy? In: Innovations in Computer Science, pp. 332–341 (2010)
13. Charikar, M., Guha, S., Tardos, É., Shmoys, D.B.: A constant-factor approximation algorithm for the k-median problem. J. Comput. Syst. Sci. 65(1), 129–149 (2002)
14. Charikar, M., O'Callaghan, L., Panigrahy, R.: Better streaming algorithms for clustering problems. In: STOC, pp. 30–39 (2003)
15. de la Vega, W.F., Karpinski, M., Kenyon, C., Rabani, Y.: Approximation schemes for clustering problems. In: STOC, pp. 50–58 (2003)
16. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1979)
17. Guha, S., Khuller, S.: Greedy strikes back: Improved facility location algorithms. J. Algorithms 31(1), 228–248 (1999)
18. Guha, S., Meyerson, A., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams: Theory and practice. IEEE Trans. Knowl. Data Eng. 15(3), 515–528 (2003)
19. Jain, K., Mahdian, M., Saberi, A.: A new greedy approach for facility location problems. In: STOC, pp. 731–740. ACM (2002)
20. Kumar, A., Sabharwal, Y., Sen, S.: Linear-time approximation schemes for clustering problems in any dimensions. J. ACM 57(2) (2010)
21. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2), 129–137 (1982)
22. Muthukrishnan, S.: Data streams: algorithms and applications. In: SODA, p. 413 (2003)
23. Ostrovsky, R., Rabani, Y., Schulman, L.J., Swamy, C.: The effectiveness of lloyd-type methods for the k-means problem. In: FOCS, pp. 165–176 (2006)
24. Schalekamp, F., Yu, M., van Zuylen, A.: Clustering with or without the Approximation. In: Thai, M.T., Sahni, S. (eds.) COCOON 2010. LNCS, vol. 6196, pp. 70–79. Springer, Heidelberg (2010)
25. van Rooij, J.M.M., van Kooten Niekerk, M.E., Bodlaender, H.L.: Partition into Triangles on Bounded Degree Graphs. In: Černá, I., Gyimóthy, T., Hromkovič, J., Jefferey, K., Královič, R., Vukolić, M., Wolf, S. (eds.) SOFSEM 2011. LNCS, vol. 6543, pp. 558–569. Springer, Heidelberg (2011)

A Dominating Set Promise Problem

A **dominating set** in a unweighted graph $G = (V, E)$ is a subset $D \subseteq V$ of vertices such that each vertex in $V \setminus D$ has a neighbor in D. A dominating set is **perfect** if each vertex in $D \setminus V$ has exactly one neighbor in D. The problems of finding the smallest dominating set and smallest perfect dominating set are NP-hard. Here we introduce a related problem, called the **perfect dominating set promise problem**. In this problem we are promised that the input graph is such that all its dominating sets of size less at most d are perfect, and we are asked to find a set of cardinality at most d. We first prove the following.

Theorem 7. *The* ***perfect dominating set promise problem*** *(PDS-PP) is NP-hard.*

Proof. The **3d matching problem** (3DM) is as follows: let X, Y, Z be finite disjoint sets with $m = |X| = |Y| = |Z|$. Let T contain triples (x, y, z) with $x \in X, y \in Y, z \in Z$ with $L = |T|$. $M \subseteq T$ is a perfect $3d$-matching if for any two triples $(x_1, y_1, z_1), (x_2, y_2, z_2) \in M$, we have $x_1 \neq x_2, y_1 \neq y_2, z_1 \neq z_2$. We notice that M is a disjoint partition. Determining whether a perfect $3d$-matching exists (YES vs. NO instance) in a $3d$-matching instance is known to be NP-complete.

Now we reduce an instance of the 3DM problem to PDS-PP on $G = (V, E)$. For 3DM elements X, Y, and Z we construct vertices V_X, V_Y, and V_Z, respectively. For each triple in T we construct a vertex in set V_T. Additionally, we make an extra vertex v. This gives $V = V_X \cup V_Y \cup V_Z \cup V_T \cup \{v\}$. We make the edge set E as follows. Every vertex in V_T (which corresponds to a triple) has an edge to the vertices that it contains in the corresponding 3DM instance (one in each of V_X, V_Y, and V_Z). Every vertex in V_T also has an edge to v.

Now we will examine the structure of the smallest dominating set D in the constructed PDS-PP instance. The vertex v must belong to D so that all vertices in V_T are covered. Then, what remains is to optimally cover the vertices in $V_X \cup V_Y \cup V_Z$ – the cheapest solution is to use m vertices from V_T , and this is precisely the 3DM problem, which is NP-hard. Hence, any solution of size $d = m + 1$ for the PDS-PP instance gives a solution to the $3DM$ instance.

We also observe that such a solution makes a perfect dominating set. Each vertex in $V_T \setminus D$ has one neighbor in D, namely v. Each vertex in $V_X \cup V_Y \cup V_Z$ has a unique neighbor in D, namely the vertex in V_T corresponding to its respective set in the 3DM instance. □

B Results for Additive Stability

Proof (of Lemma 4). We prove that for every point p and its center c_i in the optimal clustering of an additive β-perturbation resilient instance, it holds that $d(p, c_j) > d(p, c_i) + \beta$ for any $j \neq i$.

Consider an additive β-perturbation resilient clustering instance. Assume we blow up all the pairwise distances within cluster C_i by an additive factor of β. As this is a legitimate perturbation of the distance function, the optimal clustering under this perturbation is the same as the original one. Hence, p is still assigned to the same cluster. Furthermore, since the distances within C_i were all changed by the same constant factor, c_i will remain the center of the cluster. The same holds for any other optimal clusters. Since the optimal clustering under the perturbed distances is unique it follows that even in the perturbed distance function, p prefers c_i to c_j, which implies the lemma. □

Proof (of Lemma 6). Assume to the contrary that the instance is β-perturbation resilient but is not β-min-sum stable. Then, there exist clusters C_i, C_j in the optimal solution $\mathcal{C}$ and a point $p \in C_i$ such that $d(p, C_i) + \beta(|C_i| - 1) \geq d(p, C_j)$. Then, we perturb d as follows. We define d' such that for all points $q \in C_i$, $d'(p, q) = d(p, q) + \beta$, and for the remaining distances $d' = d$. Clearly d' is a valid additive β-perturbation of d.

We now note that C is not optimal under d'. Namely, we can create a cheaper solution $\mathcal{C}'$ that assigns point p to cluster C_j, and leaves the remaining clusters unchanged, which contradicts optimality of $\mathcal{C}$. This shows that $\mathcal{C}$ is not the optimum under d' which is contradictory to the fact that the instance is additive β-perturbation resilient. Therefore we conclude that if a clustering instance is additive β-perturbation resilient, then it is also additive β-min-sum stable. □

C Average Linkage for Min-Sum Stability

In this appendix, we further support the claim that algorithms designed for α-perturbation resilient instances w.r.t. the min-sum objective can often be made to work for data satisfying the more general α-min-sum stability property.

One such algorithm is the Average Linkage algorithm of Balcan and Liang [9]. The algorithm requires the condition in Lemma 8 to hold, which we can prove indeed holds for α-min-sum stable instances (their proof of the lemma holds for the more restricted class of perturbation-resilient instances). To state the lemma, we first define the distance between two point sets, A and B:

$$d(A, B) \doteq \sum_{p \in A} \sum_{q \in B} d(p, q).$$

Lemma 8. *Assume the optimal clustering is α-min-sum stable. For any two different clusters C and C' in $\mathcal{C}$ and every $A \subset C$, $\alpha d(A, \bar{A}) < d(A, C')$.*

Proof. From the definition of $\alpha d(A, \bar{A})$, we have

$$\begin{aligned} \alpha d(A, \bar{A}) &= \alpha \sum_{p \in A} \sum_{q \in \bar{A}} d(p, q) \leq \alpha \sum_{p \in A} \sum_{q \in C} d(p, q) \\ &= \sum_{p \in A} \alpha \sum_{q \in C} d(p, q) < \sum_{p \in A} \sum_{q \in C'} d(p, q) = d(A, C'). \end{aligned}$$

The first inequality comes from $\bar{A} \subset C$ and the second by definition of min-sum stability. □

This, in addition to Lemma 2, can be used to show their algorithm can be employed for min-sum stable instances.

Thompson Sampling: An Asymptotically Optimal Finite-Time Analysis

Emilie Kaufmann[1], Nathaniel Korda[2], and Rémi Munos[2]

[1] Telecom Paristech UMR CNRS 5141
[2] INRIA Lille-Nord Europe

Abstract. The question of the optimality of Thompson Sampling for solving the stochastic multi-armed bandit problem had been open since 1933. In this paper we answer it positively for the case of Bernoulli rewards by providing the first finite-time analysis that matches the asymptotic rate given in the Lai and Robbins lower bound for the cumulative regret. The proof is accompanied by a numerical comparison with other optimal policies, experiments that have been lacking in the literature until now for the Bernoulli case.

1 Introduction

In a stochastic bandit problem an agent is repeatedly asked to choose one action from an action set, each of which produces a reward drawn from an underlying, fixed, but unknown distribution associated with each action. In this paper we focus on stochastic bandits with Bernoulli rewards, initially proposed by Thompson in his paper of 1933 [14] to model medical allocation problems. Thompson's paper also presented the first bandit algorithm, Thompson Sampling. This algorithm has received much attention in the recent literature, and in this paper we give the first theoretical proof of the asymptotic optimality of this algorithm in the context of cumulative regret minimisation. Furthermore we achieve this result by giving a finite time analysis for the algorithm.

Associated with each action, a, is an unknown Bernoulli distribution $\mathcal{B}(\mu_a)$, whose expectation is μ_a. At each time t the agent chooses to observe an action $A_t \in \{1, \ldots, K\}$ and receives a reward R_t drawn from the distribution $\mathcal{B}(\mu_{A_t})$. A policy, or bandit algorithm, is defined to be a (possibly randomised) method for choosing A_t given the past history of observations and actions. The agent's goal is to minimize the expected cumulative regret of his policy, which is defined to be:

$$\mathcal{R}(T) := T\mu^* - \mathbb{E}\left[\sum_{t=1}^{T} R_t\right] = \sum_{a \in A} (\mu^* - \mu_a)\mathbb{E}[N_{a,t}] \tag{1}$$

where $\mu^* = \max_a \mu_a$ denotes the expectation of the best arm[1], or optimal action, and $N_{a,t}$ the number of draws of arm a at the end of round t. Lai and Robbins

[1] The words arms and actions are used interchangably.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 199–213, 2012.

proved in [10] that all *strongly consistent* policies (i.e. policies satisfying $\mathcal{R}(t) = o(t^\alpha)$ for all $\alpha \in (0, 1)$) must satisfy, for any suboptimal arm a

$$\liminf_{T\to\infty} \frac{\mathbb{E}[N_{a,T}]}{\ln T} \geq \frac{1}{K(\mu_a, \mu^*)} \tag{2}$$

where $K(p, q)$ denotes the Kullback-Leibler divergence between $\mathcal{B}(p)$ and $\mathcal{B}(q)$:

$$K(p, q) := p \ln \frac{p}{q} + (1 - p) \ln \frac{1 - p}{1 - q}.$$

Their result, which holds for more general classes of reward distributions, motivates defining policies that satisfy (2) with equality to be *asymptotically optimal.*

In the same paper [10] Lai and Robbins were able to describe an asymptotically optimal policy, however no finite-time analysis was provided, nor was it an efficient policy to implement. The UCB1 algorithm by Auer et al. [4] was the first of a series of efficient policies, like UCB-V [3] or MOSS [2], for which good regret bounds in finite time were also provided. These policies all use an upper confidence bound on the empirical mean of past rewards as an optimistic index for each arm, choosing at each time the action with the highest current index. However, for each of these algorithms we only have the result that there exists two constants $K_1 > 2$ and $K_2 > 0$ such that for every suboptimal action a, with $\Delta_a = \mu^* - \mu_a$,

$$\mathbb{E}[N_{a,T}] \leq \frac{K_1}{\Delta_a^2} \ln(T) + K_2. \tag{3}$$

This does not imply (2) with equality since by the Pinsker inequality one has $2K(\mu_a, \mu^*) > \Delta_a^2$. On the contrary, recently proposed index policies such as DMED [8] and KL-UCB [6, 11], which use indices obtained from KL-based confidence regions, have been shown to be asymptotically optimal.

Unlike most of this family of upper confidence bound algorithms that has been so successful, Thompson Sampling is a policy that uses ideas from Bayesian modelling and yet it solves the fundamentally frequentist problem of regret minimisation. Assume a uniform prior on each parameter μ_a, let $\pi_{a,t}$ denote the posterior distribution for μ_a after the t^{th} round of the algorithm. Let $\theta_{a,t}$ denote a sample from $\pi_{a,t}$; we sometimes refer to $\theta_{a,t}$ as a *Thompson sample.* Thompson sampling is the policy which at time t chooses to observe the action with the highest Thompson sample $\theta_{a,t}$, i.e. it chooses action a with the probability that this action has the highest expected reward under the posterior distribution.

Before Agrawal and Goyal's recent paper [1] Thompson Sampling had been investigated in [7] as the Bayesian Learning Automaton, and in [12] where an optimistic version was also proposed; however these papers only provided weak theoretical guarantees. In [5] extensive numerical experiments were carried out for Thompson Sampling beyond the scope of the Bernoulli bandit setting (to the Generalized Linear Bandit Model) but without any theoretical guarantee at all. Consequently the first finite-time analysis of Thompson Sampling in [1] was a major breakthrough, yet the upper bound for the regret that is shown in this paper scales like (3) and the question of Thompson Sampling's asymptotic optimality was still open.

Meanwhile, there has been a resurgence of interest in Bayesian strategies for bandit problems (see [9] for a review of them). The Bayes-UCB algorithm, an upper confidence bound policy which uses an adaptive quantile of $\pi_{a,t}$ as an optimistic index, was the first Bayesian algorithm to be proved asymptotically optimal. In this paper we are able to show that the same is true for a randomised Bayesian algorithm, Thompson Sampling. Moreover, we refer in our analysis to the Bayes-UCB index when introducing the deviation between a Thompson Sample and the corresponding posterior quantile.

Contributions. We provide a finite-time regret bound for Thompson Sampling, that follows from (1) and the result on suboptimal draws given in Theorem 2:

Theorem 1. *For every $\epsilon > 0$ there exists a problem-dependent constant $C(\epsilon, \mu_1, \ldots, \mu_K)$ such that the regret of Thompson Sampling satisfies:*

$$\mathcal{R}(T) \leq (1+\epsilon) \sum_{a\in A:\mu_a \neq \mu^*} \frac{\varDelta_a(\ln(T) + \ln\ln(T))}{K(\mu_a, \mu^*)} + C(\epsilon, \mu_1, \ldots, \mu_K).$$

Besides this asymptotically optimal regret bound, we also provide the first numerical experiments that show Thompson Sampling outperforming the current best optimal policies. The rest of the paper is structured as follows. Section 2 contains notations or results from [1], [6] and [9] that are useful in our finite-time analysis given in Section 3. Numerical experiments are presented in Section 4.

2 Preliminaries

We gather together here some useful preliminaries such as notations not already given in the introduction:

- For the rest of this paper, we assume action 1 is the unique optimal action. Without loss of generality[2], we can assume that the parameter $\mu = (\mu_1, ..., \mu_K)$ of the problem is such that $\mu_1 > \mu_2 \geq ... \geq \mu_K$.
- We shall denote by $S_{a,t}$ the number of successes observed from action a by time t, and denote the empirical mean by: $\hat{\mu}_{a,t} := S_{a,t}/N_{a,t}$.
- In the Bernoulli case, with a uniform prior on the parameters μ_a of the arms, the posterior on arm a at time t is explicitly
$$\pi_{a,t} = \text{Beta}\left(S_{a,t}+1, N_{a,t} - S_{a,t} + 1\right).$$
- Let $F^{\text{Beta}}_{a,b}$ denote the cdf of a Beta(a,b) distribution and $F^{\text{B}}_{j,\mu}$ (resp $f^{\text{B}}_{j,\mu}$) the cdf (resp pdf) of a Binomial(j,μ) distribution. We recall an important link between Beta and Binomial distribution used in both [1] and [9]:
$$F^{\text{Beta}}_{a,b}(y) = 1 - F^{B}_{a+b-1,y}(a-1)$$
We use this 'Beta-Binomial trick' at several stages of our analysis.

[2] In Appendix A of [1] the authors show that adding a second optimal arm can only improve the regret performance of Thompson Sampling.

- We denote by $u_{a,t}$ (resp. $q_{a,t}$) the KL-UCB (resp. Bayes-UCB) index at time t, and define them, with $Q(\alpha, \pi)$ being the α-quantile of distribution π, by

$$u_{a,t} := \underset{x > \frac{S_{a,t}}{N_{a,t}}}{\operatorname{argmax}} \left\{ K\left(\frac{S_{a,t}}{N_{a,t}}, x\right) \leq \frac{\ln(t) + \ln(\ln(T))}{N_{a,t}} \right\}$$
$$q_{a,t} := Q\left(1 - \frac{1}{t\ln(T)}, \pi_{a,t}\right).$$

A special link between these two indices is shown in [9]: $q_{a,t} < u_{a,t}$.

3 Finite Time Analysis

3.1 Sketch of Analysis

Unlike Agrawal and Goyal's analysis, which is based on explicit computation of the expectation $\mathbb{E}[N_{2,T}]$, we are inspired by standard analysis of frequentist index policies (at each round t, for each arm a these policies compute an index $l_{a,t}$ from the sequence of observed rewards from a by time t, and choose $A_t = \operatorname{argmax}_a l_{a,t}$). Such an analysis aims to bound the number of draws of a suboptimal arm, a, by considering two possible events that might lead to a play of this arm:

- the optimal arm (arm 1) is under-estimated, i.e. $l_{1,t} < \mu_1$;
- the optimal arm is not under-estimated and the suboptimal arm a is drawn.

Taking these to be a good description of when the suboptimal arm is drawn leads to the decomposition

$$\mathbb{E}[N_{a,T}] \leq \sum_{t=1}^{T} \mathbb{P}\left(l_{1,t} < \mu_1\right) + \sum_{t=1}^{T} \mathbb{P}\left((l_{a,t} \geq l_{1,t} > \mu_1) \cap (A_t = a)\right)$$

The analysis of an optimistic algorithm then proceeds by showing that the left term (the"under-estimation" term) is $o\left(\ln(T)\right)$ and the right term is of the form $\frac{1}{K(\mu_a,\mu_1)}\ln(T) + o\left(\ln(T)\right)$ (or at worst $\frac{2}{\Delta_a^2}\ln(T) + o\left(\ln(T)\right)$ as in the analysis of UCB1). This style of argument works for example for the KL-UCB algorithm [6] and also for the Bayesian optimistic algorithm Bayes-UCB [9].

However we cannot directly apply this approach to analyse Thompson Sampling, since the sample $\theta_{a,t}$ is not an optimistic estimate of μ_a. Indeed, even when $\pi_{1,t}$ is well concentrated and therefore close to a Gaussian distribution centred at μ_1, $\mathbb{P}\left(\theta_{1,t} < \mu_1\right)$ is close to $\frac{1}{2}$ and the under-estimation term is not $o\left(\ln(T)\right)$. Hence we will not compare in our proof the sample $\theta_{a,t}$ to μ_a, but to $\mu_a - \sqrt{6\ln(t)/N_{a,t}}$ (if $N_{a,t} > 0$) which is the lower bound of an UCB interval. We set the convention that if $N_{a,t} = 0$, $\sqrt{6\ln(t)/N_{a,t}} = \infty$.

As is observed in [1] the main difficulty in a regret analysis for Thompson Sampling is to control the number of draws of the optimal arm. We provide this control in the form of Proposition 1 whose proof, given in Section 3.3, explores in depth the randomised nature of Thompson Sampling.

Proposition 1. *There exists constants $b = b(\mu_1, \mu_2) \in (0,1)$ and $C_b < \infty$ such that*

$$\sum_{t=1}^{\infty} \mathbb{P}\left(N_{1,t} \leq t^b\right) \leq C_b.$$

Remark 1. In general, a result on the regret like $\mathbb{E}[N_{1,t}] \geq t - K\ln(t)$ does not imply a deviation inequality for $N_{1,t}$ (see [13]). Proposition 1 is therefore a strong result, that enables us to adapt the standard analysis mentioned above.

We can then reduce to analysing the behaviour of the algorithm once it has seen a reasonable number of draws on arm 1, and thus the posterior distribution is well concentrated. Using Proposition 1 and the new decomposition yields:

Theorem 2. *Let $\epsilon > 0$. With b as in Proposition 1, for every suboptimal arm a, there exist constants $D_\epsilon(\mu_1, \mu_a), N_\epsilon(b, \mu_1, \mu_a)$ and $N_0(b)$ such that:*

$$\mathbb{E}[N_{a,T}] \leq (1+\epsilon)\frac{\ln T + \ln\ln T}{K(\mu_a, \mu_1)} + D_\epsilon(\mu_1, \mu_a) + N_\epsilon(b, \mu_1, \mu_a) + N_0(b) + 5 + 2C_b.$$

The constants are made more explicit in the proofs of Proposition 1 and Theorem 2. The fact that Theorem 2 holds for every $\epsilon > 0$ gives us the asymptotic optimality of Thompson Sampling.

3.2 Proof of Theorem 2

Step 1: Decomposition First we recall the modified decomposition mentioned above:

$$\begin{aligned}
\mathbb{E}[N_{a,T}] &\leq \sum_{t=1}^{T} \mathbb{P}\left(\theta_{1,t} \leq \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right) + \sum_{t=1}^{T} \mathbb{P}\left(\theta_{a,t} > \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}, A_t = a\right) \\
&\leq \sum_{t=1}^{T} \mathbb{P}\left(\theta_{1,t} \leq \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right) \\
&+ \sum_{t=1}^{T} \mathbb{P}\left(\theta_{a,t} > \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}, A_t = a, \theta_{a,t} < q_{a,t}\right) + \sum_{t=1}^{T} \mathbb{P}\left(\theta_{a,t} > q_{a,t}\right)
\end{aligned}$$

The sample $\theta_{a,t}$ is not very likely to exceed the quantile of the posterior distribution $q_{a,t}$ we introduced:

$$\sum_{t=1}^{T} \mathbb{P}\left(\theta_{a,t} > q_{a,t}\right) \leq \sum_{t=1}^{T} \frac{1}{t\ln(T)} \leq \frac{1+\ln(T)}{\ln(T)} \leq 2$$

where this last inequality follows for $T \geq e$. So finally, using that $u_{a,t} \geq q_{a,t}$,

$$\mathbb{E}[N_{a,t}] \leq \underbrace{\sum_{t=1}^{T} \mathbb{P}\left(\theta_{1,t} \leq \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right)}_{A} + \underbrace{\sum_{t=1}^{T} \mathbb{P}\left(u_{a,t} > \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}, A_t = a\right)}_{B} + 2 \quad (4)$$

Step 2: Bounding term A To deal with term A we show a new self-normalized inequality adapted to the randomisation in each round of Thompson Sampling.

Lemma 1. *There exists some deterministic constant $N_0(b)$ such that*

$$\sum_{t=1}^{\infty} \mathbb{P}\left(\theta_{1,t} \leq \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right) \leq N_0(b) + 3 + C_b < \infty$$

with C_b defined as in Proposition 1.

Proof. Let (U_t) denote a sequence of i.i.d. uniform random variables, and let $\Sigma_{1,s}$ be the sum of the first s rewards from arm 1. In the following, we make the first use of the link between Beta and Binomial distributions:

$$\begin{aligned}
&\mathbb{P}\left(\theta_{1,t} \leq \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right) = \mathbb{P}\left(U_t \leq F^{\text{Beta}}_{S_{1,t}+1, N_{1,t}-S_{1,t}+1}\left(\mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right)\right) \\
&= \mathbb{P}\left(\left(U_t \leq 1 - F^{\text{B}}_{N_{1,t}+1, \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}}(S_{1,t})\right) \cap \left(N_{1,t} \geq t^b\right)\right) + \mathbb{P}\left(N_{1,t} \leq t^b\right) \\
&= \mathbb{P}\left(\left(F^{\text{B}}_{N_{1,t}+1, \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}}(S_{1,t}) \leq U_t\right) \cap \left(N_{1,t} \geq t^b\right)\right) + \mathbb{P}\left(N_{1,t} \leq t^b\right) \\
&\leq \mathbb{P}\left(\exists s \in \{t^b ... t\} : F^{\text{B}}_{s+1, \mu_1 - \sqrt{\frac{6\ln t}{s}}}(\Sigma_{1,s}) \leq U_t\right) + \mathbb{P}\left(N_{1,t} \leq t^b\right) \\
&= \sum_{s=\lceil t^b \rceil}^{t} \mathbb{P}\left(\Sigma_{1,s} \leq (F^{\text{B}})^{-1}_{s+1, \mu_1 - \sqrt{\frac{6\ln t}{s}}}(U_t)\right) + \mathbb{P}\left(N_{1,t} \leq t^b\right)
\end{aligned}$$

The first term in the final line of this display now deals only with Binomial random variables with large numbers of trials (greater than t^b), and so we can draw on standard concentration techniques to bound this term. Proposition 1 takes care of the second term.

Note that $(F^{\text{B}})^{-1}_{s+1, \mu_1 - \sqrt{6\ln t/s}}(U_t) \sim \text{Bin}\left(s+1, \mu_1 - \sqrt{6\ln t/s}\right)$ and is independent from $\Sigma_{1,s} \sim \text{Bin}(s, \mu_1)$. For each s, we define two i.i.d. sequences of Bernoulli random variables:

$$(X_{1,l})_l \sim \mathcal{B}\left(\mu_1 - \sqrt{\frac{6\ln t}{s}}\right) \text{ and } (X_{2,l})_l \sim \mathcal{B}(\mu_1),$$

and we let $Z_l := X_{2,l} - X_{1,l}$, another i.i.d. sequence, with mean $\sqrt{\frac{6\ln t}{s}}$. Using these notations,

$$\mathbb{P}\left(\Sigma_{1,s} \leq (F^{\text{B}})^{-1}_{s+1, \mu_1 - \sqrt{\frac{6\ln t}{s}}}(U_t)\right) \leq \mathbb{P}\left(\sum_{l=1}^{s}\left(Z_l - \sqrt{\frac{6\ln t}{s}}\right) \leq -\left(\sqrt{6s\ln t} - 1\right)\right).$$

Let $N_0(b)$ be such that if $t \geq N_0(b)$, $\sqrt{6t^b \ln t} - 1 > \sqrt{5t^b \ln t}$. For $t \geq N_0(b)$, we can apply Hoeffding's inequality to the bounded martingale difference sequence $Z'_l = Z_l - \sqrt{6 \ln t / s}$ to get

$$\mathbb{P}\left(\Sigma_{1,s} < (F^{\mathrm{B}})^{-1}_{s+1,\mu_1 - \sqrt{\frac{6\ln t}{s}}}(U_t)\right) \leq \exp\left(-2\frac{(\sqrt{5s\ln t})^2}{4s}\right) = e^{-\frac{5}{2}\ln t} = \frac{1}{t^{\frac{5}{2}}}.$$

We conclude that

$$\sum_{t=1}^{\infty} \mathbb{P}\left(\theta_1(t) < \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}\right) \leq N_0(b) + \sum_{t=1}^{\infty} \frac{1}{t^{\frac{3}{2}}} + C_b \leq N_0(b) + 3 + C_b.$$

Step 3: Bounding Term B We specifically show that:

Lemma 2. *For all $a = 2, \ldots, K$, for any $\epsilon > 0$ there exist positive constants $N_\epsilon(b, \mu_1, \mu_a), D_\epsilon(\mu_1, \mu_a)$ such that for all $T > N_\epsilon(b, \mu_1, \mu_a)$*

$$(B) \leq (1+\epsilon)\frac{\ln(T) + \ln\ln(T)}{K(\mu_a, \mu_1)} + D_\epsilon(\mu_1, \mu_a).$$

Proof. First rewrite term B so that we can apply Proposition 1:

$$\begin{aligned}(B) &\leq \sum_{t=1}^{T} \mathbb{P}\left(u_{a,t} > \mu_1 - \sqrt{\frac{6\ln t}{N_{1,t}}}, A_t = a, N_{1,t} \geq t^b\right) + \sum_{t=1}^{T} \mathbb{P}\left(N_{1,t} \leq t^b\right) \\ &\leq \sum_{t=1}^{T} \mathbb{P}\left(u_{a,t} > \mu_1 - \sqrt{\frac{6\ln t}{t^b}}, A_t = a\right) + C_b\end{aligned}$$

For ease of notation we introduce

$$K^+(x,y) := K(x,y)\mathbf{1}_{(x \leq y)}, \;\; f_T(t) := \ln t + \ln(\ln(T))$$
$$\beta_t = \sqrt{\frac{6\ln t}{t^b}}, \text{ and } K_{T,a}(\epsilon) = (1+\epsilon)\frac{\ln(T) + \ln\ln(T)}{K(\mu_a, \mu_1)}.$$

Now

$$(u_{a,t} \geq \alpha) = \left(N_{2,t} K^+(\hat{\mu}_{2,N_{2,t}}, \alpha) \leq f_T(t)\right)$$

and so summing over the values of $N_{2,t}$ and inverting the sums we get

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{P}\left(u_{a,t} > \mu_1 - \beta_t, A_t = a\right) &= \mathbb{E}\left[\sum_{\substack{s \leq K_{T,a} \\ s \leq t}} \mathbf{1}_{(sK^+(\hat{\mu}_{a,s}, \mu_1 - \beta_t) \leq f_T(t))} \mathbf{1}_{(I_t = a, N_{2,t} = s)}\right] \\ &+ \mathbb{E}\left[\sum_{s=\lfloor K_{T,a}\rfloor + 1}^{T} \sum_{t=s}^{T} \mathbf{1}_{(sK^+(\hat{\mu}_{a,s}, \mu_1 - \beta_t) \leq f_T(t))} \mathbf{1}_{(A_t = a, N_{2,t} = s)}\right].\end{aligned}$$

Given that $\sum_{t=s}^{T}\mathbf{1}_{(A_t=a,N_{2,t}=s)} \leq 1$ for all s, the first term is upper bounded by $K_{a,T}$ whereas the second is upper bounded by

$$\mathbb{E}\left[\sum_{s=\lfloor K_{T,a}\rfloor+1}^{T} \mathbf{1}_{\left(sK^{+}\left(\hat{\mu}_{a,s},\mu_1-\beta_{K_{T,a}}\right)\leq f_T(T)\right)}\right],$$

where we use that $y \mapsto K^{+}\left(\hat{\mu}_{a,s},y\right)$ is increasing and that for t large enough ($t \geq e^{1/b}$) $t \mapsto \beta_t$ is decreasing, and so the last inequality holds for T such that

$$K_{T,a}(\epsilon) \geq e^{1/b}. \tag{5}$$

Finally, for such T,

$$(B) \leq K_{T,a} + \sum_{\lfloor K_{T,a}\rfloor+1} \mathbb{P}\left(K^{+}\left(\hat{\mu}_{a,s},\mu_1-\beta_{K_{T,a}}\right) \leq \frac{K(\mu_a,\mu_1)}{1+\epsilon}\right).$$

Because $K^{+}\left(\hat{\mu}_{a,s},.\right)$ is convex, we can show that on the above event:

$$K^{+}(\hat{\mu}_{a,s},\mu_1) \leq K^{+}(\hat{\mu}_{a,s},\mu_1-\beta_{K_{a,T}}) + \frac{2}{\mu_1(1-\mu_1)}\beta_{K_{a,T}} \leq \frac{K(\mu_a,\mu_1)}{1+\epsilon/2} \tag{6}$$

where the last inequality holds for large enough T. So there exists some $N = N_\epsilon(b,\mu_1,\mu_a)$ such that all $T > N$ satisfy both (5) and (6). Hence, for all $T \geq N$

$$(B) \leq K_{T,a} + \sum_{\lfloor K_{T,a}\rfloor+1} \mathbb{P}\left(K^{+}\left(\hat{\mu}_{a,s},\mu_1\right) \leq \frac{K(\mu_a,\mu_1)}{1+\frac{\epsilon}{2}}\right).$$

Since this last sum is bounded above explicitly by some constant $D_\epsilon(\mu_1,\mu_a)$ in [11] we have proved the lemma. One has $D_\epsilon(\mu_1,\mu_a) = \frac{(1+\epsilon/2)^2}{\epsilon^2(\min(\mu_a(1-\mu_a);\mu_1(1-\mu_1)))^2}$.

Conclusion: The result now follows from Lemmas 1, 2 and inequality (4).

3.3 Proof of Proposition 1

Since we focus on the number of draws of the optimal arm, let τ_j be the occurence of the j^{th} play of the optimal arm (with $\tau_0 := 0$). Let $\xi_j := (\tau_{j+1}-1)-\tau_j$: this random variable measures the number of time steps between the j^{th} and the $(j+1)^{th}$ play of the optimal arm, and so $\sum_{a=2}^{K} N_{a,t} = \sum_{j=0}^{N_{1,t}} \xi_j$. For each suboptimal arm, a relevant quantity is $C_a = \frac{32}{(\mu_1-\mu_a)^2}$ and let $C = \max_{a\neq 1} C_a = \frac{32}{(\mu_1-\mu_2)^2}$. We also introduce $\delta_a = \frac{\mu_1-\mu_a}{2}$ and let $\delta = \delta_2$.

Step 1: Initial Decomposition of Summands First we use a union bound on the summands to extract the tails of the random variables ξ_j:

$$\begin{aligned}\mathbb{P}(N_{1,t} \le t^b) &= \mathbb{P}\left(\sum_{a=2}^{K} N_{2,t} \ge t - t^b\right)\\ &\le \mathbb{P}\left(\exists j \in \left\{0, .., \lfloor t^b \rfloor\right\} : \xi_j \ge t^{1-b} - 1\right)\\ &\le \sum_{j=0}^{\lfloor t^b \rfloor} \mathbb{P}(\xi_j \ge t^{1-b} - 1) \qquad (7)\end{aligned}$$

This means that there exists a time range of length $t^{1-b} - 1$ during which only suboptimal arms are played. In the case of two arms this implies that the (unique) suboptimal arm is played $\lceil \frac{t^{1-b}-1}{2} \rceil$ times during the first half of this time range. Thus its posterior becomes well concentrated around its mean with high probability, and we can use this fact to show that the probability the suboptimal action is chosen a further $\lceil \frac{t^{1-b}-1}{2} \rceil$ times in a row is very small.

To generalise this approach we introduce a notion of a *saturated*, suboptimal action:

Definition 1. *Let t be fixed. For any $a \neq 1$, an action a is said to be* saturated *at time s if it has been chosen at least $C_a \ln(t)$ times. That is $N_{a,s} \ge C_a \ln(t)$. We shall say that it is* unsaturated *otherwise. Furthermore at any time we call a choice of an unsaturated, suboptimal action an* interruption*.*

We want to study the event $E_j = \{\xi_j \ge t^{1-b} - 1\}$. We introduce the interval $\mathcal{I}_j = \{\tau_j, \tau_j + \lceil t^{1-b} - 1 \rceil\}$ (included in $\{\tau_j, \tau_{j+1}\}$ on E_j) and begin by decomposing it into K subintervals:

$$\mathcal{I}_{j,l} := \left\{\tau_j + \left\lceil \frac{(l-1)(t^{1-b}-1)}{K} \right\rceil, \tau_j + \left\lceil \frac{l(t^{1-b}-1)}{K} \right\rceil \right\},\ l = 1, \ldots, K.$$

Now for each interval $\mathcal{I}_{j,l}$, we introduce:

- $F_{j,l}$: the event that by the end of the interval $\mathcal{I}_{j,l}$ at least l suboptimal actions are saturated;
- $n_{j,l}$: the number of interruptions during this interval.

We use the following decomposition to bound the probability of the event E_j:

$$\mathbb{P}(E_j) = \mathbb{P}(E_j \cap F_{j,K-1}) + \mathbb{P}(E_j \cap F_{j,K-1}^c) \qquad (8)$$

To bound both probabilities, we will need the fact, stated in Lemma 3, that the probability of $\theta_{1,s}$ being smaller than $\mu_2 + \delta$ during a long subinterval of $\mathcal{I}_j$ is small. This follows from the fact that the posterior on the optimal arm is always $\text{Beta}(S_{1,\tau_j} + 1, j - S_{1,\tau_j} + 1)$ on $\mathcal{I}_j$: hence, when conditioned on S_{1,τ_j}, $\theta_{1,s}$ is an i.i.d. sequence with non-zero support above $\mu_2 + \delta$, and thus is unlikely to remain below $\mu_2 + \delta$ for a long time period. This idea is also an important tool in the analysis of Thompson Sampling in [1].

Lemma 3. *$\exists\lambda_0 = \lambda_0(\mu_1,\mu_2) > 1$ such that for $\lambda \in]1,\lambda_0[$, for every (random) interval $\mathcal{J}$ included in $\mathcal{I}_j$, and every positive function f, one has*

$$\mathbb{P}\left(\forall s \in \mathcal{J}, \theta_{1,s} \leq \mu_2 + \delta \ , \ |\mathcal{J}| \geq f(t)\right) \leq (\alpha_{\mu_1,\mu_2})^{f(t)} + C_{\lambda,\mu_1,\mu_2}\frac{1}{f(t)^\lambda}e^{-jd_{\lambda,\mu_1,\mu_2}}$$

where $C_{\lambda,\mu_1,\mu_2}, d_{\lambda,\mu_1,\mu_2} > 0$ and $\alpha_{\mu_1,\mu_2} = (1/2)^{1-\mu_2-\delta}$.

Due to space limitations we omit the proof of this important lemma which can be found in the arxiv version of this paper. Another key point in the proof is the fact that a sample from a saturated suboptimal arm cannot fall too far from its true mean. The following lemma is easily adapted from Lemma 2 in [1].

Lemma 4

$$\mathbb{P}\left(\exists s \leq t, \exists a \neq 1 : \theta_{a,s} > \mu_a + \delta_a, N_{a,s} > C_a \ln(t)\right) \leq \frac{2(K-1)}{t^2}.$$

Step 2: Bounding $\mathbb{P}(E_j \cap F_{j,K-1})$ On the event $E_j \cap F_{j,K-1}$, only saturated suboptimal arms are drawn on the interval $\mathcal{I}_{j,K}$. Using the concentration results for samples of these arms in Lemma 4, we get

$$\begin{aligned}\mathbb{P}(E_j \cap F_{j,K-1}) \leq & \mathbb{P}(\{\exists s \in \mathcal{I}_{j,K}, a \neq 1 : \theta_{a,s} > \mu_a + \delta\} \cap E_j \cap F_{j,K-1}) \\ & + \mathbb{P}(\{\forall s \in \mathcal{I}_{j,K}, a \neq 1 : \theta_{a,s} \leq \mu_a + \delta_a\} \cap E_j \cap F_{j,K-1}) \\ \leq & \mathbb{P}(\exists s \leq t, a \neq 1 : \theta_{a,s} > \mu_a + \delta_a, N_{a,t} > C_a \ln(t)) \\ & + \mathbb{P}(\{\forall s \in \mathcal{I}_{j,K}, a \neq 1 : \theta_{a,s} \leq \mu_2 + \delta\} \cap E_j \cap F_{j,K-1}) \\ \leq & \frac{2(K-1)}{t^2} + \mathbb{P}(\theta_{1,s} \leq \mu_2 + \delta, \forall s \in \mathcal{I}_{j,K}).\end{aligned}$$

The last inequality comes from the fact that if arm 1 is not drawn, the sample $\theta_{1,s}$ must be smaller than some sample $\theta_{a,s}$ and therefore smaller than $\mu_2 + \delta$. Since $\mathcal{I}_{j,K}$ is an interval in $\mathcal{I}_j$ of size $\left\lceil \frac{t^{1-b}-1}{K} \right\rceil$ we get using Lemma 3, for some fixed $\lambda \in]1,\lambda_0[$,

$$\begin{aligned}&\mathbb{P}(\theta_{1,s} \leq \mu_2 + \delta, \forall s \in \mathcal{I}_{j,K}\}) \\ &\quad \leq (\alpha_{\mu_1,\mu_2})^{\frac{t^{1-b}-1}{K}} + C_{\lambda,\mu_1,\mu_2}\frac{1}{\left(\frac{t^{1-b}-1}{K}\right)^\lambda}e^{-jd_{\lambda,\mu_1,\mu_2}} =: g(\mu_1,\mu_2,b,j,t).\end{aligned} \tag{9}$$

Hence we have show that

$$\mathbb{P}(E_j \cap F_{j,K-1}) \leq \frac{2(K-1)}{t^2} + g(\mu_1,\mu_2,b,j,t), \tag{10}$$

and choosing b such that $b < 1 - \frac{1}{\lambda}$, the following hypothesis on g holds:

$$\sum_{t\geq 1}\sum_{j \leq t^b} g(\mu_1,\mu_2,b,j,t) < +\infty.$$

Step 3: Bounding $\mathbb{P}(E_j \cap F^c_{j,K-1})$ We show through an induction that for all $2 \leq l \leq K$, if t is larger than some deterministic constant $N_{\mu_1,\mu_2,b}$ specified in the base case,

$$\mathbb{P}(E_j \cap F^c_{j,l-1}) \leq (l-2)\left(\frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t)\right)$$

for some function f such that $\sum_{t\geq 1}\sum_{1\leq j\leq t^b} f(\mu_1,\mu_2,b,j,t) < \infty$. For $l = K$ we get

$$\mathbb{P}(E_j \cap F^c_{j,K-1}) \leq (K-2)\left(\frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t)\right). \tag{11}$$

Step 4: The Base Case of the induction Note that on the event E_j only suboptimal arms are played during $\mathcal{I}_{j,1}$. Hence at least one suboptimal arm must be played $\lceil\frac{t^{1-b}-1}{K^2}\rceil$ times.

There exists some deterministic constant $N_{\mu_1,\mu_2,b}$ such that for $t \geq N_{\mu_1,\mu_2,b}$, $\lceil\frac{t^{1-b}-1}{K^2}\rceil \geq C\ln(t)$ (the constant depends only on μ_1 and μ_2 because $C = C_2$). So when $t \geq N_{\mu_1,\mu_2,b}$, at least one suboptimal arm must be saturated by the end of $\mathcal{I}_{j,1}$. Hence, for $t \geq N_{\mu_1,\mu_2,b}$

$$\mathbb{P}(E_j \cap F^c_{j,1}) = 0.$$

This concludes the base case.

Step 5: The Induction As an inductive hypothesis we assume that for some $2 \leq l \leq K-1$ if $t \geq N_{\mu_1,\mu_2,b}$ then

$$\mathbb{P}(E_j \cap F^c_{j,l-1}) \leq (l-2)\left(\frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t)\right).$$

Then, making use of the inductive hypothesis,

$$\begin{aligned}\mathbb{P}(E_j \cap F^c_{j,l}) &\leq \mathbb{P}(E_j \cap F^c_{j,l-1}) + \mathbb{P}(E_j \cap F^c_{j,l} \cap F_{j,l-1})\\ &\leq (l-2)\left(\frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t)\right) + \mathbb{P}(E_j \cap F^c_{j,l} \cap F_{j,l-1}).\end{aligned}$$

To complete the induction we therefore need to show that:

$$\mathbb{P}(E_j \cap F^c_{j,l} \cap F_{j,l-1}) \leq \frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t). \tag{12}$$

On the event $(E_j \cap F^c_{j,l} \cap F_{j,l-1})$, there are exactly $l-1$ saturated arms at the beginning of interval $\mathcal{I}_{j,l}$ and no new arm is saturated during this interval. As a result there cannot be more than $KC\ln(t)$ interruptions during this interval, and so we have

$$\mathbb{P}(E_j \cap F^c_{j,l} \cap F_{j,l-1}) \leq \mathbb{P}(E_j \cap F_{j,l-1} \cap \{n_{j,l} \leq KC\ln(t)\}).$$

Let $\mathcal{S}_l$ denote the set of saturated arms at the end of $\mathcal{I}_{j,l}$ and introduce the following decomposition:

$$\begin{aligned}
&\mathbb{P}(E_j \cap F_{j,l-1} \cap \{n_{j,l} \leq KC\ln(t)\}) \\
&\qquad \leq \underbrace{\mathbb{P}(\{\exists s \in \mathcal{I}_{j,l}, a \in \mathcal{S}_{l-1} : \theta_{a,s} > \mu_a + \delta_a\} \cap E_j \cap F_{j,l-1})}_{A} \\
&\quad + \underbrace{\mathbb{P}(\{\forall s \in \mathcal{I}_{j,l}, a \in \mathcal{S}_{l-1} : \theta_{a,s} \leq \mu_a + \delta_a\} \cap E_j \cap F_{j,l-1} \cap \{n_{j,l} \leq KC\ln(t)\})}_{B}.
\end{aligned}$$

Clearly, using Lemma 4:

$$(A) \leq \mathbb{P}\left(\exists s \leq t, \exists a \neq 1 : \theta_{a,s} > \mu_a + \delta_a, N_{a,s} > C_a \ln(t)\right) \leq \frac{2(K-1)}{t^2}.$$

To deal with term (B), we introduce for k in $\{0, \ldots, n_{j,l}-1\}$ the random intervals $\mathcal{J}_k$ as the time range between the k^{th} and $(k+1)^{st}$ interruption in $\mathcal{I}_{j,l}$. For $k \geq n_{j,l}$ we set $\mathcal{J}_k = \varnothing$. Note that on the event in the probability (B) there is a subinterval of $\mathcal{I}_{j,l}$ of length $\left\lceil \frac{t^{1-b}-1}{CK^2\ln(t)} \right\rceil$ during which there are no interruptions. Moreover on this subinterval of $\mathcal{I}_{j,l}$, for all $a \neq 1$, $\theta_{a,s} \leq \mu_2 + \delta_2$. (This holds for unsaturated arms as well as for saturated arms since their samples are smaller than the maximum sample of a saturated arm.) Therefore,

$$\begin{aligned}
(B) &\leq \mathbb{P}\big(\big\{\exists k \in \{0,...,n_{j,l}\} : |\mathcal{J}_k| \geq (t^{1-b}-1)/(CK^2\ln(t))\big\} \\
&\qquad\qquad \cap \{\forall s \in \mathcal{I}_{j,l}, a \in \mathcal{S}_{l-1} : \theta_{a,s} \leq \mu_2 + \delta\} \cap E_j \cap F_{j,l-1}) \\
&\leq \sum_{k=1}^{KC\ln(t)} \mathbb{P}\left(\left\{|\mathcal{J}_k| \geq \frac{t^{1-b}-1}{CK^2\ln(t)}\right\} \cap \{\forall s \in \mathcal{J}_k, a \neq 1 : \theta_{a,s} \leq \mu_2 + \delta\} \cap E_j\right) \\
&\leq \sum_{k=1}^{KC\ln(t)} \mathbb{P}\left(\left\{|\mathcal{J}_k| \geq \frac{t^{1-b}-1}{CK^2\ln(t)}\right\} \cap \{\forall s \in \mathcal{J}_k,\ \theta_{1,s} \leq \mu_2 + \delta\}\right) \qquad (13)
\end{aligned}$$

Now, we have to bound the probability that $\theta_{1,s} \leq \mu_2 + \delta$ for all s in an interval of size $\frac{t^{1-b}-1}{CK^2\ln(t)}$ in $\mathcal{I}_j$. So we apply Lemma 3 to get:

$$(B) \leq CK\ln(t)(\alpha_{\mu_1,\mu_2})^{\frac{t^{1-b}-1}{CK^2\ln(t)}} + C_{\lambda,\mu_1,\mu_2}\frac{CK\ln(t)}{\left(\frac{t^{1-b}-1}{CK^2\ln(t)}\right)^{\lambda}}e^{-jd_{\lambda,\mu_1,\mu_2}} := f(\mu_1,\mu_2,b,j,t).$$

Choosing the same b as in (9), we get that $\sum_{t\geq 1}\sum_{1\leq j\leq t^b} f(\mu_1,\mu_2,b,j,t) < +\infty$. It follows that for this value of b, (12) holds and the induction is complete.

Step 8: Conclusion Let b be the constant chosen in Step 2. From the decomposition (8) and the two upper bounds (10) and (11), we get, for $t \geq N_{\mu_1,\mu_2,b}$:

$$\mathbb{P}(E_j) \leq (K-2)\left(\frac{2(K-1)}{t^2} + f(\mu_1,\mu_2,b,j,t)\right) + \frac{2(K-1)}{t^2} + g(\mu_1,\mu_2,b,j,t).$$

Recalling (7), summing over the possible values of j and t we obtain:

$$\sum_{t\geq 1}\mathbb{P}(N_{1,t}\leq t^b)\leq N_{\mu_1,\mu_2,b}+2(K-1)^2\sum_{t\geq 1}\frac{1}{t^{2-b}} \\ +\sum_{t\geq 1}\sum_{j=1}^{t^b}[Kf(\mu_1,\mu_2,b,j,t)+g(\mu_1,\mu_2,b,j,t)]<C_{\mu_1,\mu_2,b}$$

for some constant $C_{\mu_1,\mu_2,b}<\infty$.

4 Experiments

We illustrate the performance of Thompson Sampling on numerical experiments with Bernoulli rewards. First we compare the cumulative regret of Thompson Sampling to UCB, KL-UCB and Bayes-UCB on two different two-arms problems up to a horizon $T = 10000$, one with small and the other with high rewards, with different gaps between the parameters of the arms. Figure 1 shows Thompson Sampling always outperforms KL-UCB and also Bayes-UCB eventually. The three optimal policies are significantly better than UCB, even for small horizons.

Figure 2 displays for several algorithms an estimation of the distribution of the cumulative regret based on $N = 50000$ trials, for a horizon $T = 20000$ in a 10-armed bandit problem with small rewards already studied in [6]. The first two algorithms are variants of UCB. Of these the UCB-V algorithm is close to the index policy to which Thompson Sampling is compared in [5] in the Bernoulli setting, but this policy is not known to be optimal. This algorithm incorporates an estimation of the variance of the rewards in the index which is defined to be, for an arm that have produced k rewards in n draws,

$$\frac{k}{n}+\sqrt{\frac{2\ln(t)}{n}\frac{k}{n}\left(1-\frac{k}{n}\right)}+\frac{3\ln(t)}{n}$$

The other algorithms displayed in Figure 2 have a mean regret closer (sometimes smaller) than the (asymptotic) lower bound, and among them, Thompson is the best. It is also the easiest optimal policy to implement, since the optimization problem solved in KL-UCB and the computation of the quantiles in Bayes-UCB are more costly than producing one sample from the posterior for each arm.

5 Discussion

This paper provides the first proof of the asymptotic optimality of Thompson Sampling for Bernoulli bandits. Moreover the proof consists in a finite time analysis comparable with that of other known optimal policies. We also provide here simulations showing that Thompson Sampling outperforms currently known optimal policies.

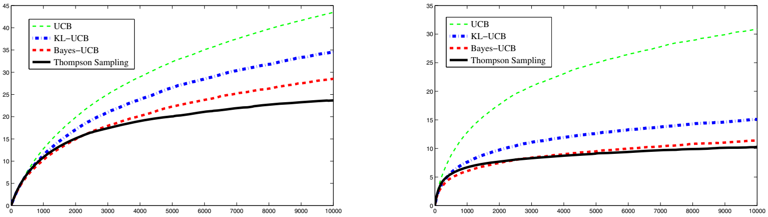


Fig. 1. Cumulated regret for the two-arms problems with $\mu_1 = 0.2, \mu_2 = 0.25$ (left) and $\mu_1 = 0.8, \mu_2 = 0.9$ (right). Regret is estimated as an average over $N = 20000$ trials.

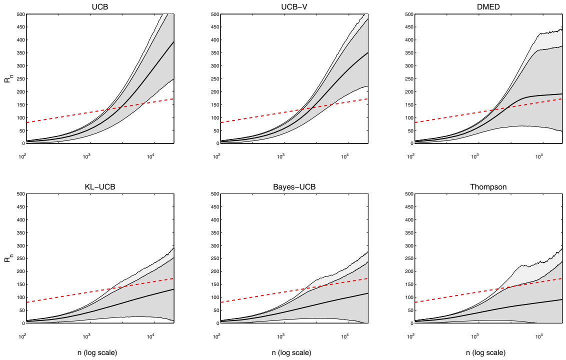


Fig. 2. Regret as a function of time (on a log scale). The red dashed line shows the lower bound, the solid bold curve corresponds to the mean regret while the dark and light shaded regions show respectively the central 99% and the upper 0.05%.

Our proof of optimality borrows some ideas from [1], such as the notion of saturated arms. However we make use of these ideas together with our own to obtain a stronger result, namely control over the tail of $N_{1,t}$ rather than its expectation. This is a valuable result which justifies the complexity of the proof of Proposition 2. Control over these tails allows us to give a simpler finite time analysis for Thompson Sampling which is closer to the arguments for UCB-like algorithms, and yields the optimal asymptotic rate of Lai and Robbins.

Thanks to the generalisation pointed out in [1], the Bernoulli version of Thompson Sampling can be applied to bandit problems with bounded rewards, and is therefore an excellent alternative to UCB policies. It would also be very natural to generalise Thompson to more complex reward distributions, choosing a prior appropriate for the assumptions on these distributions. Indeed, even in complex settings where the prior is not computable, Thompson Sampling only requires one sample from the posterior, which can be obtained efficiently using MCMC. Encouraging numerical experiments for reward distributions in the exponential family using a conjugate prior suggest that a generalisation of the proof is achievable. However this poses quite a challenge since the proof here is often heavily dependent on specific properties of Beta distributions. A natural generalisation would need a prior-dependent finite-time result controlling the tail probabilities of posterior distributions as the number of samples increases.

Acknowledgments. We thank Aurélien Garivier and Olivier Cappé, for many fruitful discussions and for giving us the opportunity to work together.

This work was supported by the French National Research Agency (ANR-08-COSI-004 project EXPLO-RA) and the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreements n°216886 (PASCAL2), and n°270327 (CompLACS).

References

[1] Agrawal, S., Goyal, N.: Analysis of thompson sampling for the multi-armed bandit problem. In: Conference on Learning Theory, COLT (2012)

[2] Audibert, J.-Y., Bubeck, S.: Regret bounds and minimax policies under partial monitoring. Journal of Machine Learning Research 11, 2785–2836 (2010)

[3] Audibert, J.-Y., Munos, R., Szepesvári, C.: Exploration-exploitation trade-off using variance estimates in multi-armed bandits. Theoretical Computer Science 410(19), 1876–1902 (2009)

[4] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2), 235–256 (2002)

[5] Chapelle, O., Li, L.: An empirical evaluation of thompson sampling. In: NIPS (2011)

[6] Garivier, A., Cappé, O.: The kl-ucb algorithm for bounded stochastic bandits and beyond. In: Conference on Learning Theory, COLT (2011)

[7] Granmo, O.C.: Solving two-armed bernoulli bandit problems using a bayesian learning automaton. International Journal of Intelligent Computing and Cybernetics 3(2), 207–234 (2010)

[8] Honda, J., Takemura, A.: An asymptotically optimal bandit algorithm for bounded support models. In: Conference on Learning Theory, COLT (2010)

[9] Kaufmann, E., Garivier, A., Cappé, O.: On bayesian upper-confidence bounds for bandit problems. In: AISTATS (2012)

[10] Lai, T.L., Robbins, H.: Asymptotically efficient adaptive allocation rules. Advances in Applied Mathematics 6(1), 4–22 (1985)

[11] Maillard, O.-A., Munos, R., Stoltz, G.: A finite-time analysis of multi-armed bandits problems with kullback-leibler divergences. In: Conference on Learning Theory, COLT (2011)

[12] May, B.C., Korda, N., Lee, A., Leslie, D.: Optimistic bayesian sampling in contextual bandit problems. Journal of Machine Learning Research 13, 2069–2106 (2012)

[13] Salomon, A., Audibert, J.-Y.: Deviations of Stochastic Bandit Regret. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 159–173. Springer, Heidelberg (2011)

[14] Thompson, W.R.: On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25, 285–294 (1933)

Regret Bounds for Restless Markov Bandits

Ronald Ortner[1,2], Daniil Ryabko[2], Peter Auer[1], and Rémi Munos[2]

[1] Montanuniversitaet Leoben
[2] INRIA Lille-Nord Europe, équipe SequeL
{rortner,auer}@unileoben.ac.at, daniil@ryabko.net, remi.munos@inria.fr

Abstract. We consider the restless Markov bandit problem, in which the state of each arm evolves according to a Markov process independently of the learner's actions. We suggest an algorithm that after T steps achieves $\tilde{O}(\sqrt{T})$ regret with respect to the best policy that knows the distributions of all arms. No assumptions on the Markov chains are made except that they are irreducible. In addition, we show that index-based policies are necessarily suboptimal for the considered problem.

1 Introduction

In the bandit problem the learner has to decide at time steps $t = 1, 2, \ldots$ which of the finitely many available arms to pull. Each arm produces a reward in a stochastic manner. The goal is to maximize the reward accumulated over time.

Following [1], traditionally it is assumed that the rewards produced by each given arm are independent and identically distributed (i.i.d.). If the probability distributions of the rewards of each arm are known, the best strategy is to only pull the arm with the highest expected reward. Thus, in the i.i.d. bandit setting the *regret* is measured with respect to the best arm. An extension of this setting is to assume that the rewards generated by each arm are not i.i.d., but are governed by some more complex stochastic process. Markov chains suggest themselves as an interesting and non-trivial model. In this setting it is often natural to assume that the stochastic process (Markov chain) governing each arm does not depend on the actions of the learner. That is, the chain takes transitions independently of whether the learner pulls that arm or not (giving the name *restless bandit* to the problem). The latter property makes the problem rather challenging: since we are not observing the state of each arm, the problem becomes a partially observable Markov decision process (POMDP), rather than being a (special case of) a fully observable MDP, as in the traditional i.i.d. setting. One of the applications that motivate the restless bandit problem is the so-called *cognitive radio* problem (e.g., [2]): Each arm of the bandit is a radio channel that can be busy or available. The learner (an appliance) can only sense a certain number of channels (in the basic case only a single one) at a time, which is equivalent to pulling an arm. It is natural to assume that whether the channel is busy or not at a given time step depends on the past — so a Markov chain is the simplest realistic model — but does not depend on which channel

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 214–228, 2012.

the appliance is sensing. (See also Example 1 in Section 3 for an illustration of a simple instance of this problem.)

What makes the restless Markov bandit problem particularly interesting is that *one can do much better than pulling the best arm.* This can be seen already on simple examples with two-state Markov chains (see Section 3 below). Remarkably, this feature is often overlooked, notably by some early work on restless bandits, e.g. [3], where the regret is measured with respect to the mean reward of the best arm. This feature also makes the problem more difficult and in some sense more general than the non-stochastic bandit problem, in which the regret usually is measured with respect to the best arm in hindsight [4]. Finally, it is also this feature that makes the problem principally different from the so-called *rested* bandit problem, in which each Markov chain only takes transitions when the corresponding arm is pulled.

Thus, in the restless Markov bandit problem that we study, the regret should be measured not with respect to the best arm, but with respect to the best policy knowing the distribution of all arms. To understand what kind of regret bounds can be obtained in this setting, it is useful to compare it to the i.i.d. bandit problem and to the problem of learning an MDP. In the i.i.d. bandit problem, the minimax regret expressed in terms of the horizon T and the number of arms only is $O(\sqrt{T})$, cf. [5]. If we allow problem-dependent constants into consideration, then the regret becomes of order $\log T$ but depends also on the gap between the expected reward of the best and the second-best arm. In the problem of learning to behave optimally in an MDP, nontrivial problem-independent finite-time regret guarantees (that is, regret depending only on T and the number of states and actions) are not possible to achieve. It is possible to obtain $O(\sqrt{T})$ regret bounds that also depend on the diameter of the MDP [6] or similar related constants, such as the span of the optimal bias vector [7]. Regret bounds of order $\log T$ are only possible if one additionally allows into consideration constants expressed in terms of policies, such as the gap between the average reward obtained by the best and the second-best policy [6]. The difference between these constants and constants such as the diameter of an MDP is that one can try to estimate the latter, while estimating the former is at least as difficult as solving the original problem — finding the best policy. Turning to our restless Markov bandit problem, so far, to the best of our knowledge no regret bounds are available for the general problem. However, several special cases have been considered. Specifically, $O(\log T)$ bounds have been obtained in [8] and [9]. While the latter considers the two-armed restless bandit case, the results of [8] are constrained by some ad hoc assumptions on the transition probabilities and on the structure of the optimal policy of the problem. Also the dependence of the regret bound on the problem parameters is unclear, while computational aspects of the algorithm (which alternates exploration and exploitation steps) are neglected. Finally, while regret bounds for the Exp3.S algorithm [4] could be applied, these depend on the "hardness" of the reward sequences, which in the case of reward sequences generated by a Markov chain can be arbitrarily high.

Here we present an algorithm for which we derive $\tilde{O}(\sqrt{T})$ regret bounds, making no assumptions on the distribution of the Markov chains. The algorithm is based on constructing an approximate MDP representation of the POMDP problem, and then using a modification of the UCRL2 algorithm of [6] to learn this approximate MDP. In addition to the horizon T and the number of arms and states, the regret bound also depends on the diameter and the mixing time (which can be eliminated however) of the Markov chains of the arms. If the regret has to be expressed only in these terms, then our lower bound shows that the dependence on T cannot be significantly improved.

2 Preliminaries

Given are K arms, where underlying each arm j there is an irreducible Markov chain with state space S_j and transition matrix P_j. For each state s in S_j there are mean rewards $r_j(s)$, which we assume to be bounded in $[0,1]$. For the time being, we will assume that the learner knows the number of states for each arm and that all Markov chains are aperiodic. In Section 7, we discuss periodic chains, while in Section 8 we indicate how to deal with unknown state spaces. In any case, the learner knows neither the transition probabilities nor the mean rewards.

For each time step $t = 1, 2, \dots$ the learner chooses one of the arms, observes the current state s of the chosen arm i and receives a random reward with mean $r_i(s)$. After this, the state of each arm j changes according to the transition matrices P_j. The learner however is not able to observe the current state of the individual arms. We are interested in competing with the optimal policy π^* which knows the mean rewards and transition matrices, yet observes as the learner only the current state of the chosen arm. Thus, we are looking for algorithms which after any T steps have small regret with respect to π^*, i.e. minimize

$$T \cdot \rho^* - \textstyle\sum_{t=1}^{T} r_t,$$

where r_t denotes the (random) reward earned at step t and ρ^* is the average reward of the optimal policy π^*. (It will be seen in Section 5 that π^* and ρ^* are indeed well-defined.)

Mixing Times and Diameter. If an arm j is not selected for a large number of time steps, the distribution over states when selecting j will be close to the stationary distribution μ_j of the Markov chain underlying arm j. Let μ_s^t be the distribution after t steps when starting in state $s \in S_j$. Then setting

$$d_j(t) := \max_{s \in S_j} \|\mu_s^t - \mu_j\|_1 := \max_{s \in S_j} \sum_{s' \in S_j} |\mu_s^t(s') - \mu_j(s')|,$$

we define the *ε-mixing time* of the Markov chain as

$$T_{\text{mix}}^j(\varepsilon) := \min\{t \in \mathbb{N} \mid d_j(t) \leq \varepsilon\}.$$

Setting somewhat arbitrarily *the* mixing time of the chain to $T^j_{\text{mix}} := T^j_{\text{mix}}(\frac{1}{4})$, one can show (cf. eq. 4.36 in [10]) that

$$T^j_{\text{mix}}(\varepsilon) \leq \left\lceil \log_2 \tfrac{1}{\varepsilon} \right\rceil \cdot T^j_{\text{mix}}. \tag{1}$$

Finally, let $T_j(s, s')$ be the expected time it takes in arm j to reach s' when starting in s. We set the *diameter* of arm j to be $D_j := \max_{s,s' \in S_j} T_j(s, s')$.

3 Examples

Next we present a few examples that give insight into the nature of the problem and the difficulties in finding solutions. In particular, the examples demonstrate that (i) the optimal reward can be (much) bigger than the average reward of the best arm, (ii) the optimal policy does not maximize the immediate reward, (iii) the optimal policy cannot always be expressed in terms of arm indexes.

Example 1. In this example the average reward of each of the two arms of a bandit is $\frac{1}{2}$, but the reward of the optimal policy is close to $\frac{3}{4}$. Consider a two-armed bandit. Each arm has two possible states, 0 and 1, which are also the rewards. Underlying each of the two arms is a (two-state) Markov chain with transition matrix $\begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}$, where ϵ is small. Thus, a typical trajectory of each arm looks like this: 000000000001111111111111111000000000..., and the average reward for each arm is $\frac{1}{2}$. It is easy to see that the optimal policy starts with any arm, and then switches the arm whenever the reward is 0, and otherwise sticks to the same arm. The average reward is close to $\frac{3}{4}$ — much larger than the reward of each arm.

This example has a natural interpretation in terms of *cognitive radio*: two radio channels are available, each of which can be either busy (0) or available (1). A device can only sense (and use) one channel at a time, and one wants to maximize the amount of time the channel it tries to use is available.

Example 2. Consider the previous example, but with ϵ close to 1. Thus, a typical trajectory of each arm is now 01010101001010110..., and the optimal policy switches arms if the previous reward was 1 and stays otherwise.

Example 3. In this example the optimal policy does not maximize the immediate reward. Again, consider a two-armed bandit. Arm 1 is as in Example 1, and arm 2 provides Bernoulli i.i.d. rewards with probability $\frac{1}{2}$ of getting reward 1. The optimal policy (which knows the distributions) will sample arm 1 until it obtains reward 0, when it switches to arm 2. However, it will sample arm 1 again after some time t (depending on ϵ), and only switch back to arm 2 when the reward on arm 1 is 0. Note that whatever t is, the expected reward for choosing arm 1 will be strictly smaller than $\frac{1}{2}$, since the last observed reward was 0 and the limiting probability of observing reward 1 (when $t \to \infty$) is $\frac{1}{2}$. At the same time, the expected reward of the second arm is always $\frac{1}{2}$. Thus, the optimal policy will sometimes "explore" by pulling the arm with the smaller expected reward.

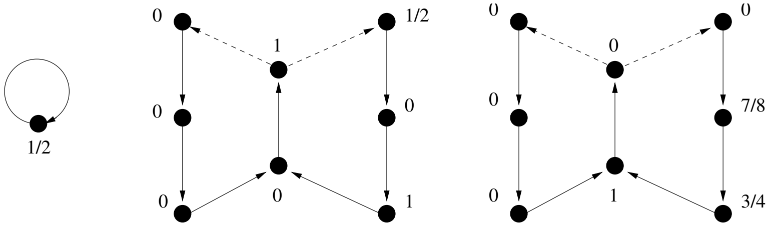


Fig. 1. *Example 4.* Dashed transitions are with probability $\frac{1}{2}$, others are deterministic with probability 1. Numbers are rewards in the respective state.

An intuitively appealing idea is to look for an optimal policy in an *index* form. That is, for each arm the policy maintains an index which is a function of time, states, and rewards *of this arm only.* At each time step, the policy samples the arm that has maximal index. This seems promising for at least two reasons: First, the distributions of the arms are assumed independent, so it may seem reasonable to evaluate them independently as well; second, this works in the i.i.d. case (e.g., the Gittins index [11] or UCB [12]). This idea also motivates the setting when just one out of two arms is Markov and the other is i.i.d., see e.g. [9]. Index policies for restless Markov bandits were also studied in [13]. Despite their intuitive appeal, in general, index policies are suboptimal.

Theorem 1. *For each index-based policy π there is a restless Markov bandit problem in which π behaves suboptimally.*

Proof. Consider the three bandits L (left), C (center), and R (right) in Figure 1, where C and R start in the 1 reward state. (Arms C and R can easily be made aperiodic by adding further sufficiently small transition probabilities.) Assume that C has been observed in the $\frac{1}{2}$ reward state one step before, while R has been observed in the 1 reward state three steps ago. The optimal policy will choose arm L which gives reward $\frac{1}{2}$ with certainty (C gives reward 0 with certainty, while R gives reward $\frac{7}{8}$ with probability $\frac{1}{2}$) and subsequently arms C and R. However, if arm C was missing, in the same situation, the optimal policy would choose R: Although the immediate expected reward is smaller than when choosing L, sampling R gives also information about the current state, which can earn reward $\frac{3}{4}$ a step later. Clearly, no index based policy will behave optimally in both settings. □

4 Main Results

Theorem 2. *Consider a restless bandit with K aperiodic arms having state spaces S_j, diameters D_j, and mixing times T^j_{mix} ($j = 1, \ldots, K$). Then with probability at least $1-\delta$ the regret of Algorithm 2 (presented in Section 5 below) after T steps is upper bounded by*

$$\mathrm{const} \cdot S \cdot T_{\mathrm{mix}}^{3/2} \cdot \textstyle\prod_{j=1}^{K}(4D_j) \cdot \max_i \log(D_i) \cdot \log^2\left(\frac{T}{\delta}\right) \cdot \sqrt{T},$$

where $S := \sum_{j=1}^{K} |S_j|$ is the total number of states and $T_{\mathrm{mix}} := \max_j T_{\mathrm{mix}}^j$ the maximal mixing time. Further, the dependence on T_{mix} can be eliminated to show that with probability at least $1-\delta$ the regret is bounded by

$$O\left(S \cdot \prod_{j=1}^{K}(4D_j) \cdot \max_i \log(D_i) \cdot \log^{7/2}\left(\tfrac{T}{\delta}\right) \cdot \sqrt{T}\right).$$

Remark 1. For periodic chains the bound of Theorem 2 has worse dependence on the state space, for details see Remark 5 in Section 7.

Theorem 3. *For any algorithm, any $K > 1$ and any $m \geq 1$ there is a K-armed restless bandit problem with a total number of $S := Km$ states, such that the regret after T steps is lower bounded by $\Omega(\sqrt{ST})$.*

Remark 2. While it is easy to see that lower bounds depend on the total number of states over all arms, the dependence on other parameters in our upper bound is not clear. For example, intuitively, while in the general MDP case one wrong step may cost up to D — the MDP's diameter [6] — steps to compensate for, here the Markov chains evolve independently of the learner's actions, and the upper bound's dependence on the diameter may be just an artefact of the proof.

5 Constructing the Algorithm

MDP Representation. We represent the setting as an MDP by recalling for each arm the last observed state and the number of time steps which have gone by since this last observation. Thus, each state of the MDP representation is of the form $(s_j, n_j)_{j=1}^K := (s_1, n_1, s_2, n_2, \dots, s_K, n_K)$ with $s_j \in S_j$ and $n_j \in \mathbb{N}$, meaning that each arm j has not been chosen for n_j steps when it was in state s_j. More precisely, $(s_j, n_j)_{j=1}^K$ is a state of the considered MDP if and only if (i) all n_j are distinct and (ii) there is a j with $n_j = 1$.[1] The action space of the MDP is $\{1, 2, \dots, K\}$, and the transition probabilities from a state $(s_j, n_j)_{j=1}^K$ are given by the n_j-step transition probabilities $p_j^{(n_j)}(s, s')$ of the Markov chain underlying the chosen arm j (these are defined by the matrix power of the single step transition probability matrix, i.e. $P_j^{n_j}$). That is, the probability for a transition from state $(s_j, n_j)_{j=1}^K$ to $(s'_j, n'_j)_{j=1}^K$ under action j is given by $p_j^{(n_j)}(s_j, s'_j)$ iff (i) $n'_j = 1$, (ii) $n'_\ell = n_\ell + 1$ and $s_\ell = s'_\ell$ for all $\ell \neq j$. All other transition probabilities are 0. Finally, the mean reward for choosing arm j in state $(s_j, n_j)_{j=1}^K$ is given by $\sum_{s \in S_j} p_j^{(n_j)}(s_j, s) \cdot r_j(s)$. This MDP representation has already been considered in [8].

Obviously, within T steps any policy can reach only states with $n_j \leq T$. Correspondingly, if we are interested in the regret within T steps, it will be sufficient to consider the finite sub-MDP consisting of states with $n_j \leq T$. We call this the *T-step representation* of the problem, and the regret will be measured with respect to the optimal policy in this T-step representation.

[1] Actually, one would need to add for each arm j with $|S_j| > 1$ a special state for not having sampled j so far. However, for the sake of simplicity we assume that in the beginning each arm is sampled once. The respective regret is negligible.

Algorithm 1. The colored UCRL2 algorithm

```
Input: Confidence parameter $\delta > 0$, aggregation parameter $\varepsilon > 0$, state space $S$,
action space $A$, coloring and translation functions, a bound $B$ on the size of the
support of transition probability distributions.
Initialization: Set $t := 1$, and observe the initial state $s_1$.
for episodes $k = 1, 2, \ldots$ do
    Initialize episode $k$:
    Set the start time of episode $k$, $t_k := t$. Let $N_k(c)$ be the number of times a state-action
    pair of color $c$ has been visited prior to episode $k$, and $v_k(c)$ the number of times a
    state-action pair of color $c$ has been visited in episode $k$. Compute estimates $\hat{r}_k(s,a)$
    and $\hat{p}_k(s''|s,a)$ for rewards and transition probabilities, using all samples from
    state-action pairs of the same color $c(s,a)$, respectively.
    Compute policy $\tilde{\pi}_k$:
    Let $\mathcal{M}_k$ be the set of plausible MDPs with rewards $\tilde{r}(s,a)$ and transition
    probabilities $\tilde{p}(\cdot|s,a)$ satisfying
        $\left|\tilde{r}(s,a) - \hat{r}_k(s,a)\right| \leq \varepsilon + \sqrt{\frac{7\log(2Ct_k/\delta)}{2\max\{1, N_k(c(s,a))\}}}$,   (2)
        $\left\|\tilde{p}(\cdot|s,a) - \hat{p}_k(\cdot|s,a)\right\|_1 \leq \varepsilon + \sqrt{\frac{56B\log(4Ct_k/\delta)}{\max\{1, N_k(c(s,a))\}}}$,   (3)
    where $C$ is the number of distinct colors. Let $\rho(\pi, M)$ be the average reward of a policy
    $\pi : S \to A$ on an MDP $M \in \mathcal{M}_k$. Choose (e.g. by extended value iteration [6]) an
    optimal policy $\tilde{\pi}_k$ and an optimistic $\tilde{M}_k \in \mathcal{M}_k$ such that
        $\rho(\tilde{\pi}_k, \tilde{M}_k) = \max\{\rho(\pi, M) \,|\, \pi : S \to A,\ M \in \mathcal{M}_k\}$.   (4)
    Execute policy $\tilde{\pi}_k$:
    while $v_k(c(s_t, \tilde{\pi}_k(s_t))) < \max\{1, N_k(c(s_t, \tilde{\pi}_k(s_t)))\}$ do
    ▷ Choose action $a_t = \tilde{\pi}_k(s_t)$, obtain reward $r_t$, and observe next state $s_{t+1}$.
    ▷ Set $t := t + 1$.
    end while
end for
```

Structure of the MDP Representation. The MDP representation of our problem has some special structural properties. In particular, rewards and transition probabilities for choosing arm j only depend on the state of arm j, i.e. s_j and n_j. Moreover, the support for each transition probability distribution is bounded, and for $n_j \geq T^j_{\text{mix}}(\varepsilon)$ the transition probability distribution will be close to the stationary distribution of arm j. Thus, one could reduce the T-step representation further by aggregating states[2] $(s_j, n_j)_{j=1}^K$, $(s'_j, n'_j)_{j=1}^K$ whenever $n_j, n'_j \geq T^j_{\text{mix}}(\varepsilon)$ and $s_\ell = s'_\ell$, $n_\ell = n'_\ell$ for $\ell \neq j$. The rewards and transition probability distributions of aggregated states are ε-close, so that the error by

[2] Aggregation of states $s_1, \ldots, s_n$ means that these states are replaced by a new state s_{agg} inheriting rewards and transition probabilities from an arbitrary s_i (or averaging over all s_j). Transitions to this state are set to $p(s_{\text{agg}}|s,a) := \sum_j p(s_j|s,a)$.

Algorithm 2. The restless bandits algorithm

```
**Input:** Confidence parameter $\delta > 0$, the number of states $S_j$ and mixing time $T^j_{\rm mix}$
of each arm $j$, horizon $T$.
▷ Choose $\varepsilon = 1/\sqrt{T}$ and execute colored UCRL2 (with confidence parameter $\delta$) on
the $\varepsilon$-structured MDP described in the "coloring" paragraph at the end of Section 5.
```

aggregation can be bounded by results given in [14]. While this is helpful for approximating the problem when all parameters are known, it cannot be used directly when learning, since the observations in the aggregated states do not correspond to an MDP anymore. Thus, while standard reinforcement learning algorithms are still applicable, there are no theoretical guarantees for them.

ε-structured MDPs and Colored UCRL2. In the following, we exploit the special structure of the MDP representation. We generalize some of its structural properties in the following definition.

Definition 1. *An ε-structured MDP is an MDP with finite state space S, finite action space A, transition probability distributions $p(\cdot|s,a)$, mean rewards $r(s,a) \in [0,1]$, and a coloring function $c : S \times A \to \mathcal{C}$, where $\mathcal{C}$ is a set of colors. Further, for each two pairs (s,a), $(s',a') \in S \times A$ with $c(s,a) = c(s',a')$ there is a bijective translation function $\phi_{s,a,s',a'} : S \to S$ such that $\sum_{s''} \left|p(s''|s,a) - p(\phi_{s,a,s',a'}(s'')|s',a')\right| < \varepsilon$ and $|r(s,a) - r(s',a')| < \varepsilon$.*

If there are states s, s' in an ε-structured MDP such that $c(s,a) = c(s',a)$ for all actions a and the associated translation function $\phi_{s,a,s',a}$ is the identity, we may aggregate the states (cf. footnote 2). We call the MDP in which all such states are aggregated the *aggregated ε-structured MDP*.

For learning in ε-structured MDPs we consider a modification of the UCRL2 algorithm of [6]. The *colored* UCRL2 algorithm is shown in Figure 1. As the original UCRL2 algorithm it maintains confidence intervals for rewards and transition probabilities which define a set of plausible MDPs $\mathcal{M}_k$. In each episode k, the algorithm chooses an optimistic MDP $\tilde{M}_k \in \mathcal{M}_k$ and an optimal policy which maximize the average reward, cf. (4). Colored UCRL2 calculates estimates from all samples of state-action pairs of the same color, and works with respectively adapted confidence intervals and a corresponding adapted episode termination criterion. Basically, an episode ends when for some color c the number of visits in state-action pairs of color c has doubled.

Coloring the T-Step Representation. Now, we can turn the T-step representation into an ε-structured MDP, assigning the same color to state-action pairs where the chosen arm is in the same state, that is, $c((s_i,n_i)_{i=1}^K, j) = c((s'_i,n'_i)_{i=1}^K, j')$ iff $j = j'$, $s_j = s'_j$, and either $n_j = n'_j$ or $n_j, n'_j \geq T^j_{\rm mix}(\varepsilon)$. The translation functions are chosen accordingly. This ε-structured MDP can be learned with colored UCRL2, see Algorithm 2, our restless bandits algorithm. (The dependence on the horizon T and the mixing times $T^j_{\rm mix}$ as input parameters can be eliminated, cf. the proof of Theorem 2 in Section 7.)

6 Regret Bounds for Colored UCRL2

The following is a generalization of the regret bounds for UCRL2 to ε-structured MDPs. The theorem gives improved (with respect to UCRL2) bounds if there are only a few parameters to estimate in the MDP to learn. Recall that the *diameter* of an MDP is the maximal expected transition time between any two states (choosing an appropriate policy), cf. [6].

Theorem 4. *Let M be an ε-structured MDP with finite state space S, finite action space A, transition probability distributions $p(\cdot|s,a)$, mean rewards $r(s,a) \in [0,1]$, coloring function c and associate translation functions. Assume the learner has complete knowledge of state-action pairs $\Psi_K \subseteq S \times A$, while the state-action pairs in $\Psi_U := S \times A \setminus \Psi_K$ are unknown and have to be learned. However, the learner knows c and all associate translation functions as well as an upper bound B on the size of the support of each transition probability distribution in Ψ_U. Then with probability at least $1-\delta$, after any T steps colored* UCRL2 [3] *gives regret upper bounded by*

$$42 D_\varepsilon \sqrt{B C_U T \log\left(\tfrac{T}{\delta}\right)} + \varepsilon(D_\varepsilon + 2)T,$$

where C_U is the total number of colors for states in Ψ_U, and D_ε is the diameter of the aggregated ε-structured MDP.

The proof of this theorem is given in the appendix.

Remark 3. For $\varepsilon = 0$, one can also obtain logarithmic bounds analogously to Theorem 4 of [6]. With no additional information for the learner one gets the original UCRL2 bounds (with a slightly larger constant), trivially choosing B to be the number of states and assigning each state-action pair an individual color.

7 Proofs

We start with bounding the diameter in the aggregated ε-structured MDP.

Lemma 1. *For $\varepsilon \leq 1/4$, the diameter D_ε in the aggregated ε-structured MDP can be upper bounded by $2\left\lceil \log_2(4 \max_j D_j)\right\rceil \cdot T_{\text{mix}}(\varepsilon) \cdot \prod_{j=1}^{K}(4D_j)$, where we set $T_{\text{mix}}(\varepsilon) := \max_j T^j_{\text{mix}}(\varepsilon)$.*

Proof. Let μ_j be the stationary distribution of arm j. It is well-known that the expected *first return time* $\tau_j(s)$ in state s satisfies $\mu_j(s) = 1/\tau_j(s)$. Set $\tau_j := \max_s \tau_j(s)$, and $\tau := \max_j \tau_j$. Then, $\tau_j \leq 2D_j$.

Now consider the following scheme to reach a given state $(s_j, n_j)_{j=1}^K$: First, order the states (s_j, n_j) descendingly with respect to n_j. Thus, assume that $n_{j_1} > n_{j_2} > \ldots > n_{j_K} = 1$. Take $T_{\text{mix}}(\varepsilon)$ samples from arm j_1. (Then each arm

[3] For the sake of simplicity the algorithm was given for the case $\Psi_K = \varnothing$. It is obvious how to extend the algorithm when some parameters are known.

will be ε-close to the stationary distribution, and the probability of reaching the right state s_{j_i} when sampling arm j_i afterwards is at least $\mu_{j_i}(s_{j_i}) - \varepsilon$.) Then sample each arm $j_2, j_3, \ldots$ exactly $n_{j_{i-1}} - n_{j_i}$ times.

We first show the lemma for $\varepsilon \leq \mu_0 := \min_{j,s} \mu_j(s)/2$. As observed before, for each arm j_i the probability of reaching the right state s_{j_i} is at least $\mu_{j_i}(s_{j_i}) - \varepsilon \geq \mu_{j_i}(s_{j_i})/2$. Consequently, the expected number of restarts of the scheme necessary to reach a particular state $(s_j, n_j)_{j=1}^K$ is upper bounded by $\prod_{j=1}^K 2/\mu_j(s_j)$. As each trial takes at most $2T_{\mathrm{mix}}(\varepsilon)$ steps, recalling that $1/\mu_j(s) = \tau_j(s) \leq 2D_j$ proves the bound for $\varepsilon \leq \mu_0$.

Now assume that $\varepsilon > \mu_0$. Since $D_\varepsilon \leq D_{\varepsilon'}$ for $\varepsilon > \varepsilon'$ we obtain a bound of $2T_{\mathrm{mix}}(\varepsilon') \prod_{j=1}^K (4D_j)$ with $\varepsilon' := \mu_0 = 1/2\tau$. By (1), we have $T_{\mathrm{mix}}(\varepsilon') \leq \lceil \log_2(1/\varepsilon') \rceil \, T_{\mathrm{mix}}(1/4) \leq \lceil \log_2(4\tau) \rceil \, T_{\mathrm{mix}}(\varepsilon)$, which proves the lemma. □

Proof of Theorem 2. Note that in each arm j the support of the transition probability distribution is upper bounded by $|S_j|$. Hence, Theorem 4 with $C_U = \sum_{j=1}^K |S_j| \, T^j_{\mathrm{mix}}(\varepsilon)$ and $B = \max_j |S_j|$ shows that the regret is bounded by $42D_\varepsilon \sqrt{\max_i |S_i| \cdot \sum_{j=1}^K |S_j| \cdot T^j_{\mathrm{mix}}(\varepsilon) \cdot T \log\left(\frac{T}{\delta}\right)} + \varepsilon(D_\varepsilon + 2)T$ with probability $\geq 1 - \delta$. Since $\varepsilon = 1/\sqrt{T}$, this proves the first bound by Lemma 1 and recalling (1).

If the horizon T is not known, guessing T using the doubling trick (i.e., executing the algorithm for $T = 2^i$ with confidence parameter $\delta/2^i$ in rounds $i = 1, 2, \ldots$) achieves the bound given in Theorem 2 with worse constants.

Similarly, if T_{mix} is unknown, one can perform the algorithm in rounds $i = 1, 2, \ldots$ of length 2^i with confidence parameter $\delta/2^i$, choosing an increasing function $a(t)$ to guess an upper bound on T_{mix} at the beginning t of each round. This gives a bound of order $a(T)^{3/2}\sqrt{T}$ with a corresponding additive constant. In particular, choosing $a(t) = \log t$ the regret is bounded by $O\big(S \cdot \prod_{j=1}^K (4D_j) \cdot \max_i \log(D_i) \cdot \log^{7/2}(T/\delta) \cdot \sqrt{T}\big)$ with probability $\geq 1 - \delta$. □

Remark 4. Whereas it is not easy to obtain upper bounds on the mixing time in general, for *reversible* Markov chains T_{mix} can be linearly upper bounded by the diameter, cf. Lemma 15 in Chapter 4 of [15]. While it is possible to compute an upper bound on the diameter of a Markov chain from samples of the chain, we did not succeed in deriving any useful results on the quality of such bounds.

Remark 5. Periodic Markov chains do not converge to a stationary distribution. However taking into account the period of the arms, one can generalize our results to the periodic case. Considering in an m-periodic Markov chain the m-step transition probabilities given by the matrix P^m, one obtains m distinct aperiodic chains (depending on the initial state) each of which converges to a stationary distribution with respective mixing times. The maximum over these mixing times can be considered to be *the* mixing time of the chain.

Thus, instead of aggregating states (s_j, n_j), (s'_j, n'_j) with $n_j, n'_j \geq T^j_{\mathrm{mix}}(\varepsilon)$ as in the case of aperiodic chains, one aggregates them only if additionally $n_j \equiv n'_j \mod m_j$. If the periods m_j are not known to the learner, one can use the least common denominator of $1, 2, \ldots, |S_j|$ as period. Since by the prime number theorem the latter is exponential in $|S_j|$, the obtained results for periodic arms show worse dependence on the number of states. (Concerning the proof of Lemma 1 the sampling scheme has to be slightly adapted so that one samples in the right period when trying to reach a particular state.)

Proof of Theorem 3. Consider K arms all of which are deterministic cycles of length m and hence m-periodic. Then the learner faces m distinct learning problems with K arms, each of which can be made to force regret of order $\Omega(\sqrt{KT/m})$ in the T/m steps the learner deals with the problem [4]. Overall, this gives the claimed bound of $\Omega(\sqrt{mKT}) = \Omega(\sqrt{ST})$. Adding a sufficiently small probability (with respect to the horizon T) of staying in some state of each arm, one obtains the same bounds for aperiodic arms. □

8 Extensions and Outlook

Unknown State Space. If (the size of) the state space of the individual arms is unknown, some additional exploration of each arm will sooner or later determine the state space. Thus, we may execute our algorithm on the known state space where between two episodes we sample each arm until all known states have been sampled at least once. The additional exploration is upper bounded by $O(\log T)$, as there are only $O(\log T)$ many episodes, and the time of each exploration phase can be bounded with known results. That is, the expected number of exploration steps needed until all states of an arm j have been observed is upper bounded by $D_j \log(3|S_j|)$ (cf. Theorem 11.2 of [10]), while the deviation from the expectation can be dealt with by Markov inequality or results from [16]. That way, one obtains bounds as in Theorem 2 for the case of unknown state space.

Improving the Bounds. All parameters considered, there is still a large gap between the lower and the upper bound on the regret. As a first step, it would be interesting to find out whether the dependence on the diameter of the arms is necessary. Also, the current regret bounds do not make use of the interdependency of the transition probabilities in the Markov chains and treat n-step and n'-step transition probabilities independently. Finally, a related open question is how to obtain estimates and upper bounds on mixing times.

More General Models. After considering bandits with i.i.d. and Markov arms, the next natural step is to consider more general time-series distributions. Generalizations are not straightforward: already for the case of Markov chains of order (or memory) 2 the MDP representation of the problem (Section 5) breaks down, and so the approach taken here cannot be easily extended. Stationary ergodic distributions are an interesting more general case, for which the first question is whether it is possible to obtain asymptotically sublinear regret.

Acknowledgments. This research was funded by the Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council and FEDER (Contrat de Projets Etat Region CPER 2007-2013), ANR projects EXPLO-RA (ANR-08-COSI-004), Lampada (ANR-09-EMER-007) and CoAdapt, and by the European Community's FP7 Program under grant agreements n° 216886 (PASCAL2) and n° 270327 (CompLACS). The first author is currently funded by the Austrian Science Fund (FWF): J 3259-N13.

References

[1] Lai, T.L., Robbins, H.: Asymptotically efficient adaptive allocation rules. Adv. in Appl. Math. 6, 4–22 (1985)

[2] Akyildiz, I.F., Lee, W.Y., Vuran, M.C., Mohanty, S.: A survey on spectrum management in cognitive radio networks. IEEE Commun. Mag. 46(4), 40–48 (2008)

[3] Anantharam, V., Varaiya, P., Walrand, J.: Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays, part II: Markovian rewards. IEEE Trans. Automat. Control 32(11), 977–982 (1987)

[4] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SIAM J. Comput. 32, 48–77 (2002)

[5] Audibert, J.-Y., Bubeck, S.: Minimax policies for adversarial and stochastic bandits. In: COLT 2009. Proc. 22nd Annual Conf. on Learning Theory, pp. 217–226 (2009)

[6] Jaksch, T., Ortner, R., Auer, P.: Near-optimal regret bounds for reinforcement learning. J. Mach. Learn. Res. 11, 1563–1600 (2010)

[7] Bartlett, P.L., Tewari, A.: REGAL: A regularization based algorithm for reinforcement learning in weakly communicating MDPs. In: Proc. 25th Conference on Uncertainty in Artificial Intelligence, UAI 2009, pp. 35–42. AUAI Press (2009)

[8] Tekin, C., Liu, M.: Adaptive learning of uncontrolled restless bandits with logarithmic regret. In: 49th Annual Allerton Conference, pp. 983–990. IEEE (2011)

[9] Filippi, S., Cappe, O., Garivier, A.: Optimally sensing a single channel without prior information: The tiling algorithm and regret bounds. IEEE J. Sel. Topics Signal Process. 5(1), 68–76 (2011)

[10] Levin, D.A., Peres, Y., Wilmer, E.L.: Markov chains and mixing times. American Mathematical Society (2006)

[11] Gittins, J.C.: Bandit processes and dynamic allocation indices. J. R. Stat. Soc. Ser. B Stat. Methodol. 41(2), 148–177 (1979)

[12] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multi-armed bandit problem. Mach. Learn. 47, 235–256 (2002)

[13] Whittle, P.: Restless bandits: Activity allocation in a changing world. J. Appl. Probab. 25, 287–298 (1988)

[14] Ortner, R.: Pseudometrics for State Aggregation in Average Reward Markov Decision Processes. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS (LNAI), vol. 4754, pp. 373–387. Springer, Heidelberg (2007)

[15] Aldous, D.J., Fill, J.: Reversible Markov Chains and Random Walks on Graphs (in preparation), http://www.stat.berkeley.edu/~aldous/RWG/book.html

[16] Aldous, D.J.: Threshold limits for cover times. J. Theoret. Probab. 4, 197–211 (1991)

A Proof of Theorem 4

Splitting into Episodes. We follow the proof of Theorem 2 in [6]. First, as shown in Section 4.1 of [6], setting $\Delta_k := \sum_{s,a} v_k(s,a)(\rho^* - r(s,a))$ with probability at least $1 - \frac{\delta}{12T^{5/4}}$ the regret after T steps can be upper bounded by

$$\textstyle\sum_{k=1}^m \Delta_k + \sqrt{\frac{5}{8} T \log\left(\frac{8T}{\delta}\right)}\,. \tag{5}$$

Failing Confidence Intervals. Concerning the regret with respect to the true MDP M being not contained in the set of plausible MDPs $\mathcal{M}_k$, we cannot use the same argument (that is, Lemma 17 in Appendix C.1) as in [6], since the random variables we consider for rewards and transition probabilities are independent, yet not identically distributed.

Instead, fix a state-action pair (s,a), let $S(s,a)$ be the set of states s' with $p(s'|s,a) > 0$ and recall that $\hat{r}(s,a)$ and $\hat{p}(\cdot|s,a)$ are the estimates for rewards and transition probabilities calculated from all samples of state-action pairs of the same color $c(s,a)$. Now assume that at step t there have been $n > 0$ samples of state-action pairs of color $c(s,a)$ and that in the i-th sample action a_i has been chosen in state s_i and a transition to state s'_i has been observed ($i = 1,\dots,n$). Then

$$\begin{aligned}
&\Big\|\hat{p}(\cdot|s,a) - \mathbb{E}[\hat{p}(\cdot|s,a)]\Big\|_1 = \sum_{s'\in S(s,a)} \Big|\hat{p}(s'|s,a) - \mathbb{E}[\hat{p}(s'|s,a)]\Big| \\
&\le \sup_{x\in\{0,1\}^{|S(s,a)|}} \sum_{s'\in S(s,a)} \Big(\hat{p}(s'|s,a) - \mathbb{E}[\hat{p}(s'|s,a)]\Big)\, x(s') \\
&= \sup_{x\in\{0,1\}^{|S(s,a)|}} \tfrac{1}{n}\sum_{i=1}^{n} \Big(x(\phi_{s_i,a_i,s,a}(s'_i)) - \sum_{s'} p(s'|s_i,a_i)\cdot x(\phi_{s_i,a_i,s,a}(s'))\Big).
\end{aligned} \tag{6}$$

For fixed $x \in \{0,1\}^{|S(s,a)|}$, $X_i := x(\phi_{s_i,a_i,s,a}(s'_i)) - \sum_{s'} p(s'|s_i,a_i)\cdot x(\phi_{s_i,a_i,s,a}(s'))$ is a martingale difference sequence with $|X_i| \le 2$, so that by Azuma-Hoeffding inequality (e.g., Lemma 10 in [6]), $\Pr\{\sum_{i=1}^n X_i \ge \theta\} \le \exp(-\theta^2/8n)$ and in particular

$$\Pr\Big\{ \textstyle\sum_{i=1}^n X_i \ge \sqrt{56Bn\log\big(\frac{4tC_U}{\delta}\big)}\Big\} \le \Big(\frac{\delta}{4tC_U}\Big)^{7B} < \frac{\delta}{2^B 20t^7 C_U}.$$

Recalling that by assumption $|S(s,a)| \le B$, a union bound over all sequences $x \in \{0,1\}^{|S(s,a)|}$ then shows from (6) that

$$\Pr\Big\{\Big\|\hat{p}(\cdot|s,a) - \mathbb{E}[\hat{p}(\cdot|s,a)]\Big\|_1 \ge \sqrt{\tfrac{56B}{n}\log\left(4C_U t/\delta\right)}\Big\} \le \tfrac{\delta}{20t^7 C_U}. \tag{7}$$

Concerning the rewards, as in the proof of Lemma 17 in Appendix C.1 of [6] — but now using Hoeffding for independent and not necessarily identically distributed random variables — we have that

$$\Pr\Big\{\big|\hat{r}(s,a) - \mathbb{E}[\hat{r}(s,a)]\big| \ge \sqrt{\tfrac{7}{2n}\log\left(2C_U t/\delta\right)}\Big\} \le \tfrac{\delta}{60t^7 C_U}. \tag{8}$$

A union bound over all t possible values for n and all C_U colors of states in Ψ_U shows that the confidence intervals in (7) and (8) hold with probability at least $1 - \frac{\delta}{15t^6}$ for the actual counts $N(c(s,a))$ and all state-action pairs (s,a). (Note that equations (7) and (8) are the same for state-action pairs of the same color.)

By linearity of expectation, $\mathbb{E}[\hat{r}(s,a)]$ can be written as $\frac{1}{n}\sum_{i=1}^n r(s_i,a_i)$ for the sampled state-action pairs (s_i,a_i). Since the (s_i,a_i) are assumed to have the same color $c(s,a)$, it holds that $|r(s_i,a_i) - r(s,a)| < \varepsilon$ and hence $|\mathbb{E}[\hat{r}(s,a)] - r(s,a)| < \varepsilon$. Similarly, $\big\|\mathbb{E}[\hat{p}(\cdot|s,a)] - p(\cdot|s,a)\big\|_1 < \varepsilon$. Together with (7) and (8) this shows that with probability at least $1-\frac{\delta}{15t^6}$ for all state-action pairs (s,a)

$$\big|\hat{r}(s,a) - r(s,a)\big| < \varepsilon + \sqrt{\tfrac{7}{2n}\log\left(2C_U t/\delta\right)}, \tag{9}$$

$$\Big\|\hat{p}(\cdot|s,a) - p(\cdot|s,a)\Big\|_1 < \varepsilon + \sqrt{\tfrac{56B}{n}\log\left(4C_U t/\delta\right)}. \tag{10}$$

Thus, the true MDP is contained in the set of plausible MDPs $\mathcal{M}(t)$ at step t with probability at least $1-\frac{\delta}{15t^6}$, just as in Lemma 17 of [6]. The argument that

$$\textstyle\sum_{k=1}^{m} \Delta_k \mathbb{1}_{M\notin\mathcal{M}_k} \;\leq\; \sqrt{T} \tag{11}$$

with probability at least $1-\frac{\delta}{12T^{5/4}}$ then can be taken without any changes from Section 4.2 of [6].

Episodes with $\boldsymbol{M \in \mathcal{M}_k}$. Now assuming that the true MDP M is in $\mathcal{M}_k$, we first reconsider extended value iteration. In Section 4.3.1 of [6] it is shown that for the state values $u_i(s)$ in the i-th iteration it holds that $\max_s u_i(s) - \min_s u_i(s) \leq D$, where D is the diameter of the MDP. Now we want to replace D with the diameter D_ε of the aggregated MDP. For this, first note that for any two states s, s' which are aggregated we have by definition of the aggregated MDP that $u_i(s) = u_i(s')$. As it takes at most D_ε steps on average to reach any aggregated state, repeating the argument of Section 4.3.1 of [6] shows that

$$\max_s u_i(s) - \min_s u_i(s) \leq D_\varepsilon. \tag{12}$$

Let $\tilde{\boldsymbol{P}}_k := \big(\tilde{p}_k(s'|s,\tilde{\pi}_k(s))\big)_{s,s'}$ be the transition matrix of $\tilde{\pi}_k$ on $\tilde{M}_k$, and $\boldsymbol{v}_k := \big(v_k\big(s,\tilde{\pi}_k(s)\big)\big)_s$ the row vector of visit counts in episode k for each state and the corresponding action chosen by $\tilde{\pi}_k$. Then as shown in Sect. 4.3.1 of [6][4]

$$\Delta_k \leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + \sum_{s,a} v_k(s,a)\big(\tilde{r}_k(s,a) - r(s,a)\big),$$

where $\boldsymbol{w}_k$ is the normalized state value vector with $w_k(s) := u(s) - (\min_s u(s) - \max_s u(s))/2$, so that $\|\boldsymbol{w}_k\| \leq \frac{D_\varepsilon}{2}$. Now for $(s,a) \in \Psi_K$ we have $\tilde{r}_k(s,a) = r(s,a)$, while for $(s,a) \in \Psi_U$ the term $\tilde{r}_k(s,a) - r(s,a) \leq |\tilde{r}_k(s,a) - \hat{r}_k(s,a)| + |r(s,a) - \hat{r}_k(s,a)|$ is bounded according to (2) and (9), as we assume that $\tilde{M}_k, M \in \mathcal{M}_k$. Summarizing state-action pairs of the same color we get

$$\Delta_k \leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + 2\sum_{c\in C(\Psi_U)} v_k(c)\cdot\left(\varepsilon + \sqrt{\tfrac{7\log(2C_U t_k/\delta)}{2\max\{1,N_k(c)\}}}\right),$$

where $C(\Psi_U)$ is the set of colors of state-action pairs in Ψ_U. Let T_k be the length of episode k. Then noting that $N_k'(c) := \max\{1, N_k(c)\} \leq t_k \leq T$ we get

$$\Delta_k \leq \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k + 2\varepsilon T_k + \sqrt{14\log\left(\tfrac{2C_U T}{\delta}\right)}\sum_{c\in C(\Psi_U)}\frac{v_k(c)}{\sqrt{N_k'(c)}}. \tag{13}$$

[4] Here we neglect the error by value iteration explicitly considered in Sect. 4.3.1 of [6].

The True Transition Matrix. Let $\boldsymbol{P}_k := \big(p(s'|s,\tilde{\pi}_k(s))\big)_{s,s'}$ be the transition matrix of $\tilde{\pi}_k$ in the true MDP M. We split

$$\boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{I}\big)\boldsymbol{w}_k = \boldsymbol{v}_k\big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k\big)\boldsymbol{w}_k + \boldsymbol{v}_k\big(\boldsymbol{P}_k - \boldsymbol{I}\big)\boldsymbol{w}_k. \tag{14}$$

By assumption $\tilde{M}_k, M \in \mathcal{M}_k$, so that using (3) and (10) the first term in (14) can be bounded by (cf. Section 4.3.2 of [6])

$$\begin{aligned}
\boldsymbol{v}_k\ \big(\tilde{\boldsymbol{P}}_k - \boldsymbol{P}_k\big)\boldsymbol{w}_k\ &\leq\ \sum_{s,a} v_k(s,a)\cdot\big\|\tilde{p}_k(\cdot|s,a) - p(\cdot|s,a)\big\|_1\cdot\|\boldsymbol{w}_k\|_\infty \\
&\leq 2\sum_{c\in C(\Psi_U)} v_k(c)\cdot\Big(\varepsilon + \sqrt{\tfrac{56B\log(4C_UT/\delta)}{N'_k(c)}}\Big)\cdot\tfrac{D_\varepsilon}{2} \\
&\leq \varepsilon D_\varepsilon T_k + D_\varepsilon\sqrt{56B\log\big(\tfrac{2C_UT}{\delta}\big)}\sum_{c\in C(\Psi_U)}\frac{v_k(c)}{\sqrt{N'_k(c)}},
\end{aligned} \tag{15}$$

since — as for the rewards — the contribution of state-action pairs in Ψ_K is 0.

Concerning the second term in (14), as shown in Section 4.3.2 of [6] one has with probability at least $1-\frac{\delta}{12T^{5/4}}$

$$\sum_{k=1}^m \boldsymbol{v}_k\ (\boldsymbol{P}_k - \boldsymbol{I})\boldsymbol{w}_k\mathbb{1}_{M\in\mathcal{M}_k} \leq D_\varepsilon\sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)} + D_\varepsilon\, C_U\log_2\big(\tfrac{8T}{C_U}\big), \tag{16}$$

where m is the number of episodes, and the bound $m \leq C_U\log_2(8T/C_U)$ used to obtain (16) is derived analogously to Appendix C.2 of [6].

Summing over Episodes with $\boldsymbol{M\in\mathcal{M}_k}$. To conclude, we sum (13) over all episodes with $M\in\mathcal{M}_k$, using (14), (15), and (16), which yields that with probability at least $1-\frac{\delta}{12T^{5/4}}$

$$\begin{aligned}
\sum_{k=1}^m \Delta_k\mathbb{1}_{M\in\mathcal{M}_k} &\leq D_\varepsilon\sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)} + D_\varepsilon\, C_U\log_2\big(\tfrac{8T}{C_U}\big) + \varepsilon(D_\varepsilon+2)T \\
&+\left(D_\varepsilon\sqrt{56B\log\big(\tfrac{2C_UBT}{\delta}\big)} + \sqrt{14\log\big(\tfrac{2C_UT}{\delta}\big)}\right)\sum_{k=1}^m\sum_{c\in C(\Psi_U)}\frac{v_k(c)}{\sqrt{N'_k(c)}}.
\end{aligned} \tag{17}$$

As in Sect. 4.3.3 and Appendix C.3 of [6], one obtains $\sum_{c\in C(\Psi_U)}\sum_k\frac{v_k(c)}{\sqrt{N'_k(c)}} \leq \big(\sqrt{2}+1\big)\sqrt{C_UT}$. Thus, evaluating (5) by summing Δ_k over all episodes, by (11) and (17) the regret is upper bounded with probability $\geq 1-\frac{\delta}{4T^{5/4}}$ by

$$\begin{aligned}
&\sum_{k=1}^m \Delta_k\mathbb{1}_{M\notin\mathcal{M}_k} + \sum_{k=1}^m \Delta_k\mathbb{1}_{M\in\mathcal{M}_k} + \sqrt{\tfrac{5}{8}T\log\big(\tfrac{8T}{\delta}\big)} \\
&\leq \sqrt{\tfrac{5}{8}T\log\big(\tfrac{8T}{\delta}\big)} + \sqrt{T} + D_\varepsilon\sqrt{\tfrac{5}{2}T\log\big(\tfrac{8T}{\delta}\big)} + D_\varepsilon\, C_U\log_2\Big(\tfrac{8T}{C_U}\Big) \\
&+\varepsilon(D_\varepsilon+2)T + 3\big(\sqrt{2}+1\big)D_\varepsilon\sqrt{14BC_UT\log\big(\tfrac{2C_UBT}{\delta}\big)}\,.
\end{aligned}$$

Further simplifications as in Appendix C.4 of [6] finish the proof. □

Minimax Number of Strata for Online Stratified Sampling Given Noisy Samples

Alexandra Carpentier and Rémi Munos

INRIA Lille Nord-Europe,
40 avenue Halley, 59650 Villeneuve d'Ascq, France
{alexandra.carpentier,remi.munos}@inria.fr
https://sequel.lille.inria.fr/

Abstract. We consider the problem of online stratified sampling for Monte Carlo integration of a function given a finite budget of n noisy evaluations to the function. More precisely we focus on the problem of choosing the number of strata K as a function of the numerical budget n. We provide asymptotic and finite-time results on how an oracle that knows the smoothness of the function would choose the number of strata optimally. In addition we prove a *lower bound* on the learning rate for the problem of stratified Monte-Carlo. As a result, we are able to state, by improving the bound on its performance, that algorithm MC-UCB, defined in [1], is minimax optimal both in terms of the number of samples n and the number of strata K, up to a log factor. This enables to deduce a minimax optimal bound on the difference between the performance of the estimate output by MC-UCB, and the performance of the estimate output by the best oracle static strategy, on the class of Hölder continuous functions, and up to a log factor.

Keywords: Bandit Theory, Online learning, Stratified sampling, Monte Carlo integration, Regret bounds.

Introduction

The objective of this paper is to provide an efficient strategy for Monte-Carlo integration of a function f over a domain $[0,1]^d$. We assume that we can query the function n times. Querying the function at a time t and at a point $x_t \in [0,1]^d$ provides a noisy sample[1]

$$f(x_t) + s(x_t)\epsilon_t, \tag{1}$$

where ϵ_t is an independent noise drawn from ν_{x_t} and $s \geq 0$ is a function on $[0,1]^d$. Here ν_x is a distribution with mean 0, variance 1 and whose shape may depend on x. This model is actually very general (see Section 1).

Stratified sampling is a well-known strategy to reduce the variance of the estimate of the integral of f, when compared to the variance of the estimate provided by crude Monte-Carlo. The principle is to partition the domain in K

[1] It is the usual model for regression in heterocedastic noise. We emphasize the standard deviation $s(x)$ of the noise at x, in the expression of the noise, since this quantity is very relevant.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 229–244, 2012.

subsets called *strata* and then to sample in each stratum (see [11][Subsection 5.5] or [6]). If the variances of the samples in the strata are known, there exists an optimal static allocation strategy which allocates the number of samples in each stratum proportionally to the measure of the stratum times the standard deviation in the stratum (see Equation 3 for the variance of the resulting estimate). We refer to this allocation as optimal oracle strategy for a given partition. In the case that the variations of f and the standard deviation of the noise s are unknown, it is not possible to adopt this strategy.

Consider first that the partition of the space is fixed. A way around this problem is to estimate the variations of the function and the amount of noise on the function in the strata *online* (exploration) while allocating the samples according to the estimated optimal oracle strategy (exploitation). This setting is considered in [3, 8, 1]. In the long version [2] of the last paper, the authors describe the MC-UCB algorithm which is based on Upper-Confidence-Bounds (UCB) on the standard deviation. They provide upper bounds for the difference between the mean-squared error (w.r.t. the integral of f) of the estimate provided by MC-UCB and the mean-squared error of the estimate provided by the optimal oracle strategy (optimal oracle variance). The algorithm performs almost as well as the optimal oracle strategy. However, the authors of [2] do not verify nor assess the optimality of their algorithm. As a matter of fact, no lower bound on the rate of convergence (to the oracle optimal strategy) for the problem of stratified Monte-Carlo exists, to the best of our knowledge. Still in the same paper [2], the authors do not discuss how to *stratify* the space. In particular, they do not pose the problem of what an *optimal* partition of the space is, and do not try to answer on whether it is possible or not to achieve this.

The next step is thus to efficiently design the partition. There are some interesting papers on that topic such as [7, 10, 4]. The recent, state of the art, work of [4] describes a strategy that samples *asymptotically* almost as efficiently as the optimal oracle strategy, and at the same time adapts the direction and number of the strata online. This is a very difficult problem. The authors do not provide proofs of convergence of their algorithm. However for static allocation of the samples, they present some properties of the stratified estimate when the number of strata goes to infinity and provide convergence results under the optimal oracle strategy. As a corollary, they prove that the more strata there are, the smaller the optimal oracle variance is.

Contribution: The more strata, the smaller the variance of the estimate computed when following the optimal oracle strategy. However, the more strata there are, the more difficult it is to estimate the variance within each of these strata, and thus the more difficult it is to perform almost as well as the optimal oracle strategy. Choosing the number of strata is thus crucial and this is the problem we address in this paper. This defines a trade-off similar to the one in model selection (such as in e.g. density estimation, regression...): The wider the class of considered models, i.e. the larger the number of strata, the smaller the distance between the true model and the best model of the class, i.e. the approximation error, but the larger the estimation error.

Paper [4], although proposing no finite time bounds, develops very interesting ideas for bounding the first term, i.e. the approximation error. As pointed out in e.g. [1], it is possible to build algorithms that have a small estimation error. By constructing tight and finite-time bounds for the approximation error, it is thus possible to select a number of strata that minimizes an upper bound on the performance. It is however not clear if this choice is really optimal in some sense. The essential ingredients for choosing efficiently a partition are thus lower bounds *on the estimation error, and on the approximation error.*

The objective of this paper is to propose a method for choosing the minimax-optimal number of strata. Our contributions are the following.

- We first present results on what we call the *quality* $Q_{n,\mathcal{N}}$ of a given partition $\mathcal{N}$ in K strata (i.e., using the previous analogy to model selection, this would represent the approximation error). Using very mild assumptions we compute a lower bound on the variance of the estimate given by the optimal oracle strategy on the optimal oracle partition. Then if the function and the standard deviation of the noise are $\alpha-$Hölder, and if the strata also satisfy some conditions, we prove that $Q_{n,\mathcal{N}} = O(\frac{K^{\alpha/d}}{n})$. This bound is also minimax optimal on the class of $\alpha-$Hölder functions.
- We then present results on the estimation error for the estimate output by algorithm MC-UCB of [1] (pseudo-regret in the terminology of [1]). In this paper, we improve the analysis of the MC-UCB algorithm compared to [1] in terms of the dependence on K. The problem independent bound on the pseudo-regret in [1] is of order[2] $\tilde{O}(Kn^{-4/3})$, and we tighten this bound in this paper so that it is of order $\tilde{O}(K^{1/3}n^{-4/3})$.
- We provide the first *lower bound* (on the pseudo-regret) for the problem of online Stratified Sampling. The bound $\Omega(K^{1/3}n^{-4/3})$ is tight and *matches the upper-bound of MC-UCB both in terms of the number of strata and the number of samples* up to a $\sqrt{\log(nK)}$ factor. We believe that the proof technique for this bound is original.
- Finally, we combine the results on the quality and on the pseudo-regret of MC-UCB to provide a value on the number of strata leading to a minimax-optimal trade-off (up to a $\sqrt{\log(n)}$) on the class of $\alpha-$Hölder functions.

The rest of the paper is organized as follows. In Section 1 we formalize the problem and introduce the notations used throughout the paper. Section 2 states the results on the quality of a partition. Section 3 improves the analysis of the MC-UCB algorithm, and establishes the lower bound on the pseudo-regret. Section 4 reports the best trade-off to choose the number of strata. And in Section 5, we illustrate how important it is to carefully choose the number of strata. We finally conclude the paper and suggest future works.

Due to space constraints, we were not able to incorporate complete proofs of our results in this paper, but they are all available in the Technical Report [12].

[2] Here $\tilde{O}$ is a O up to **poly**$(\log(n))$ factor.

1 Setting

We consider the problem of numerical integration of a function $f : [0,1]^d \rightarrow \mathbb{R}$ with respect to the uniform (Lebesgue) measure. We have at our disposal a budget of n queries (samples) to the function, and we can allocate this budget *sequentially*. When querying the function at a time t and at a point x_t, we receive a noisy sample $X(t)$ of the form described in Equation 1. We now assume that the space is stratified in K Lebesgue measurable strata that form a partition $\mathcal{N}$. We index these strata, called Ω_k, with indexes $k \in \{1, \dots, K\}$, and write w_k their measure, according to the Lebesgue measure. We write $\mu_k = \frac{1}{w_k}\int_{\Omega_k} \mathbb{E}_{\epsilon\sim\nu_x}[f(x)+s(x)\epsilon]dx = \frac{1}{w_k}\int_{\Omega_k} f(x)dx$ their mean and $\sigma_k^2 = \frac{1}{w_k}\int_{\Omega_k} \mathbb{E}_{\epsilon\sim\nu_x}[(f(x)+s(x)\epsilon - \mu_k)^2]dx$ their variance. These mean and variance correspond to the mean and variance of the random variable $X(t)$ when the coordinate x at which the noisy evaluation of f is observed is chosen uniformly at random on the stratum Ω_k.

We denote by $\mathcal{A}$ an algorithm that allocates online the budget by selecting at each time step $1 \leq t \leq n$ the index $k_t \in \{1, \dots, K\}$ of a stratum and then samples uniformly in the corresponding stratum Ω_{k_t}. The objective is to return the best possible estimate $\hat{\mu}_n$ of the integral of the function f. We write $T_{k,n} = \sum_{t\leq n} \mathbb{I}\{k_t = k\}$ the number of samples in stratum Ω_k up to time n. We denote by $\big(X_{k,t}\big)_{1\leq k\leq K, 1\leq t\leq T_{k,n}}$ the samples in stratum Ω_k, and we define $\hat{\mu}_{k,n} = \frac{1}{T_{k,n}}\sum_{t=1}^{T_{k,n}} X_{k,t}$ (the empirical means in the stratum). We estimate the integral of f by $\hat{\mu}_n = \sum_{k=1}^{K} w_k \hat{\mu}_{k,n}$.

If we allocate a deterministic number of samples T_k to each stratum Ω_k and if the samples are independent and chosen uniformly on each stratum Ω_k, we have

$$\mathbb{E}(\hat{\mu}_n) = \sum_{k\leq K} w_k\mu_k = \sum_{k\leq K}\int_{\Omega_k} f(u)du = \int_{[0,1]^d} f(u)du = \mu,$$

and also

$$\mathbb{V}(\hat{\mu}_n) = \sum_{k\leq K}\frac{w_k^2\sigma_k^2}{T_k},$$

where the expectation and the variance are computed according to all the samples that the algorithm collected.

For a given algorithm $\mathcal{A}$ allocating $T_{k,n}$ samples drawn *uniformly* within stratum Ω_k, we call *pseudo-risk* the quantity

$$L_{n,\mathcal{N}}(\mathcal{A}) = \sum_{k\leq K}\frac{w_k^2\sigma_k^2}{T_{k,n}}. \tag{2}$$

Note that if an algorithm $\mathcal{A}^*$ has access the variances σ_k^2 of the strata, it can choose to allocate the budget in order to minimize the pseudo-risk, i.e., sample each stratum $T_k^* = \frac{w_k\sigma_k}{\sum_{i\leq K} w_i\sigma_i} n$ times (this is the so-called oracle allocation). These optimal numbers of samples can be non-integer values, in which case the proposed optimal allocation is not realizable. But we still use it as a benchmark.

The pseudo-risk for this algorithm (which is also the variance of the estimate here since the sampling strategy is deterministic) is then

$$L_{n,\mathcal{N}}(\mathcal{A}^*) = \frac{\Big(\sum_{k\leq K} w_k\sigma_k\Big)^2}{n} = \frac{\Sigma_{\mathcal{N}}^2}{n}, \tag{3}$$

where $\Sigma_{\mathcal{N}} = \sum_{k\leq K} w_k\sigma_k$. We also refer in the sequel as optimal proportion to $\lambda_k = \frac{w_k\sigma_k}{\sum_{i\leq K} w_i\sigma_i}$, and to optimal oracle strategy to this allocation strategy. Although, as already mentioned, the optimal allocations (and thus the optimal pseudo-risk) might not be realizable, it is still very useful in providing a lower-bound. No static (even oracle) algorithm has a pseudo-risk lower than $L_{n,\mathcal{N}}(\mathcal{A}^*)$ on partition $\mathcal{N}$.

It is straightforward to see that the more refined the partition $\mathcal{N}$ the smaller $L_{n,\mathcal{N}}(\mathcal{A}^*)$ (see e.g. [7]). We thus define the *quality of a partition* $Q_{n,\mathcal{N}}$ as the difference between the variance $L_{n,\mathcal{N}}(\mathcal{A}^*)$ of the estimate provided by the optimal oracle strategy on partition $\mathcal{N}$, and the infimum of the variance of the optimal oracle strategy on *any* partition (optimal oracle partition) (with an arbitrary number of strata):

$$Q_{n,\mathcal{N}} = L_{n,\mathcal{N}}(\mathcal{A}^*) - \inf_{\mathcal{N}' measurable} L_{n,\mathcal{N}'}(\mathcal{A}^*). \tag{4}$$

We also define the *pseudo-regret* of an algorithm $\mathcal{A}$ on a given partition $\mathcal{N}$, as the difference between its pseudo-risk and the variance of the optimal oracle strategy:

$$R_{n,\mathcal{N}}(\mathcal{A}) = L_{n,\mathcal{N}}(\mathcal{A}) - L_{n,\mathcal{N}}(\mathcal{A}^*). \tag{5}$$

We will assess the performance of an algorithm $\mathcal{A}$ by comparing its pseudo risk to the minimum possible variance of an optimal oracle strategy on the optimal oracle partition:

$$L_{n,\mathcal{N}}(\mathcal{A}) - \inf_{\mathcal{N}' measurable} L_{n,\mathcal{N}'}(\mathcal{A}^*) = R_{n,\mathcal{N}}(\mathcal{A}) + Q_{n,\mathcal{N}}. \tag{6}$$

Using the analogy of model selection mentioned in the Introduction, the quality $Q_{n,\mathcal{N}}$ is similar to the approximation error and the pseudo-risk $R_{n,\mathcal{N}}(\mathcal{A})$ to the estimation error.

Motivation for the model $f(x)+s(x)\epsilon_t$. Assume that a learner can, at each time t, choose a point x and collect an observation $F(x, W_t)$, where W_t is an independent noise, that can however depend on x. It is the general model for representing evaluations of a noisy function. There are many settings where one needs to integrate accurately a noisy function without wasting too much budget, like for instance pollution survey. Set $f(x) = \mathbb{E}_{W_t}[F(x, W_t)]$, and $s(x)\epsilon_t = F(x, W_t) - f(x)$. Since by definition ϵ_t is of mean 0 and variance 1, we have in fact $s(x) = \sqrt{\mathbb{E}_{W_t}[(F(x, W_t) - f(x))^2]}$ and $\epsilon_t = \frac{F(x,W_t)-f(x)}{s(x)}$. Observing $F(x, W_t)$ is thus equivalent to observing $f(x) + s(x)\epsilon_t$, and this implies that the model that we choose is also very general.

There is also an important setting where this model is relevant, and this is for the integration of a function F in high dimension d^*. Stratifying in dimension d^* seems hopeless, since the budget n has to be exponential with d^* if one wants to stratify in every direction of the domain: this is the curse of dimensionality. It is necessary to reduce the dimension by choosing *a small number* of directions $(1, \dots, d)$ that are particularly relevant, and control/stratify only in these d directions[3]. Then the control/stratification is only on those d coordinates, so when sampling at a time t, one chooses $x = (x_1, \dots, x_d)$, and the other $d^* - d$ coordinates $U(t) = (U_{d+1}(t), \dots, U_{d^*}(t))$ are uniform random variables on $[0,1]^{d^*-d}$ (without any control). When sampling in x at a time t, we observe $F(x, U(t))$. By writing $f(x) = \mathbb{E}_{U(t)\sim\mathcal{U}([0,1]^{d^*-d})}[F(x, U(t))]$, and $s(x)\epsilon_t = F(x, U(t)) - f(x)$, we obtain that the model we propose is also valid in this case.

2 The Quality of a Partition: Analysis of the Term $Q_{n,\mathcal{N}}$

In this Section, we focus on the *quality* of a partition defined in Section 1.

Convergence under very mild assumptions. As mentioned in Section 1, the more refined the partition $\mathcal{N}$ of the space, the smaller $L_{n,\mathcal{N}}(\mathcal{A}^*)$, and thus $\Sigma_{\mathcal{N}}$. Through this monotony property, we know that $\inf_{\mathcal{N}} \Sigma_{\mathcal{N}}$ is also the limit of the $(\Sigma_{\mathcal{N}_p})_p$ of a sequence of partitions $(\mathcal{N}_p)_p$ such that the diameter of each stratum goes to 0. We state in the following Proposition that for *any* such sequence, $\lim_{p\to+\infty} \Sigma_{\mathcal{N}_p} = \int_{[0,1]^d} s(x)dx$. Consequently $\inf_{\mathcal{N}} \Sigma_{\mathcal{N}} = \int_{[0,1]^d} s(x)dx$.

Proposition 1. *Let $(\mathcal{N}_p)_p = (\Omega_{k,p})_{k\in\{1,\dots,K_p\}, p\in\{1,\dots,+\infty\}}$ be a sequence of measurable partitions (where K_p is the number of strata of partition $\mathcal{N}_p$) such that*

- AS1: *$0 < w_{k,p} \le v_p$, for some sequence $(v_p)_p$, where $v_p \to 0$ for $p \to +\infty$.*
- AS2: *The diameters according to the $||.||_2$ norm on $\mathbb{R}^d$ of the strata are such that $\mathbf{Diam}(\Omega_{k,p}) \le D(w_{k,p})$, for some real valued function $D(\cdot)$, such that $D(w) \to 0$ for $w \to 0$.*

If the functions m and s are in $\mathbb{L}_2([0,1]^d)$, then

$$\lim_{p\to+\infty} \Sigma_{\mathcal{N}_p} = \inf_{\mathcal{N} measurable} \Sigma_{\mathcal{N}} = \int_{[0,1]^d} s(x)dx,$$

which implies that $n \times Q_{n,\mathcal{N}_p} \to 0$ for $p \to +\infty$.

The full proof of this Proposition (omitted due to space constraints) is available in the Technical Report [12].

In Proposition 1, even though the optimal oracle allocation might not be realizable (in particular if the number of strata is larger than the budget), we can still compute the quality of a partition, as defined in Equation 4. It does not correspond to any reachable pseudo-risk, but rather to a lower bound on any (even oracle) static allocation.

[3] This is actually a very common technique for computing the price of options, see [6].

When f and s are in $\mathbb{L}_2([0,1]^d)$, for any appropriate sequence of partitions $(\mathcal{N}_p)_p$, $\Sigma_{\mathcal{N}_p}$ (which is the principal ingredient of the variance of the optimal oracle allocation) converges to the smallest possible $\Sigma_{\mathcal{N}}$ for given f and s. Note however that this condition is not sufficient to obtain a *rate* of convergence.

Finite-Time analysis under Hölder assumption: We make the following assumption on the functions f and s.

Assumption 1. *The functions f and s are $(M,\alpha)-$Hölder continuous, i.e., for $g \in \{f,s\}$, for any x and $y \in [0,1]^d, |g(x) - g(y)| \leq M||x-y||_2^{\alpha}$.*

The Hölder assumption enables us to consider arbitrarily non-smooth functions (for small α, the function can vary arbitrarily fast), and is thus a fairly general assumption.

We also consider the following partitions in K hyper-cubes.

Definition 1. *We write $\mathcal{N}_K$ the partition of $[0,1]^d$ in K hyper-cubic strata of measure $w_k = w = \frac{1}{K}$ and side length $(\frac{1}{K})^{1/d}$: we assume for simplicity that there exists an integer l such that $K = l^d$.*

The following Proposition holds.

Proposition 2. *Under Assumption 1 we have for any partition $\mathcal{N}_K$ as defined in Definition 1 that*

$$\Sigma_{\mathcal{N}_K} - \int_{[0,1]^d} s(x)dx \leq \sqrt{2d}MK^{-\alpha/d}, \tag{7}$$

which implies

$$Q_{n,\mathcal{N}_K} \leq \frac{2\sqrt{2d}M\Sigma_{\mathcal{N}_1}}{n}K^{-\alpha/d},$$

where $\mathcal{N}_1$ stands for the "partition" with one stratum.

The full proof of this Proposition (omitted due to space constraints) is available in the Technical Report [12].

2.1 General Comments

The impact of α and d: The quantity $Q_{n,\mathcal{N}_K}$ increases with the dimension d, because the Hölder assumption becomes less constraining when d increases. This can easily be seen since a squared strata of measure w has a diameter of order $w^{1/d}$. $Q_{n,\mathcal{N}_K}$ decreases with the smoothness α of the function, which is a consequence of the Hölder assumption. Note also that when defining the partitions $\mathcal{N}_K$ in Definition 1, we made the crucial assumption that $K^{1/d}$ is an integer. This is of little importance in small dimension, but matters in high dimensions, as we will highlight in the last remark of Section 4.

Minimax optimality of this rate: The rate $n^{-1}K^{-\alpha/d}$ is minimax optimal on the class of $\alpha-$Hölder functions since for any n and K one can easily build a function with Hölder exponent α such that the corresponding $\Sigma_{\mathcal{N}_K}$ is at least $\int_{[0,1]^d} s(x)dx + cK^{-\alpha/d}$ for some constant c.

Discussion of the shape of the strata: Whatever the shape of the strata, as long as their diameter goes to zero[4], $\Sigma_{\mathcal{N}_K}$ converges to $\int_{[0,1]^d} s(x)dx$. The shape of the strata have an influence only on the negligible term, i.e. the speed of convergence to this quantity. This result was already made explicit, in a different setting and under different assumptions, in [4]. Choosing small strata of same shape and size is also minimax optimal on the class of Hölder functions. Working on the shape of the strata could, however, improve the speed of convergence in some specific cases, e.g. when the noise is very localized. It could also be interesting to consider strata of varying size, and have this size depend on the specific problem.

The decomposition of the variance: The variance σ_k^2 within each stratum Ω_k comes from two sources. First, σ_k^2 comes from the noise, that contributes to it by $\frac{1}{w_k}\int_{\Omega_k} s(x)^2 dx$. Second, the mean f is not a constant function, thus its contribution to σ_k^2 is $\frac{1}{w_k}\int_{\Omega_k}\big(f(x) - \frac{1}{w_k}\int_{\Omega_k} f(u)du\big)^2 dx$. Note that when the size of Ω_k goes to 0, this later contribution vanishes, and the optimal allocation is thus proportional to $\sqrt{w_k \int_{\Omega_k} s(x)^2 dx + o(1)} = \int_{\Omega_k} s(x)dx + o(1)$. This means that for small strata, the variation in the mean are negligible when compared to the variation due to the noise.

3 Algorithm MC-UCB and a Matching Lower Bound

3.1 Algorithm $MC-UCB$

In this Subsection, we describe a slight modification of the algorithm $MC-UCB$ introduced in [1]. The only difference is that we change the form of the high-probability upper confidence bound on the standard deviations, in order to improve the elegance of the proofs, and we refine their analysis. The algorithm takes as input two parameters b and $f_{\max}$ which are linked to the distribution in the strata, δ which is a (small) probability, and the partition $\mathcal{N}_K$.

We remind in Figure 1 the algorithm $MC-UCB$.

The estimates of $\hat{\sigma}^2_{k,t-1}$ and $\hat{\mu}_{k,t-1}$ are computed according to

$$\hat{\sigma}^2_{k,t-1} = \frac{1}{T_{k,t-1}}\sum_{i=1}^{T_{k,t-1}}(X_{k,i} - \hat{\mu}_{k,t-1})^2, \;\; \textit{and} \;\; \hat{\mu}_{k,t-1} = \frac{1}{T_{k,t-1}}\sum_{i=1}^{T_{k,t-1}} X_{k,i}\;. \qquad (8)$$

[4] And note that in this *noisy* setting, if the diameter of the strata does not go to 0 on non homogeneous part of m and s, then the standard deviation corresponding to the allocation is larger than $\int_{[0,1]^d} s(u)du$.

```
**Input:** $b$, $f_{\max}$, $\delta$, $\mathcal{N}_K$. Set $A = 2\sqrt{(1+3b+4f_{\max}^2)\log(2nK/\delta)}$
**Initialize:** Sample 2 states in each strata.
**for** $t = 2K+1, \dots, n$ **do**
    Compute $B_{k,t} = \frac{w_k}{T_{k,t-1}}\Big(\hat{\sigma}_{k,t-1} + A\sqrt{\frac{1}{T_{k,t-1}}}\Big)$ for each stratum $k \le K$
    Sample a point in stratum $k_t \in \arg\max_{1\le k\le K} B_{k,t}$
**end for**
**Output:** $\hat{\mu}_n = \sum_{k=1}^K w_k \hat{\mu}_{k,n}$
```

Fig. 1. The pseudo-code of the MC-UCB algorithm. The empirical standard deviations and means $\hat{\sigma}_{k,t}^2$ and $\hat{\mu}_{k,t}$ are computed using Equation 8.

3.2 Upper Bound on the Pseudo-regret of Algorithm MC-UCB

We first state the following Assumption on the noise ϵ_t:

Assumption 2. *There exist $b > 0$ such that $\forall x \in [0,1]^d$, $\forall t$, and $\forall \lambda < \frac{1}{b}$,*

$$\mathbb{E}_{\nu_x}\Big[\exp(\lambda\epsilon_t)\Big] \le \exp\Big(\frac{\lambda^2}{2(1-\lambda b)}\Big), \textit{ and } \mathbb{E}_{\nu_x}\Big[\exp(\lambda\epsilon_t^2 - \lambda)\Big] \le \exp\Big(\frac{\lambda^2}{2(1-\lambda b)}\Big).$$

This is a type of sub-Gaussian assumption, satisfied for e.g., Gaussian as well as bounded distributions. We also state an assumption on f and s.

Assumption 3. *The functions f and s are bounded by $f_{\max}$.*

Note that since the functions f and s are defined on $[0,1]^d$, if Assumption 1 is satisfied, then Assumption 3 holds with $f_{\max} = \max(f(0), s(0)) + \sqrt{2d}M$. We now prove the following bound on the pseudo-regret. Note that we state it on partitions $\mathcal{N}_K$, but that it in fact holds for any partition in K strata.

Proposition 3. *Under Assumptions 2 and 3, on partition $\mathcal{N}_K$, when $n \ge 4K$, we have*

$$\mathbb{E}[R_{n,\mathcal{N}_K}(\mathcal{A}_{MC-UCB})] \le C\frac{K^{1/3}}{n^{4/3}}\sqrt{\log(nK)} + \frac{14K\Sigma_{\mathcal{N}_K}^2}{n^2},$$

where $C = 24\sqrt{2}\Sigma_{\mathcal{N}_K}\sqrt{(1+3b+4f_{\max}^2)}\Big(\frac{f_{\max}+4}{4}\Big)^{1/3}$.

The proof of this Proposition is close to the one of MC-UCB in [1]. But an improved analysis leads to a better dependency in terms of number of strata K. Recall that in [1], the bound is of order $\tilde{O}(Kn^{-4/3})$. This improvement is crucial here since the larger K is, the closer $\Sigma_{\mathcal{N}_K}$ is to $\int_{[0,1]^d} s(x)dx$. The full proof of this Proposition is available in the Technical Report [12]. The next Subsection states that the rate $K^{1/3}\tilde{O}(n^{-4/3})$ of MC-UCB is optimal both in terms of K and n.

3.3 Lower Bound

We now study the minimax rate for the pseudo-regret of any algorithm on a given partition $\mathcal{N}_K$.

Theorem 1. *Let* $K \in \mathbb{N}$*. Let* inf *be the infimum taken over all online stratified sampling algorithms on* $\mathcal{N}_K$ *and* sup *represent the supremum taken over all environments, then:*

$$\inf \sup \mathbb{E}[R_{n,\mathcal{N}_K}] \geq C \frac{K^{1/3}}{n^{4/3}},$$

where C *is a numerical constant.*

Proof (Proof sketch (the full proof of this Theorem is available in the Technical Report [12])). We consider a partition with $2K$ strata. On the K first strata, the samples are drawn from Bernoulli distributions of parameter μ_k where $\mu_k \in \{\frac{\mu}{2}, \mu, 3\frac{\mu}{2}\}$, and on the K last strata, the samples are drawn from a Bernoulli of parameter $1/2$. We write $\sigma = \sqrt{\mu(1-\mu)}$ the standard deviation of a Bernoulli of parameter μ. We index by υ a set of 2^K possible environments, where $\upsilon = (\upsilon_1, \ldots, \upsilon_K) \in \{-1, +1\}^K$, and the K first strata are defined by $\mu_k = \mu + \upsilon_k \frac{\mu}{2}$. Write $\mathbb{P}_\upsilon$ the probability under such an environment, also consider $\mathbb{P}_\sigma$ the probability under which all the K first strata are Bernoulli with mean μ.

We define Ω_υ the event on which there are less than $\frac{K}{3}$ strata not pulled correctly for environment υ (i.e. for which $T_{k,n}$ is larger than the optimal allocation corresponding to μ when actually $\mu_k = \frac{\mu}{2}$, or smaller than the optimal allocation corresponding to μ when $\mu_k = 3\frac{\mu}{2}$). See the Appendix D in [12] for a precise definition of these events. Then, the idea is that there are so many such environments that any algorithm will be such that for at least one of them we have $\mathbb{P}_\sigma(\Omega_\upsilon) \leq \exp(-K/72)$. Then we derive by a variant of Pinsker's inequality applied to an event of small probability that $\mathbb{P}_\upsilon(\Omega_\upsilon) \leq \frac{KL(\mathbb{P}_\sigma, \mathbb{P}_\upsilon)}{K} = O(\frac{\sigma^{3/2} n}{K})$. Finally, by choosing σ of order $(\frac{K}{n})^{1/3}$, we have that $\mathbb{P}_\upsilon(\Omega_\upsilon^c)$ is bigger than a constant, and on Ω_υ^c we know that there are more than $\frac{K}{3}$ strata not pulled correctly. This leads to an expected pseudo-regret in environment υ of order $\Omega(\frac{K^{1/3}}{n^{4/3}})$.

This is the first lower-bound for the problem of online stratified sampling for Monte-Carlo. Note that this bound is of same order as the upper bound for the pseudo-regret of algorithm MC-UCB. It means that this algorithm is, up to a factor $\sqrt{\log(nK)}$, minimax optimal, both in terms of the number of samples and in terms of the number of strata. It however holds only on the partitions $\mathcal{N}_K$ (we conjecture that a similar result holds for *any* measurable partition $\mathcal{N}$, but with a bound of order $\Omega\Big(\sum_{x \in \mathcal{N}} \frac{w_x^{2/3}}{n^{4/3}}\Big)$).

4 Minimax-Optimal Trade-Off between $Q_{n,\mathcal{N}_K}$ and $R_{n,\mathcal{N}_K}(\mathcal{A}_{MC-UCB})$

4.1 Minimax-Optimal Trade-Off

We consider in this Section the hyper-cubic partitions $\mathcal{N}_K$ as defined in Definition 1, and we want to find the minimax-optimal number of strata K_n as a function of n. Using the results in Section 2 and Subsection 3.1, it is possible to deduce an optimal number of strata K to give as parameter to the algorithm MC-UCB. Note that since the performance of the algorithm is defined as the sum of the quality of partition $\mathcal{N}_K$, i.e. $Q_{n,\mathcal{N}_K}$ and of the pseudo-regret of the algorithm MC-UCB, namely $R_{n,\mathcal{N}_K}(\mathcal{A}_{MC-UCB})$, one wants to (i) on the one hand take many strata so that $Q_{n,\mathcal{N}_K}$ is small but (ii) on the other hand, pay attention to the impact this number of strata has on the pseudo-regret $R_{n,\mathcal{N}_K}(\mathcal{A}_{MC-UCB})$. A good way to do that is to choose K_n in function of n such that $Q_{n,\mathcal{N}_{K_n}}$ and $R_{n,\mathcal{N}_{K_n}}(\mathcal{A}_{MC-UCB})$ are of the same order.

Theorem 2. *Under Assumptions 1 and 2 (since Assumption 1 implies Assumption 3, by setting $f_{\max} = X(1)+\sqrt{2d}M$), with $K_n = \Big(\lfloor(n^{\frac{d}{d+3\alpha}})^{1/d}\rfloor\Big)^d (\leq n^{\frac{d}{d+3\alpha}} \leq n)$, we have*

$$\mathbb{E}[L_n(\mathcal{A}_{MC-UCB})] - \frac{1}{n}\Big(\int_{[0,1]^d} s(x)dx\Big)^2 \leq cd^{\frac{2\alpha}{3d}+\frac{1}{2}}\sqrt{\log(n)}n^{-\frac{d+4\alpha}{d+3\alpha}}(1 + d^{\alpha}n^{-\frac{\alpha}{d+3\alpha}}),$$

where $c = 70(1+M)\Sigma_{\mathcal{N}_K}\sqrt{(1+3b+4(f(0)+s(0)+M)^2)}\Big(\frac{(f(0)+s(0)+M)+4}{4}\Big)^{1/3}$.

If $d \ll n$, then $\mathbb{E}[L_n(\mathcal{A}_{MC-UCB})] - \frac{1}{n}\Big(\int_{[0,1]^d} s(x)dx\Big)^2 = \tilde{O}(n^{-\frac{d+4\alpha}{d+3\alpha}})$.

We can also prove a matching (up to $\sqrt{\log(n)}$) minimax lower bound using the results in Theorem 1.

Theorem 3. *Let* sup *represent the supremum taken over all $\alpha-$Hölder functions and* inf *be the infimum taken over all algorithms that partition the space in convex strata of same shape, then the following holds true:*

$$\inf\sup \mathbb{E}L_n(\mathcal{A}) - \frac{1}{n}\Big(\int_{[0,1]^d} s(x)dx\Big)^2 = \Omega(n^{-\frac{d+4\alpha}{d+3\alpha}}).$$

4.2 Discussion

Optimal pseudo-risk. The dominant term in the pseudo-risk of MC-UCB with the proper number of strata is $\frac{(\inf_{\mathcal{N}}\Sigma_{\mathcal{N}})^2}{n} = \frac{1}{n}\big(\int_{[0,1]^d} s(x)dx\big)^2$ (the other term is negligible). This means that algorithm MC-UCB is almost as efficient as the optimal oracle strategy on the optimal oracle partition. In comparison, the variance of the estimate given by crude Monte-Carlo is $\int_{[0,1]^d}\big(f(x) - \int_{[0,1]^d} f(u)du\big)^2 dx + \int_{[0,1]^d} s(x)^2dx$. Thus MC-UCB enables to have the term coming from the variations in the mean vanish, and the noise term decreases (since by Cauchy-Schwarz, $\big(\int_{[0,1]^d} s(x)dx\big)^2 \leq \int_{[0,1]^d} s(x)^2dx$).

Minimax-optimal trade-off for algorithm MC-UCB. The optimal trade-off on the number of strata K_n of order $n^{\frac{d}{d+3\alpha}}$ depends on the dimension and the smoothness of the function. The higher the dimension, the more strata are needed in order to have a decent speed of convergence for $\Sigma_{\mathcal{N}_K}$. The smoother the function, the fewer strata are needed.

It is yet important to remark that this trade-off is not exact. We provide an almost minimax-optimal order of magnitude for K_n, in terms of n, so that the rate of convergence of the algorithm is minimax-optimal up to a $\sqrt{\log(n)}$ factor.

Link between risk and pseudo-risk. It is important to compare the pseudo-risk $L_n(\mathcal{A}) = \sum_{k=1}^{K} \frac{w_k^2\sigma_k^2}{T_{k,n}}$ and the true risk $\mathbb{E}[(\hat{\mu}_n - \mu)^2]$. Note that these quantities are in general not equal for an algorithm $\mathcal{A}$ that allocates the samples in a dynamic way: indeed, the quantities $T_{k,n}$ are in that case stopping times and the variance of estimate $\hat{\mu}_n$ is not equal to the pseudo-risk. However, in the paper [2], the authors highlighted for $MC-UCB$ some links between the risk and the pseudo-risk. More precisely, they established links between $L_n(\mathcal{A})$ and $\sum_{k=1}^{K} w_k^2\mathbb{E}[(\hat{\mu}_{k,n}-\mu_k)^2]$. This step is possible since $\mathbb{E}[(\hat{\mu}_{k,n}-\mu_k)^2] \leq \frac{w_k^2\sigma_k^2}{\underline{T}_{k,n}^2}\mathbb{E}[T_{k,n}]$, where $\underline{T}_{k,n}$ is a lower-bound on the number of pulls $T_{k,n}$ on a high probability event. Then they bounded the cross products $\mathbb{E}[(\hat{\mu}_{k,n} - \mu_k)(\hat{\mu}_{p,n} - \mu_p)]$ and provided some upper bounds on these terms. A tight analysis of these terms as a function of the number of strata K remains to be investigated.

Knowledge of the Hölder exponent. In order to be able to choose properly the number of strata to achieve the rate in Theorem 2, it is needed to possess a proper lower bound on the Hölder exponent of the function: indeed, the rougher the function is, the more strata are required. On the other hand, such a knowledge on the function is not always available and an interesting question is whether it is possible to estimate this exponent fast enough. There are interesting papers on that subject like [9] where the authors tackle the problem of regression and prove that it is possible to adapt to the unknown smoothness of the function. The authors in [5] add to that (in the case of density estimation) and prove that it is even possible under the assumption that the function attain its Hölder exponent to have a proper estimation of this exponent and thus adaptive confidence bands. An idea would be to try to adapt these results in the case of finite sample.

MC-UCB On a noiseless function. Consider the case where $s = 0$ almost surely, i.e. the collected samples are noiseless. Proposition 1 ensures that $\inf_{\mathcal{N}} \Sigma_{\mathcal{N}} = 0$: it is thus possible in this case to achieve a pseudo-risk that has a faster rate than $O(\frac{1}{n})$. If the function m is smooth, e.g. Hölder with an exponent α which is not too small, it is efficient to use low discrepancy methods to integrate the functions. An idea is to stratify the domain in n hyper-rectangular strata of minimal diameter, and to pick at random one sample per stratum. The variance of the resulting estimate is of order $O(\frac{1}{n^{1+2\alpha/d}})$. Algorithm MC-UCB is not as efficient as a low discrepancy scheme: it needs a number of strata $K < n$ in order to be able to estimate the variance within each stratum. Its pseudo-risk is then of order $O(\frac{1}{nK^{2\alpha/d}})$.

However, this only holds when the samples are noiseless. Otherwise, the variance of the estimate is of order $1/n$, no matter what strategy the learner chooses.

In high dimension. The first bound in Theorem 2 expresses precisely how the performance of the estimate output by MC-UCB depends on d. The first bound states that the quantity $L_n(\mathcal{A}) - \frac{1}{n}\Big(\int_{[0,1]^d} s(x)dx\Big)^2$ is negligible when compared to $1/n$ when n is exponential in d. This is not surprising since our technique aims at stratifying equally in every direction. It is not possible to stratify in every directions of the domain if the function lies in a very high dimensional domain.

This is however *not* a reason for not using our algorithm in high dimension. Indeed, stratifying even in a small number of strata already reduces the variance, and in high dimension, any variance reduction techniques are welcome. As mentioned at the end of Section 1, the model that we propose for the function is suitable for modeling d^* dimensional functions that we only stratify in $d < d^*$ directions (and $d \ll n$). A reasonable trade-off for d can also be inferred from the bound, but we believe that a good choice of d depends heavily on the problem. We then believe that it is a good idea to select the number of strata in the minimax way according to our results. Again, having a very high dimensional function that one stratifies in only a few directions is a very common technique in financial mathematics, for pricing options (practitioners stratify an infinite dimensional process in only 1 to 5 carefully chosen dimensions). We illustrate this in the next Section.

5 Numerical Experiment: Influence of the Number of Strata in the Pricing of an Asian Option

We consider the pricing problem of an Asian option introduced in [7] and later considered in [10, 3]. This uses a Black-Scholes model with strike C and maturity T. Let $(W(t))_{0\leq t\leq T}$ be a Brownian motion. The discounted payoff of the Asian option is defined as a function of W, by:

$$F((W)_{0\leq t\leq T}) = \exp(-rT)\max\Big[\int_0^T S_0 \exp\Big((r-\tfrac{1}{2}s_0^2)t + s_0 W_t\Big)dt - C, 0\Big],$$

where S_0, r, and s_0 are constants.

We want to estimate the price $p = \mathbb{E}_W[F(W)]$ by Monte-Carlo simulations (by sampling on W). In order to reduce the variance of the estimated price, we can stratify the space of W. [7] suggest to stratify according to a one dimensional projection of W, i.e., by choosing a time t and stratifying according to the quantiles of W_t (and simulating the rest of the Brownian according to a Brownian Bridge, see [10]). They further argue that the best direction for stratification is to choose $t = T$, i.e., to stratify according to the last time of T. This choice of stratification is also intuitive since W_T has the highest variance, the largest exponent in the payoff and thus the highest volatility. We stratify according to the quantiles of W_T, that is to say the quantiles of a normal distribution $\mathcal{N}(0,T)$.

When stratifying in K strata, we stratify according to the $1/K$-th quantiles (so that the strata are hyper-cubes of same measure).

We choose the same numerical values as [10]: $S_0 = 100$, $r = 0.05$, $s_0 = 0.30$, $T = 1$ and $d = 16$. We discretize also, as in [10], the Brownian motion in 16 equidistant times, so that we are able to simulate it. We choose $C = 120$.

In this paper, we only do experiments for MC-UCB, and exhibit the influence of the number of strata. For a comparison between MC-UCB and other algorithms, see [1]. By studying the range of the $F(W)$, we set the parameter of the algorithm MC-UCB to $A = 150 \log(n)$.

For $n = 200$ and $n = 2000$, we observe the influence of the number of strata in Figure 2 (the number of strata varying from 2 to 100). We plot results for MC-UCB, uniform stratified Monte-Carlo (that allocates a number of samples in each stratum proportional to the measure of the stratum), and also for crude, unstratified, Monte-Carlo. We observe the trade-off that we mentioned between pseudo-regret and quality, in the sense that the mean squared error of the estimate output by MC-UCB (when compared to the true integral of f) first decreases with K and then increases. Note that, without surprise, for a large n the minimum of mean squared error is reached with more strata. Finally, note that our technique is never outperformed by uniform stratified Monte-Carlo.

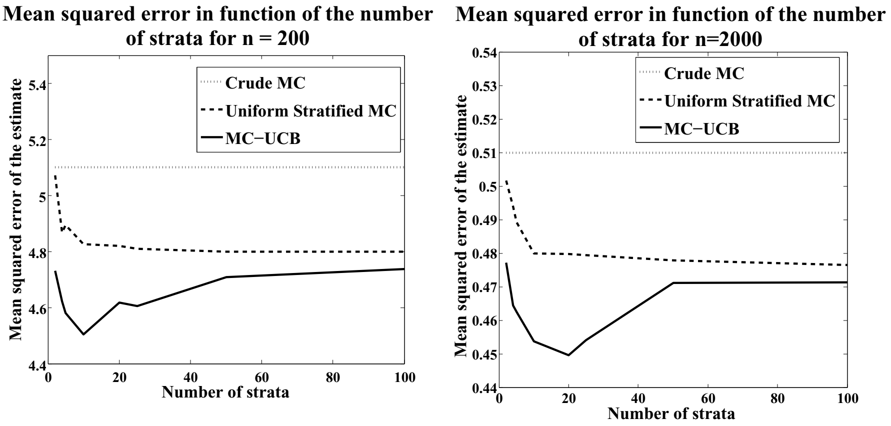


Fig. 2. Mean squared error for crude Monte-Carlo, uniform stratified sampling and MC-UCB, for different numbers of strata, for (Left:) n=200 and (Right:) n=2000

6 Conclusion

In this paper we studied the problem of online stratified sampling for the numerical integration of a function given noisy evaluations, and more precisely we discussed the problem of choosing the *minimax-optimal number* of strata.

We explained why, to our minds, this is a crucial problem when one wants to design an efficient algorithm. We highlighted the trade-off between having many strata (and a good approximation error, i.e. quality of a partition), and not too many, in order to perform almost as well as the optimal oracle allocation on a given partition (small estimation error, i.e. pseudo-regret). When the function is noisy, the noise is the dominant quantity in the optimal oracle variance on the optimal oracle partition. Indeed, decreasing the size of the strata does not diminish the (local) variance of the noise. In this case, the pseudo-risk of algorithm MC-UCB is equal, up to negligible terms, to the mean squared error of the estimate output by the optimal oracle strategy on the best (oracle) partition, at a rate of $O(n^{-\frac{d+4\alpha}{d+3\alpha}})$ where α is the Hölder exponent of s and m. This rate is minimax optimal on the class of α-Hölder functions: it is not possible, to do better on simultaneously all α-Hölder functions.

There are (at least) three very interesting remaining open questions:

- The first one is to investigate whether it is possible to estimate online the Hölder exponent *fast enough*. Indeed, one needs it in order to compute the proper number of strata for MC-UCB, and the lower bound on the Hölder exponent appears in the bound. It is thus a crucial parameter.
- The second direction is to build a more efficient algorithm in the noiseless case. We remarked that MC-UCB is not as efficient in this case as a simple non-adaptive method. The problem comes from the fact that in the case of a noiseless function, it is important to sample the space in a way that ensures that the points are as spread as possible.
- Another question is the relevance of fixing the strata in advance. Although it is minimax-optimal on the class of $\alpha-$Hölder functions to have hyper-cubic strata of same measure, it might in some cases be more interesting to focus and stratify more finely at places where the function is rough.

Acknowledgements. This research was partially supported by Region Nord-Pas-de-Calais Regional Council, French ANR EXPLO-RA (ANR-08-COSI-004), the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement 270327 (project CompLACS), and by Pascal-2.

References

[1] Carpentier, A., Munos, R.: Finite-time analysis of stratified sampling for monte carlo. In: Neural Information Processing Systems, NIPS (2011a)

[2] Carpentier, A., Munos, R.: Finite-time analysis of stratified sampling for monte carlo. Technical report, INRIA-00636924 (2011b)

[3] Etoré, P., Jourdain, B.: Adaptive optimal allocation in stratified sampling methods. Methodol. Comput. Appl. Probab. 12(3), 335–360 (2010)

[4] Etoré, P., Fort, G., Jourdain, B., Moulines, É.: On adaptive stratification. Ann. Oper. Res. (2011) (to appear)

[5] Giné, E., Nickl, R.: Confidence bands in density estimation. The Annals of Statistics 38(2), 1122–1170 (2010)

[6] Glasserman, P.: Monte Carlo methods in financial engineering. Springer (2004) ISBN 0387004513

[7] Glasserman, P., Heidelberger, P., Shahabuddin, P.: Asymptotically optimal importance sampling and stratification for pricing path-dependent options. Mathematical Finance 9(2), 117–152 (1999)

[8] Grover, V.: Active learning and its application to heteroscedastic problems. Department of Computing Science, Univ. of Alberta, MSc thesis (2009)

[9] Hoffmann, M., Lepski, O.: Random rates in anisotropic regression. Annals of Statistics, 325–358 (2002)

[10] Kawai, R.: Asymptotically optimal allocation of stratified sampling with adaptive variance reduction by strata. ACM Transactions on Modeling and Computer Simulation (TOMACS) 20(2), 1–17 (2010) ISSN 1049-3301

[11] Rubinstein, R.Y., Kroese, D.P.: Simulation and the Monte Carlo method. Wiley-interscience (2008) ISBN 0470177942

[12] Carpentier, A., Munos, R.: Minimax Number of Strata for Online Stratified Sampling given Noisy Samples. Technical report, INRIA-00698517 (2012)

Weighted Last-Step Min-Max Algorithm with Improved Sub-logarithmic Regret

Edward Moroshko and Koby Crammer

Department of Electrical Engineering,
The Technion, Haifa, Israel
edward@tx.technion.ac.il, koby@ee.technion.ac.il

Abstract. In online learning the performance of an algorithm is typically compared to the performance of a fixed function from some class, with a quantity called regret. Forster [4] proposed a last-step min-max algorithm which was simpler than the algorithm of Vovk [12], yet with the same regret. In fact the algorithm he analyzed assumed that the choices of the adversary are bounded, yielding artificially only the two extreme cases. We fix this problem by weighing the examples in such a way that the min-max problem will be well defined, and provide analysis with logarithmic regret that may have better multiplicative factor than both bounds of Forster [4] and Vovk [12]. We also derive a new bound that may be sub-logarithmic, as a recent bound of Orabona et.al [9], but may have better multiplicative factor. Finally, we analyze the algorithm in a weak-type of non-stationary setting, and show a bound that is sublinear if the non-stationarity is sub-linear as well.

1 Introduction

We consider the online learning regression problem, in which a learning algorithm tries to predict real numbers in a sequence of rounds given some side-information or inputs $\mathbf{x}_t$. Real-world example applications for these algorithms are weather or stockmarket predictions. The goal of the algorithm is to have a small discrepancy between its predictions and the associated outcomes y_t. This discrepancy is measured with a loss function, such as the square loss. It is common to evaluate algorithms by their regret, the difference between the cumulative loss of an algorithm with the cumulative loss of any function taken from some class.

Forster [4] proposed a last-step min-max algorithm for online regression that makes a prediction assuming it is the last example to be observed, and the goal of the algorithm is indeed to minimize the regret with respect to linear functions. The resulting optimization problem he obtained was convex in both the algorithm's choice and the adversary's choice, which yielded an unbounded problem. Forster circumvented this problem by assuming a bound Y over the choices of the adversary that should be known to the algorithm, yet his analysis is for the version with no bound.

We propose a modified last-step min-max algorithm with weights over examples, that are controled in a way to obtain a concave-convex problem over the adversary's choices and the algorithm's choices. We analyze our algorithm and show a logarithmic-regret that may have a better multiplicative factor than the analysis of Forster. We derive

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 245–259, 2012.

additional analysis that is logarithmic in the loss of the reference function, rather than the number of rounds T. This behaviour was recently given by Orabona et.al [9] for a certain online-gradient decent algorithm. Yet, their bound [9] has a similar multiplicative factor to that of Forster [4], while our bound has a potentially better multiplicative factor and it has the same dependency in the cumulative loss of the reference function as Orabona et.al [9]. Additionally, our algorithm and analysis are totally free of assuming the bound Y or knowing its value.

Competing with the best *single* function might not suffice for some problems. In many real-world applications, the true target function is not fixed, but may change from time to time. We bound the performance of our algorithm also in non-stationary environment, where we measure the complexity of the non-stationary environment by the total deviation of a collection of linear functions from some fixed reference point. We show that our algorithm maintains an average loss close to that of the best sequence of functions, as long as the total of this deviation is sublinear in the number of rounds T.

2 Problem Setting

We work in the online setting for regression evaluated with the squared loss. Online algorithms work in rounds or iterations. On each iteration an online algorithm receives an instance $\mathbf{x}_t \in \mathbb{R}^d$ and predicts a real value $\hat{y}_t \in \mathbb{R}$, it then receives a label $y_t \in \mathbb{R}$, possibly chosen by an adversary, suffers loss $\ell_t(\text{alg}) = \ell\left(y_t, \hat{y}_t\right) = \left(\hat{y}_t - y_t\right)^2$, updates its prediction rule, and proceeds to the next round. The cumulative loss suffered by the algorithm over T iterations is, $L_T(\text{alg}) = \sum_{t=1}^{T} \ell_t(\text{alg})$. The goal of the algorithm is to perform well compared to any predictor from some function class.

A common choice is to compare the performance of an algorithm with respect to *a single* function, or specifically a single linear function, $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{u}$, parameterized by a vector $\mathbf{u} \in \mathbb{R}^d$. Denote by $\ell_t(\mathbf{u}) = \left(\mathbf{x}_t^\top \mathbf{u} - y_t\right)^2$ the instantaneous loss of a vector $\mathbf{u}$, and by $L_T(\mathbf{u}) = \sum_t^T \ell_t(\mathbf{u})$. The regret with respect to $\mathbf{u}$ is defined to be,

$$R_T(\mathbf{u}) = \sum_t^T (y_t - \hat{y}_t)^2 - L_T(\mathbf{u}) \; .$$

A desired goal of the algorithm is to have $R_T(\mathbf{u}) = o(T)$, that is, the average loss suffered by the algorithm will converge to the average loss of the best linear function $\mathbf{u}$.

Below in Sec. 5 we will also consider an extension of this form of regret, and evaluate the performance of an algorithm against some T-tuple of functions, $(\mathbf{u}_1, \ldots, \mathbf{u}_T) \in \left(\mathbb{R}^d\right)^T$, $R_T(\mathbf{u}_1, \ldots, \mathbf{u}_T) = \sum_t^T (y_t - \hat{y}_t)^2 - L_T(\mathbf{u}_1, \ldots, \mathbf{u}_T)$, where $L_T(\mathbf{u}_1, \ldots, \mathbf{u}_T)$ $= \sum_t^T \ell_t(\mathbf{u}_t)$. Clearly, with no restriction of the T-tuple, any algorithm may suffer a regret linear in T, as one can set $\mathbf{u}_t = \mathbf{x}_t (y_t / \left\|\mathbf{x}_t\right\|^2)$, and suffer zero quadratic loss in all rounds. Thus, we restrict below the possible choices of T-tuple either explicitly, or implicitly via some penalty.

3 A Last Step Min-Max Algorithm

Our algorithm is derived based on a last-step min-max prediction, proposed by Forster [4] and also Takimoto and Warmuth [10]. An algorithm following this approach outputs

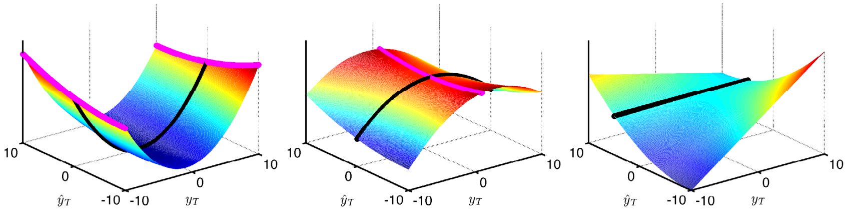


Fig. 1. An illustration of the minmax objective function $G(y_T, \hat{y}_T)$ (5). The black line is the value of the objective as a function of y_T for the optimal predictor $\hat{y}_T$. Left: Forster's optimization function (convex in y_T). Center: our optimization function (strictly concave in y_T, case 1 in Thm. 1). Right: our optimization function (invariant to y_T, case 2 in Thm. 1).

the min-max prediction assuming the current iteration is the last one. The algorithm we describe below is based on an extension of this notion. For this purpose we introduce a weighted cumulative loss using positive input-dependent weights $\{a_t\}_{t=1}^T$,

$$L_T^{\boldsymbol{a}}(\mathbf{u}) = \sum_{t=1}^{T} a_t \left(y_t - \mathbf{u}^\top \mathbf{x}_t\right)^2 \quad , \quad L_T^{\boldsymbol{a}}(\mathbf{u}_1, \ldots, \mathbf{u}_T) = \sum_{t=1}^{T} a_t \left(y_t - \mathbf{u}_t^\top \mathbf{x}_t\right)^2 \ .$$

The exact values of the weights a_t will be defined below.

Our variant of the last step min-max algorithm predicts[1]

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 - \inf_{\mathbf{u}} \left(b \left\|\mathbf{u}\right\|^2 + L_T^{\boldsymbol{a}}(\mathbf{u})\right)\right] \ , \tag{1}$$

for some positive constant $b > 0$. We next compute the actual prediction based on the optimal last step min-max solution. We start with additional notation,

$$\mathbf{A}_t = b\mathbf{I} + \sum_{s=1}^{t} a_s \mathbf{x}_s \mathbf{x}_s^\top \qquad \in \mathbb{R}^{d\times d} \tag{2}$$

$$\mathbf{b}_t = \sum_{s=1}^{t} a_s y_s \mathbf{x}_s \qquad \in \mathbb{R}^{d} \ . \tag{3}$$

The solution of the internal infimum over $\mathbf{u}$ is summarized in the following lemma,

Lemma 1. *For all $t \geq 1$, the function $f(\mathbf{u}) = b\left\|\mathbf{u}\right\|^2 + \sum_{s=1}^{t} a_s \left(y_s - \mathbf{u}^\top \mathbf{x}_s\right)^2$ is minimal at a unique point $\mathbf{u}_t$ given by,*

$$\mathbf{u}_t = \mathbf{A}_t^{-1}\mathbf{b}_t \quad \textit{and} \quad f(\mathbf{u}_t) = \sum_{s=1}^{t} a_s y_s^2 - \mathbf{b}_t^\top \mathbf{A}_t^{-1} \mathbf{b}_t \ . \tag{4}$$

[1] y_T and $\hat{y}_T$ serves both as quantifiers (over the min and max operators, respectively), and as the optimal values over this optimization problem.

The proof is similar to the proof of Lemma 1 by Forster [4]. Substituting (4) back in (1) we get the following form of the minmax problem,

$$\min_{\hat{y}_T} \max_{y_T} G(y_T, \hat{y}_T) \quad \text{for} \quad G(y_T, \hat{y}_T) = \alpha(a_T) y_T^2 + 2\beta(a_T, \hat{y}_T) y_T + \hat{y}_T^2 \ , \tag{5}$$

for some functions $\alpha(a_T)$ and $\beta(a_T, \hat{y}_T)$. Clearly, for this problem to be well defined the function G should be convex in $\hat{y}_T$ and concave in y_T.

A previous choice, proposed by Forster [4], is to have uniform weights and set $a_T = 1$, which for the particular function $\alpha(a_T)$ yields $\alpha(a_T) > 0$. Thus, $G(y_T, \hat{y}_T)$ is a convex function in y_T, implying that the optimal value of G is not bounded from above. Forster [4] addressed this problem by restricting y_T to belong to a predefined interval $[-Y, Y]$, known also to the learner. As a consequence, the adversary optimal prediction is in fact either $y_T = Y$ or $y_T = -Y$, which in turn yields an optimal predictor which is clipped at this bound, $\hat{y}_T = \text{clip}\left(\mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T, Y\right)$, where for $y > 0$ we define $\text{clip}(x, y) = x$ if $|x| \le y$ and $\text{clip}(x, y) = y\,\text{sign}(x)$ otherwise.

This phenomena is illustrated in the left panel of Fig. 1 (best viewed in color). For the minmax optimization function defined by Forster [4], for a constant value of $\hat{y}_T$, the function is convex in y_T, and the adversary would achieve a maximal value at the boundary of the feasible values of y_T interval. That is, either $y_T = Y$ or $y_T = -Y$, as indicated by the two magenta lines at $y_T = \pm 10$. The optimal predictor $\hat{y}_T$ is achieved somewhere along the lines $y_T = Y$ or $y_T = -Y$.

We propose an alternative approach to make the minmax optimal solution bounded by appropriately setting the weight a_T such that $G(y_T, \hat{y}_T)$ is concave in y_T for a constant $\hat{y}_T$. We explicitly consider two cases. First, set a_T such that $G(y_T, \hat{y}_T)$ is *strictly concave* in y_T, and thus attains a single maximum with no need to artificially restrict the value of y_T. In this case the optimal predictor $\hat{y}_T$ is achieved in the unique saddle point, as illustrated in the center panel of Fig. 1. A second case is to set a_T such that $\alpha(a_T) = 0$ and the minmax function $G(y_T, \hat{y}_T)$ becomes linear in y_T. Here, the optimal prediction is achieved by choosing $\hat{y}_T$ such that $\beta(a_T, \hat{y}_T) = 0$ which turns $G(y_T, \hat{y}_T)$ to be invariant to y_T, as illustrated in the right panel of Fig. 1.

Equipped with Lemma 1 we develop the optimal solution of the min-max predictor, summarized in the following theorem.

Theorem 1. *Assume that* $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T \le 0$. *Then the optimal prediction for the last round* T *is*

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T \ . \tag{6}$$

The proof of the theorem makes use of the following technical lemma.

Lemma 2. *For all* $t = 1, 2, \ldots, T$

$$a_t^2 \mathbf{x}_t^\top \mathbf{A}_t^{-1} \mathbf{x}_t + 1 - a_t = \frac{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t}{1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t} \ . \tag{7}$$

The proof is omitted due to lack of space. We now turn to prove the theorem.

Proof. The adversary can choose any y_T, thus the algorithm should predict $\hat{y}_T$ such that the following quantity is minimal,

$$\max_{y_T}\left(\sum_{t=1}^{T}\left(y_t-\hat{y}_t\right)^2-\inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2+\sum_{t=1}^{T}a_t\left(y_t-\mathbf{u}^\top\mathbf{x}_t\right)^2\right)\right)$$

$$\overset{(4)}{=}\max_{y_T}\left(\sum_{t=1}^{T}\left(y_t-\hat{y}_t\right)^2-\sum_{t=1}^{T}a_t y_t^2+\mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T\right) .$$

That is, we need to solve the following minmax problem

$$\min_{\hat{y}_T}\max_{y_T}\left(\sum_{t=1}^{T}\left(y_t-\hat{y}_t\right)^2-\sum_{t=1}^{T}a_t y_t^2+\mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T\right)$$

We use the following relation to re-write the optimization problem,

$$\mathbf{b}_T^\top\mathbf{A}_T^{-1}\mathbf{b}_T=\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{b}_{T-1}+2a_T y_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T+a_T^2 y_T^2\mathbf{x}_T^\top\mathbf{A}_T^{-1}\mathbf{x}_T\ . \quad (8)$$

Omitting all terms that are not depending on y_T and $\hat{y}_T$,

$$\min_{\hat{y}_T}\max_{y_T}\left(\left(y_T-\hat{y}_T\right)^2-a_T y_T^2+2a_T y_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T+a_T^2 y_T^2\mathbf{x}_T^\top\mathbf{A}_T^{-1}\mathbf{x}_T\right)$$

We manipulate the last problem to be of form (5) using Lemma 2,

$$\min_{\hat{y}_T}\max_{y_T}\left(\frac{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T}y_T^2+2y_T\left(a_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T-\hat{y}_T\right)+\hat{y}_T^2\right) , \quad (9)$$

where $\alpha(a_T)=\frac{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T}$ and $\beta(a_T,\hat{y}_T)=a_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T-\hat{y}_T$.

We consider two cases: (1) $1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T<0$ (corresponding to the middle panel of Fig. 1), and (2) $1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T=0$ (corresponding to the right panel of Fig. 1), starting with the first case,

$$1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T<0 \quad (10)$$

Denote the inner-maximization problem by, $f\left(y_T\right)=\frac{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T}y_T^2$ $+\,2y_T\left(a_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T-\hat{y}_T\right)+\hat{y}_T^2$. This function is strictly-concave with respect to y_T because of (10). Thus, it has a unique maximal value given by,

$$\begin{aligned}f^{max}(\hat{y}_T)=&-\frac{a_T}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}\hat{y}_T^2\\&+\frac{2a_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T\left(1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T\right)}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}\hat{y}_T\\&-\frac{\left(a_T\mathbf{b}_{T-1}^\top\mathbf{A}_T^{-1}\mathbf{x}_T\right)^2\left(1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T\right)}{1+a_T\mathbf{x}_T^\top\mathbf{A}_{T-1}^{-1}\mathbf{x}_T-a_T}\end{aligned}$$

Next, we solve $\min_{\hat{y}_T} f^{max}(\hat{y}_T)$, which is strictly-convex with respect to $\hat{y}_T$ because of (10). Solving this problem we get the optimal last step minmax predictor,

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T \left(1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T\right) \ . \tag{11}$$

We further derive the last equation. From (2) we have,

$$\mathbf{A}_T^{-1} a_T \mathbf{x}_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} = \mathbf{A}_T^{-1} \left(\mathbf{A}_T - \mathbf{A}_{T-1}\right) \mathbf{A}_{T-1}^{-1} = \mathbf{A}_{T-1}^{-1} - \mathbf{A}_T^{-1} \tag{12}$$

Substituting (12) in (11) we have the following equality as desired,

$$\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T + \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} a_T \mathbf{x}_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T \ . \tag{13}$$

We now move to the second case for which, $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T - a_T = 0$, which is written equivalently as,

$$a_T = 1/\left(1 - \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T\right) \ . \tag{14}$$

Substituting (14) in (9) we get, $\min_{\hat{y}_T} \max_{y_T} \left(2 y_T \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T - \hat{y}_T\right) + \hat{y}_T^2\right)$. For $\hat{y}_T \neq a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T$, the value of the optimization problem is not-bounded as the adversary may choose $y_T = z^2 \left(a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T - \hat{y}_T\right)$ for $z \to \infty$. Thus, the optimal last step minmax prediction is to set $\hat{y}_T = a_T \mathbf{b}_{T-1}^\top \mathbf{A}_T^{-1} \mathbf{x}_T$. Substituting $a_T = 1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{x}_T$ and following the derivation from (11) to (13) above, yields the desired identity. ■

We conclude by noting that although we did not restrict the form of the predictor $\hat{y}_T$, it turns out that it is a linear predictor defined by $\hat{y}_T = \mathbf{x}_T^\top \boldsymbol{w}_{T-1}$ for $\boldsymbol{w}_{T-1} = \mathbf{A}_{T-1}^{-1} \mathbf{b}_{T-1}$. In other words, the functional form of the optimal predictor is the same as the form of the comparison function class - linear functions in our case. We call the algorithm (defined using (2), (3) and (6)) WEMM for weighted min-max prediction. WEMM can also be seen as an incremental off-line algorithm [1] or follow-the-leader, on a weighted sequence. The prediction $\hat{y}_T = \mathbf{x}_T^\top \boldsymbol{w}_{T-1}$ is with a model that is optimal over a prefix of length $T-1$. The prediction of the optimal predictor defined in (4) is $\mathbf{x}_T^\top \mathbf{u}_{T-1} = \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1} \mathbf{b}_{T-1} = \hat{y}_T$, where $\hat{y}_T$ was defined in (6).

4 Analysis

We analyze the algorithm in two steps. First, in Thm. 2 we show that the algorithm suffers a *constant* regret compared with the optimal weight vector $\mathbf{u}$ evaluated using *the weighted* loss, $L^{\boldsymbol{a}}(\mathbf{u})$. Second, in Thm. 3 and Thm. 4 we show that the difference of the weighted-loss $L^{\boldsymbol{a}}(\mathbf{u})$ to the true loss $L(\mathbf{u})$ is only logarithmic in T or in $\sum_t \ell_t(\mathbf{u})$.

Theorem 2. *Assume $1+a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t \leq 0$ for $t = 1 \ldots T$ (which is satisfied by our choice later). Then, the loss of the last-step minmax predictor (6), $\hat{y}_t = \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t$ for $t = 1 \ldots T$, is upper bounded by,*

$$L_T(alg) \leq \inf_{\mathbf{u} \in \mathbb{R}^d} \left(b \left\|\mathbf{u}\right\|^2 + L_T^{\boldsymbol{a}}(\mathbf{u})\right) \ .$$

Furthermore, if $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1} \mathbf{x}_t - a_t = 0$, then the last inequality is in fact an equality.

Proof Sketch: Long algebraic manipulation given in Sec. A yields,

$$\ell_t(\text{alg}) + \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + L^{\boldsymbol{a}}_{t-1}(\mathbf{u})\right) - \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + L^{\boldsymbol{a}}_{t}(\mathbf{u})\right)$$
$$= \frac{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}\left(y_t - \hat{y}_t\right)^2 \leq 0$$

Summing over t gives the desired bound. ■

Next we decompose the weighted loss $L^{\boldsymbol{a}}_T(\mathbf{u})$ into a sum of the actual loss $L_T(\mathbf{u})$ and a logarithmic term. We give two bounds - one is logarithmic in T (Thm. 3), and the second is logarithmic in $\sum_t \ell_t(\mathbf{u})$ (Thm. 4). We use the following notation of the loss suffered by $\mathbf{u}$ over the worst example,

$$S = S(\mathbf{u}) = \sup_{1\leq t\leq T}\ \ell_t(\mathbf{u}), \tag{15}$$

where clearly S depends explicitly in $\mathbf{u}$, which is omitted for simplicity. We now turn to state our first result,

Theorem 3. *Assume* $\|\mathbf{x}_t\| \leq 1$ *for* $t = 1\ldots T$ *and* $b > 1$. *Assume further that* $a_t = \frac{1}{1-\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}$ *for* $t = 1\ldots T$. *Then* $L^{\boldsymbol{a}}_T(\mathbf{u}) \leq L_T(\mathbf{u}) + \frac{b}{b-1}S\ln\left|\frac{1}{b}\mathbf{A}_T\right|$.

The proof follows similar steps to Forster [4]. A detailed proof is given in Sec. B.

Proof Sketch: We decompose the weighted loss,

$$L^{\boldsymbol{a}}_T(\mathbf{u}) = L_T(\mathbf{u}) + \sum_t(a_t - 1)\ell_t(\mathbf{u}) \leq L_T(\mathbf{u}) + S\sum_t(a_t - 1)\ . \tag{16}$$

From the definition of a_t we have, $a_t - 1 = a_t^2\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t \leq \frac{b}{b-1}a_t\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t$ (see (30)). Finally, following similar steps to Forster [4] we have, $\sum_{t=1}^T a_t\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t \leq \ln\left|\frac{1}{b}\mathbf{A}_T\right|$. ■

Next we show a bound that may be sub-logarithmic if the comparison vector $\mathbf{u}$ suffers sub-linear amount of loss. Such a bound was previously proposed by Orabona et.al [9]. We defer the discussion about the bound after providing the proof below.

Theorem 4. *Assume* $\|\mathbf{x}_t\| \leq 1$ *for* $t = 1\ldots T$, *and* $b > 1$. *Assume further that*

$$a_t = \frac{1}{1 - \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \tag{17}$$

for $t = 1\ldots T$. *Then,*

$$L^{\boldsymbol{a}}_T(\mathbf{u}) \leq L_T(\mathbf{u}) + \frac{b}{b-1}Sd\left[1 + \ln\left(1 + \frac{L_T\left(\mathbf{u}\right)}{Sd}\right)\right]\ . \tag{18}$$

We prove the theorem with a refined bound on the sum $\sum_t(a_t - 1)\ell_t(\mathbf{u})$ of (16) using the following two lemmas. In Thm. 3 we bound the loss of all examples with S and then bound the remaining term. Here, instead we show a relation to a subsequence "pretending" all examples of it as suffering a loss S, yet with the same cumulative loss, yielding an effective shorter sequence, which we then bound. In the next lemma we show how to find this subsequence, and in the following one bound the performance.

Lemma 3. *Let $I \subset \{1 \ldots T\}$ be the indices of the $T^{'} = \left\lceil \sum_{t=1}^{T} \ell_t / S \right\rceil$ largest elements of a_t, that is $|I| = T'$ and $\min_{t \in I} a_t \geq a_\tau$ for all $\tau \in \{1 \ldots T\}/I$. Then,*

$$\sum_{t=1}^{T} \ell_t \left(\mathbf{u}\right) \left(a_t - 1\right) \leq S \sum_{t \in I} \left(a_t - 1\right) \ .$$

Proof. For a vector $\boldsymbol{v} \in \mathbb{R}^T$ define by $I(\boldsymbol{v})$ the set of indicies of the $T^{'}$ maximal absolute-valued elements of $\boldsymbol{v}$, and define $f(\boldsymbol{v}) = \sum_{t \in I(\boldsymbol{v})} |\boldsymbol{v}_t|$. The function $f(\boldsymbol{v})$ is a norm [3] with a dual norm $g(\mathbf{u}) = \max\left\{\|\mathbf{u}\|_\infty, \frac{\|\mathbf{u}\|_1}{T'}\right\}$. From the property of dual norms we have $\boldsymbol{v} \cdot \mathbf{u} \leq f(\boldsymbol{v}) g(\mathbf{u})$. Applying this inequality to $\boldsymbol{v} = (a_1 - 1 ... a_T - 1)$ and $\mathbf{u} = (\ell_1 \ldots \ell_T)$ we get, $\sum_{t=1}^{T} \ell_t(\mathbf{u})(a_t - 1) \leq \max\left\{S, \frac{\sum_{t=1}^{T} \ell_t}{T'}\right\} \sum_{t \in I}(a_t - 1)$. Combining with $ST' \geq \sum_{t=1}^{T} \ell_t$, completes the proof. ■

Note that the quantity $\sum_{t \in I} a_t$ is based only on T' examples, yet was generated using all T examples. In fact by running the algorithm with only these T' examples the corresponding sum cannot get smaller. Specifically, assume the algorithm is run with inputs $(\mathbf{x}_1, y_1)...(\mathbf{x}_T, y_T)$ and generated a corresponding sequence $(a_1...a_T)$. Let I be the set of indices with maximal values of a_t as before. Assume the algorithm is run with the subsequence of examples from I (with the same order) and generated $\alpha_1 ... \alpha_T$ (where we set $\alpha_t = 0$ for $t \notin I$). Then, $\alpha_t \geq a_t$ for all $t \in I$. This statement follows from (2) from which we get that the matrix $\mathbf{A}_t$ is monotonically increasing in t. Thus, by removing examples we get another smaller matrix which leads to a larger value of α_t.

We continue the analysis with a sequence of length T' rather than a subsequence of the original sequence of length T being analyzed. The next lemma upper bounds the sum $\sum_{t}^{T'} a_t$ over T' inputs with another sum of same length, yet using orthonormal set of vectors of size d.

Lemma 4. *Let $\mathbf{x}_1 ... \mathbf{x}_\tau$ be any τ inputs with unit-norm. Assume the algorithm is performing updates using (17) for some $\mathbf{A}_0$ resulting in a sequence $a_1 ... a_\tau$. Let $E = \{\boldsymbol{v}_1 ... \boldsymbol{v}_d\} \subset \mathbb{R}^d$ be an eigen-decomposition of $\mathbf{A}_0$ with corresponding eigenvalues $\lambda_1 ... \lambda_d$. Then there exists a sequence of indices $j_1 ... j_\tau$, where $j_i \in \{1...d\}$, such that $\sum_t a_t \leq \sum_t \alpha_t$, where α_t are generated using (17) on the sequence $\boldsymbol{v}_{j_1} ... \boldsymbol{v}_{j_\tau}$.*

Additionally, let n_s be the number of times eigenvector $\boldsymbol{v}_s$ is used ($s = 1...d$), that is $n_s = |\{j_t \ : \ j_t = s\}|$ (and $\sum_s n_s = \tau$), then,

$$\sum_t \alpha_t \leq \tau + \sum_{s=1}^{d} \sum_{r=1}^{n_s} \frac{1}{\lambda_s + r - 2} \ .$$

Proof. By induction over τ. For $\tau = 1$ we want to upper bound $a_1 = 1/(1 - \mathbf{x}_1^\top \mathbf{A}_0^{-1} \mathbf{x}_1)$ which is maximized when $\mathbf{x}_1 = \boldsymbol{v}_d$ the eigenvector with minimal eigenvalue λ_d, in this case we have $\alpha_1 = 1/(1 - 1/\lambda_d) = 1 + 1/(\lambda_d - 1)$, as desired.

Table 1. Comparison of regret bounds for online regression

Algorithm	Bound on Regret $R_T(\mathbf{u})$
Vovk [13]	$b\left\|\mathbf{u}\right\|^2 + dY^2 \ln(1+T/(db))$
Forster [4]	$b\left\|\mathbf{u}\right\|^2 + dY^2 \ln(1+T/(db))$
Orabona et.al. [9]	$2\left\|\mathbf{u}\right\|^2 + d(U+Y)^2 \ln\left(1+\frac{2\|\mathbf{u}\|^2+\sum_t \ell_t(\mathbf{u})}{d(U+Y)^2}\right)$
Thm. 3	$b\left\|\mathbf{u}\right\|^2 + Sd\frac{b}{b-1}\ln\left(1+\frac{T}{d(b-1)}\right)$
Thm. 4	$b\left\|\mathbf{u}\right\|^2 + Sd\frac{b}{b-1}\ln(1+\frac{\sum_t \ell_t(\mathbf{u})}{Sd})$

Next we assume the lemma holds for some $\tau-1$ and show it for τ. Let $\mathbf{x}_1$ be the first input, and let $\{\gamma_s\}$ and $\{\mathbf{u}_s\}$ be the eigen-values and eigen-vectors of $\mathbf{A}_1 = \mathbf{A}_0 + a_1\mathbf{x}_1\mathbf{x}_1^\top$. The assumption of induction implies that $\sum_{t=2}^{\tau}\alpha_t \leq (\tau-1) + \sum_{s=1}^{d}\sum_{r=1}^{n_s}\frac{1}{\gamma_s+r-2}$. From Theorem 8.1.8 of [6] we know that the eigenvalues of $\mathbf{A}_1$ satisfy $\gamma_s = \lambda_s + m_s$ for some $m_s \geq 0$ and $\sum_s m_s = 1$. We thus conclude that

$$\sum_t a_t \leq 1 + 1/(\lambda_d - 1) + (\tau-1) + \sum_{s=1}^{d}\sum_{r=1}^{n_s}\frac{1}{\lambda_s + m_s + r - 2} \ .$$

The last term is convex in $m_1 ... m_d$ and thus is maximized over a vertex of the simplex, that is when $m_u = 1$ for some u and zero otherwise. In this case, the eigen-vectors $\{\mathbf{u}_s\}$ of $\mathbf{A}_1$ are in fact the eigenvectors $\{\boldsymbol{v}_s\}$ of $\mathbf{A}_0$, and the proof is completed. ■

Equipped with these lemmas we now prove Thm. 4.

Proof. Let $T' = \left\lceil \sum_{t=1}^{T}\ell_t/S \right\rceil$. Our starting point is the equality $L_T^{\boldsymbol{a}}(\mathbf{u}) = L_T(\mathbf{u}) + \sum_{t=1}^{T}\ell_t\left(\mathbf{u}\right)\left(a_t - 1\right)$ stated in (16). From Lemma 3 we get,

$$\sum_{t=1}^{T}\ell_t\left(\mathbf{u}\right)\left(a_t-1\right) \leq S\sum_{t\in I}\left(a_t-1\right) \leq S\sum_{t}^{T'}\left(\alpha_t-1\right) \ , \tag{19}$$

where I is the subset of T' indices for which a_t are maximal, and α_t are the resulting coefficients computed with (17) using only the sub-sequence of examples $\mathbf{x}_t$ with $t \in I$.

By definition $\mathbf{A}_0 = b\mathbf{I}$ and thus from Lemma 4 we further bound (19) with,

$$\sum_{t=1}^{T}\ell_t\left(\mathbf{u}\right)\left(a_t-1\right) \leq S\sum_{s=1}^{d}\sum_{r=1}^{n_s}\frac{1}{b+r-2} \ , \tag{20}$$

for some n_s such that $\sum_s n_s = T'$. The last equation is maximized when all the counts n_s are about (as d may not divide T') the same, and thus we further bound (20) with,

$$\begin{aligned}\sum_{t=1}^{T}\ell_t\left(\mathbf{u}\right)\left(a_t-1\right) \qquad & \leq S\sum_{s=1}^{d}\sum_{r=1}^{\lceil T'/d\rceil}\frac{1}{b+r-2} \leq Sd\sum_{r=1}^{\lceil T'/d\rceil}\frac{b}{b-1}\frac{1}{r} \\ \leq \ Sd\frac{b}{b-1}\left(1+\ln\left(\left\lceil\frac{T'}{d}\right\rceil\right)\right) \quad & \leq Sd\frac{b}{b-1}\left(1+\ln\left(1+\frac{L_T\left(\mathbf{u}\right)}{Sd}\right)\right) \ ,\end{aligned}$$

which completes the proof. ■

It is instructive to compare bounds of similar algorithms, summarized in Table 1. Our first bound of Thm. 3 is most similar to the bounds of Forster [4] and Vovk [13]. The bound in the table is obtained by noting that $\log\det$ is a concave function of the eigenvalues of the matrix, upper bounded when all the eigenvalues are equal (with the same trace). They have a multiplicative factor Y^2 of the logarithm, while we have the worst-loss of $\mathbf{u}$ over all examples. Thus, our first bound is better on problems that are approximately linear $y_t \approx \mathbf{u} \cdot \mathbf{x}_t$ for $t = 1, \ldots, T$ and Y is large, and their bound is better if Y is small. Note that the analysis of Forster [4] assumes that the labels y_t are bounded, and formally the algorithm should know this bound, while we assume that the inputs are bounded.

Our second bound of Thm. 4 is similar to the bound of Orabona et.al. [9]. Both bounds have potentially sub-logarithmic regret as the cumulative loss $L(\mathbf{u})$ may be sublinear in T. Yet, their bound has a multiplicative factor of $(U+Y)^2$, while our bound has only the maximal loss S, which, as before, can be much smaller. Additionally, their analysis assumes that both the inputs $\mathbf{x}_t$ and the labels y_t are bounded, while we only assume that the inputs are bounded, and furthermore, our algorithm does not need to assume and know a compact set which contains $\mathbf{u}$ ($\|\mathbf{u}\| \leq U$), as opposed to their algorithm.

5 Learning in Non-stationary Environment

In this section we present a generalization of the last-step min-max predictor for non-stationary problems given in (1). We define the predictor to be,

$$\hat{y}_T = \arg\min_{\hat{y}_T} \max_{y_T} \left[\sum_{t=1}^{T} (y_t - \hat{y}_t)^2 - \inf_{\mathbf{u}_1,\ldots,\mathbf{u}_T,\bar{\mathbf{u}}} \left(b \left\|\bar{\mathbf{u}}\right\|^2 + cV_m + L_T^{\tilde{\boldsymbol{a}}}(\mathbf{u}_1, \ldots, \mathbf{u}_T) \right) \right] \tag{21}$$

for $V_m = \sum_{t=1}^{T} \left\|\mathbf{u}_t - \bar{\mathbf{u}}\right\|^2$, positive constants $b, c > 0$ and weights $\tilde{a}_t \geq 1$ for $1 \leq t \leq T$.

As mentioned above, we use an extended notion of function class, using different vectors $\mathbf{u}_t$ across time T. We circumvent here the problem mentioned in the end of Sec. 2, and restrict the adversary from choosing an arbitrary T-tuple $(\mathbf{u}_1, \ldots, \mathbf{u}_T)$ by introducing a reference weight-vector $\bar{\mathbf{u}}$. Specifically, indeed we replace the single-weight cumulative-loss $L_T^{\boldsymbol{a}}(\mathbf{u})$ in (1) with a multi-weight cumulative-loss $L_T^{\tilde{\boldsymbol{a}}}(\mathbf{u}_1, \ldots, \mathbf{u}_T)$ in (21), yet, we add the term cV_m to (21) penalizing a T-tuple $(\mathbf{u}_1, \ldots, \mathbf{u}_T)$ that its elements $\{\mathbf{u}_t\}$ are far from some single point $\bar{\mathbf{u}}$. Intuitively, V_m serves as a measure of complexity of the T-tuple by measuring the deviation of its elements from some vector.

The new formulation of (21) clearly subsumes the formulation of (1), as if $\mathbf{u}_1 = \ldots \mathbf{u}_T = \bar{\mathbf{u}} = \mathbf{u}$, then (21) reduces to (1). We now show that in-fact the two notions of last-step min-max predictors are equivalent. The following lemma characterizes the solution of the inner infimum of (21) over $\bar{\mathbf{u}}$.

Lemma 5. *For any* $\bar{\mathbf{u}} \in \mathbb{R}^d$*, the function* $J(\mathbf{u}_1, \ldots, \mathbf{u}_T) = b\|\bar{\mathbf{u}}\|^2 + c\sum_{t=1}^T \|\mathbf{u}_t - \bar{\mathbf{u}}\|^2 + \sum_{t=1}^T \tilde{a}_t \left(y_t - \mathbf{u}_t^\top \mathbf{x}_t\right)^2$ *is minimal for* $\mathbf{u}_t = \bar{\mathbf{u}} + \frac{c^{-1}}{\tilde{a}_t^{-1} + c^{-1}\|x_t\|^2}\left(y_t - \bar{\mathbf{u}}^\top \mathbf{x}_t\right)\mathbf{x}_t$ *for* $t = 1...T$*. The minimal value of* $J(\mathbf{u}_1, \ldots, \mathbf{u}_T)$ *is given by*

$$J_{min} = b\|\bar{\mathbf{u}}\|^2 + \sum_{t=1}^T \frac{1}{\tilde{a}_t^{-1} + c^{-1}\|\mathbf{x}_t\|^2}\left(y_t - \bar{\mathbf{u}}^\top \mathbf{x}_t\right)^2 . \tag{22}$$

The proof is omitted due to space constraints, and is obtained by setting the derivative of the objective to zero. Substituting (22) in (21) we obtain the following form of the last-step minmax predictor,

$$\begin{aligned}\hat{y}_T = \arg\min_{\hat{y}_T}\max_{y_T} & \left[\sum_{t=1}^T (y_t - \hat{y}_t)^2 - \right. \\ & \left. \inf_{\bar{\mathbf{u}} \in \mathbb{R}^d}\left(b\|\bar{\mathbf{u}}\|^2 + \sum_{t=1}^T \frac{1}{\tilde{a}_t^{-1} + c^{-1}\|\mathbf{x}_t\|^2}\left(y_t - \mathbf{x}_t^\top \bar{\mathbf{u}}\right)^2\right)\right]\end{aligned} \tag{23}$$

Clearly, both equations (23) and (1) are equivalent when identifying,

$$a_t = 1/\left(\tilde{a}_t^{-1} + c^{-1}\|\mathbf{x}_t\|^2\right) . \tag{24}$$

Therefore, we can use the results of the previous sections. Specifically, if $1 + a_T \mathbf{x}_T^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T - a_T \leq 0$ the optimal predictor developed in Thm. 1 for the stationary case is given by, $\hat{y}_T = \mathbf{b}_{T-1}^\top \mathbf{A}_{T-1}^{-1}\mathbf{x}_T$ where we replace (2) with $\mathbf{A}_t = b\mathbf{I} + \sum_{s=1}^t a_s \mathbf{x}_s \mathbf{x}_s^\top = b\mathbf{I} + \sum_{s=1}^t \frac{1}{\tilde{a}_s^{-1} + c^{-1}\|\mathbf{x}_s\|^2}\mathbf{x}_s\mathbf{x}_s^\top$ and (3) with $\mathbf{b}_t = \sum_{s=1}^t a_s y_s \mathbf{x}_s = \sum_{s=1}^t \frac{1}{\tilde{a}_s^{-1} + c^{-1}\|\mathbf{x}_s\|^2} y_s \mathbf{x}_s$. Although most of the analysis above holds for $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t \leq 0$ in the end of the day, Thm. 3 assumed that this inequality holds as equality. Substituting $a_t = \frac{1}{1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t}$ in (24) and solving for $\tilde{a}_t$ we obtain,

$$\tilde{a}_t = 1/\left(1 - \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - c^{-1}\|\mathbf{x}_t\|^2\right) . \tag{25}$$

The last-step minmax predictor (21) is convex if $\tilde{a}_t \geq 0$, which holds if, $1/b + 1/c \leq 1$, because $\mathbf{A}_{t-1}^{-1} \preceq \mathbf{A}_0^{-1} = (1/b)\mathbf{I}$ and we assume that $\|\mathbf{x}_t\|^2 \leq 1$.

Let us state the analogous statements of Thm. 2 and Thm. 3. Substituting Lemma 5 in Thm. 2 we bound the cumulative loss of the algorithm with the weighted loss of any T-tuple $(\mathbf{u}_1, \ldots, \mathbf{u}_T)$,

Corollary 1. *Assume* $\|\mathbf{x}_t\| \leq 1$, $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t \leq 0$ *for* $t = 1 \ldots T$*, and* $1/b + 1/c \leq 1$*. Then, the loss of the last-step minmax predictor,* $\hat{y}_t = \mathbf{b}_{t-1}^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t$ *for* $t = 1 \ldots T$*, is upper bounded by,* $L_T(alg) \leq \inf_{\mathbf{u} \in \mathbb{R}^d}\left(b\|\mathbf{u}\|^2 + L_T^{\mathbf{a}}(\mathbf{u})\right) = \inf_{\mathbf{u}_1, \ldots, \mathbf{u}_T, \bar{\mathbf{u}}}\left(b\|\bar{\mathbf{u}}\|^2 + c\sum_{t=1}^T \|\mathbf{u}_t - \bar{\mathbf{u}}\|^2 + L_T^{\tilde{\mathbf{a}}}(\mathbf{u}_1, \ldots, \mathbf{u}_T)\right)$*. Furthermore, if* $1 + a_t \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t = 0$*, then the last inequality is in fact an equality.*

Next we relate the weighted cumulative loss $L_T^{\tilde{\mathbf{a}}}(\mathbf{u}_1,\ldots,\mathbf{u}_T)$ to the loss itself $L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T)$,

Corollary 2. *Assume* $\|\mathbf{x}_t\| \leq 1$ *for* $t = 1\ldots T$, $b > 1$ *and* $1/b + 1/c \leq 1$. *Assume additionally that* $\tilde{a}_t = \frac{1}{1-\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - c^{-1}\|\mathbf{x}_t\|^2}$ *as given in* (25). *Then*

$$L_T^{\tilde{\mathbf{a}}}(\mathbf{u}_1,\ldots,\mathbf{u}_T) \leq L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T) + \frac{b}{b-1} S \ln\left|\frac{1}{b}\mathbf{A}_T\right| + TS\frac{1}{c\left(1-b^{-1}\right)^2 - \left(1-b^{-1}\right)}$$

Proof. We start as in the proof of Thm. 3 and decompose the weighted loss,

$$\begin{aligned} L_T^{\tilde{\mathbf{a}}}(\mathbf{u}_1,\ldots,\mathbf{u}_T) &= L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T) + \sum_t (\tilde{a}_t - 1)\ell_t(\mathbf{u}_t) \\ &\leq L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T) + S\sum_t (a_t - 1) + S\sum_t (\tilde{a}_t - a_t) . \end{aligned} \tag{26}$$

We bound the sum of the third term,

$$\begin{aligned} \tilde{a}_t - a_t &= \frac{1}{1-\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - c^{-1}\left\|\mathbf{x}_t\right\|^2} - \frac{1}{1-\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \\ &\leq \frac{c^{-1}}{\left(1-b^{-1}-c^{-1}\right)\left(1-b^{-1}\right)} = \frac{1}{c\left(1-b^{-1}\right)^2 - \left(1-b^{-1}\right)} . \end{aligned} \tag{27}$$

Additionally, as in Thm. 3 the second term is bounded with $\frac{b}{b-1}S\ln\left|\frac{1}{b}\mathbf{A}_T\right|$. Substituting this bound and (27) in (26) completes the proof. ■

Combining the last two corollaries yields the main result of this section.

Corollary 3. *Under the conditions of Cor. 2 the cumulative loss of the last-step minmax predictor is upper bounded by,*

$$L_T(alg) \leq \inf_{\mathbf{u}_1,\ldots,\mathbf{u}_T,\bar{\mathbf{u}}} \left(b\left\|\bar{\mathbf{u}}\right\|^2 + cV_m + L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T) + \frac{Sb}{b-1}\ln\left|\frac{1}{b}\mathbf{A}_T\right| + \frac{TS}{c\left(1-b^{-1}\right)^2 - \left(1-b^{-1}\right)} \right) ,$$

where V_m *is the deviation of* $\{\mathbf{u}_t\}$ *from some fixed weight-vector. Additionally, setting* $c_V = \frac{b}{b-1}\left(1+\sqrt{\frac{ST}{V_m}}\right)$ *minimizing the above bound over c,*

$$L_T(alg) \leq \inf_{\mathbf{u}_1,\ldots,\mathbf{u}_T,\bar{\mathbf{u}}} \left(b\left\|\bar{\mathbf{u}}\right\|^2 + L_T(\mathbf{u}_1,\ldots,\mathbf{u}_T) + \frac{Sb}{b-1}\ln\left|\frac{1}{b}\mathbf{A}_T\right| + \frac{b}{b-1}\left(V_m + 2\sqrt{STV_m}\right) \right) .$$

Few comments. First, it is straightforward to verify that $c_V = \frac{b}{b-1}\left(1 + \sqrt{\frac{ST}{V_m}}\right)$ satisfy the constraint $1/b + 1/c_V \leq 1$. Second, this bound strictly generalizes the bound for the stationary case, since Cor. 2 reduces to Thm. 3 when all the weight-vectors equal each other $\mathbf{u}_1 = \ldots \mathbf{u}_T = \bar{\mathbf{u}}$ (i.e. $V_m = 0$). Third, the constant c (or c_V) is not used by the algorithm, but only in the analysis. So there is no need to know the actual deviation V_m to tune the algorithm. In other words, the bound applies essentially to the same last step minmax predictor defined in Thm. 1. Finally, we can obtain a bound for the non-stationary case based on Thm. 4 instead of Thm. 3 (omitted due to lack of space).

6 Related Work and Summary

The problem of predicting reals in an online manner was studied for more than five decades. Clearly we cannot cover all previous work here, and the reader is refered to the encyclopedic book of Cesa-Bianchi and Lugosi [2] for a full survey.

An online version of the ridge regression algorithm in the worst-case setting was proposed and analyzed by Foster [5]. A related algorithm called the Aggregating Algorithm (AA) was studied by Vovk [12]. The recursive least squares (RLS) [7] is a similar algorithm proposed for adaptive filtering. Both algorithms make use of second order information, as they maintain a weight-vector and a covariance-like positive semi-definite (PSD) matrix used to re-weight the input. The eigenvalues of this covariance-like matrix grow with time t, a property which is used to prove logarithmic regret bounds. Orabona et.al. [9] showed that beyond logarithmic regret bound can be achieved when the total best linear model loss is sublinear in T. We derive a similar bound, with a multiplicative factor that depends on the worst-loss of $\mathbf{u}$, rather than a bound Y on the labels.

The derivation of our algorithm shares similarities with the work of Forster [4]. Both algorithms are motivated from the last-step min-max predictor. Yet, the formulation of Forster [4] yields a convex optimization for which the max operation over y_t is not bounded, and thus he used an artificial clipping operation to avoid unbounded solutions. With a proper tuning of a_t and a weighted loss, we are able to obtain a problem that is convex in $\hat{y}_t$ and concave in y_t, and thus well defined.

Most recent work is focused in the stationary setting. We also discuss a specific weak-notion of non-stationary setting, for which the few weight-vectors can be used for comparison and their total deviation is computed with respect to some single weight-vector. Recently, Vaits and Crammer [11] proposed an algorithm designed for non-stationary environments. Herbster and Warmuth [8] discussed general gradient descent algorithms with projection of the weight-vector using the Bregman divergence, and Zinkevich [14] developed an algorithm for online convex programming. They all use a stronger notion of diversity between vectors, as their distance is measured with consecutive vectors (that is drift that may end far from the starting point). Thus, the bounds cannot be compared in general.

Summary: We proposed a modification of the last-step min-max algorithm [4] using weights over examples, and showed how to choose these weights for the problem to be well defined – convex – which enabled us to develop the last step min-max predictor, without requiring the labels to be bounded. Our analysis bounds the regret with quantities that depends only on the loss of the competitor, with no need for any knowledge of the problem. Our prediction algorithm was motivated from the last-step minmax predictor problem for stationary setting, but we showed that the same algorithm can be used to derive a bound for a class of *non-stationary* problems as well.

We plan to perform an extensive empirical study comparing the algorithm to other algorithms. An interesting direction would be to extend the algorithm for general loss functions rather than the squared loss, or to classification tasks.

Acknowledgements: The research is partially supported by an Israeli Science Foundation grant ISF- 1567/10.

A Proof of Thm. 2

Proof. Using the Woodbury matrix identity we get

$$\mathbf{A}_t^{-1} = \mathbf{A}_{t-1}^{-1} - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}}{\frac{1}{a_t} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \tag{28}$$

therefore

$$\mathbf{A}_t^{-1}\mathbf{x}_t = \mathbf{A}_{t-1}^{-1}\mathbf{x}_t - \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{\frac{1}{a_t} + \mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} = \frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} \tag{29}$$

For $t = 1 \ldots T$ we have

$$\begin{aligned}
&\ell_t(\text{alg}) + \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + L_{t-1}^{\boldsymbol{a}}(\mathbf{u})\right) - \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + L_t^{\boldsymbol{a}}(\mathbf{u})\right)\\
&\overset{(4)}{=} \left(y_t - \hat{y}_t\right)^2 + \sum_{s=1}^{t-1} a_s y_s^2 - \mathbf{b}_{t-1}^\top\mathbf{A}_{t-1}^{-1}\mathbf{b}_{t-1} - \sum_{s=1}^{t} a_s y_s^2 + \mathbf{b}_t^\top\mathbf{A}_t^{-1}\mathbf{b}_t\\
&\overset{(8)}{=} \left(y_t - \hat{y}_t\right)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top\mathbf{A}_{t-1}^{-1}\mathbf{b}_{t-1} + \mathbf{b}_{t-1}^\top\mathbf{A}_t^{-1}\mathbf{b}_{t-1} + 2a_t y_t\mathbf{b}_{t-1}^\top\mathbf{A}_t^{-1}\mathbf{x}_t\\
&+a_t^2 y_t^2\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t\\
&\overset{(12)}{=} \left(y_t - \hat{y}_t\right)^2 - a_t y_t^2 - \mathbf{b}_{t-1}^\top\mathbf{A}_t^{-1}a_t\mathbf{x}_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{b}_{t-1} + 2a_t y_t\mathbf{b}_{t-1}^\top\mathbf{A}_t^{-1}\mathbf{x}_t\\
&+a_t^2 y_t^2\mathbf{x}_t^\top\mathbf{A}_t^{-1}\mathbf{x}_t\\
&\overset{(29)}{=} \left(y_t - \hat{y}_t\right)^2 - a_t y_t^2 + a_t\left(-\hat{y}_t\mathbf{b}_{t-1}^\top + 2y_t\mathbf{b}_{t-1}^\top + a_t y_t^2\mathbf{x}_t^\top\right)\frac{\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}\\
&= \left(y_t - \hat{y}_t\right)^2 - a_t\frac{\left(y_t - \hat{y}_t\right)^2}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t} = \frac{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t - a_t}{1 + a_t\mathbf{x}_t^\top\mathbf{A}_{t-1}^{-1}\mathbf{x}_t}\left(y_t - \hat{y}_t\right)^2 \leq 0
\end{aligned}$$

Summing over $t \in \{1, \ldots, T\}$ yields $L_T(\text{alg}) - \inf_{\mathbf{u}\in\mathbb{R}^d}\left(b\left\|\mathbf{u}\right\|^2 + L_T^{\boldsymbol{a}}(\mathbf{u})\right) \leq 0\,.$ ■

B Proof of Thm. 3

Proof. From (28) we see that $\mathbf{A}_t^{-1} \prec \mathbf{A}_{t-1}^{-1}$ and because $\mathbf{A}_0 = b\mathbf{I}$ we get $\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t < \mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t < \mathbf{x}_t^\top \mathbf{A}_{t-2}^{-1}\mathbf{x}_t < \ldots < \mathbf{x}_t^\top \mathbf{A}_0^{-1}\mathbf{x}_t = \frac{1}{b}\left\|\mathbf{x}_t\right\|^2 \leq \frac{1}{b}$ therefore $1 \leq a_t \leq \frac{1}{1-\frac{1}{b}} = \frac{b}{b-1}$. From (29) we have $\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t = \frac{\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t}{1+a_t\mathbf{x}_t^\top \mathbf{A}_{t-1}^{-1}\mathbf{x}_t} = \frac{1-\frac{1}{a_t}}{1+a_t\left(1-\frac{1}{a_t}\right)} = \frac{a_t-1}{a_t^2}$ so we can bound the term $a_t - 1$ as following

$$a_t - 1 = a_t^2\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t \leq \frac{b}{b-1}a_t\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t\ . \tag{30}$$

With an argument similar to [4] we have, $a_t\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t \leq \ln\frac{|\mathbf{A}_t|}{\left|\mathbf{A}_t - a_t\mathbf{x}_t\mathbf{x}_t^\top\right|} = \ln\frac{|\mathbf{A}_t|}{|\mathbf{A}_{t-1}|}$. Summing the last inequality over t and using the initial value $\ln\left|\frac{1}{b}\mathbf{A}_0\right| = 0$ we get $\sum_{t=1}^T a_t\mathbf{x}_t^\top \mathbf{A}_t^{-1}\mathbf{x}_t \leq \ln\left|\frac{1}{b}\mathbf{A}_T\right|$. Substituting the last equation in (30) we get the logarithmic bound $\sum_{t=1}^T (a_t - 1) \leq \frac{b}{b-1}\ln\left|\frac{1}{b}\mathbf{A}_T\right|$, as required. ■

References

[1] Azoury, K.S., Warmuth, M.W.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Machine Learning 43(3), 211–246 (2001)

[2] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press, New York (2006)

[3] Dekel, O., Long, P.M., Singer, Y.: Online learning of multiple tasks with a shared loss. Journal of Machine Learning Research 8, 2233–2264 (2007)

[4] Forster, J.: On Relative Loss Bounds in Generalized Linear Regression. In: Ciobanu, G., Păun, G. (eds.) FCT 1999. LNCS, vol. 1684, pp. 269–280. Springer, Heidelberg (1999)

[5] Foster, D.P.: Prediction in the worst case. The An. of Stat. 19(2), 1084–1090 (1991)

[6] Golub, G.H., Van Loan, C.F.: Matrix computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)

[7] Hayes, M.H.: 9.4: Recursive least squares. In: Statistical Digital Signal Processing and Modeling, p. 541 (1996)

[8] Herbster, M., Warmuth, M.K.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)

[9] Orabona, F., Cesa-Bianchi, N., Gentile, C.: Beyond logarithmic bounds in online learning. In: AISTATS (2012)

[10] Takimoto, E., Warmuth, M.K.: The Last-Step Minimax Algorithm. In: Arimura, H., Sharma, A.K., Jain, S. (eds.) ALT 2000. LNCS (LNAI), vol. 1968, pp. 279–290. Springer, Heidelberg (2000)

[11] Vaits, N., Crammer, K.: Re-adapting the Regularization of Weights for Non-stationary Regression. In: Kivinen, J., Szepesvári, C., Ukkonen, E., Zeugmann, T. (eds.) ALT 2011. LNCS, vol. 6925, pp. 114–128. Springer, Heidelberg (2011)

[12] Vovk, V.G.: Aggregating strategies. In: Proceedings of the Third Annual Workshop on Computational Learning Theory, pp. 371–383. Morgan Kaufmann (1990)

[13] Vovk, V.: Competitive on-line statistics. International Statistical Review 69 (2001)

[14] Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 928–936 (2003)

Online Prediction under Submodular Constraints

Daiki Suehiro[1], Kohei Hatano[1], Shuji Kijima[1]
, Eiji Takimoto[1], and Kiyohito Nagano[2]

[1] Department of Informatics, Kyushu University
[2] University of Tokyo
{daiki.suehiro,hatano,kijima,eiji}@inf.kyushu-u.ac.jp,
nagano@sat.t.u-tokyo.ac.jp

Abstract. We consider an online prediction problem of combinatorial concepts where each combinatorial concept is represented as a vertex of a polyhedron described by a submodular function (base polyhedron). In general, there are exponentially many vertices in the base polyhedron. We propose polynomial time algorithms with regret bounds. In particular, for cardinality-based submodular functions, we give $O(n^2)$-time algorithms.

1 Introduction

Online learning of combinatorial or structured concepts have gained much attention these days [9, 2, 13, 17]. Such combinatorial concepts includes shortest paths, k-sets, spanning trees, permutations, and so on. In typical settings, we assume a finite set $\mathcal{C}$ of combinatorial concepts where each concept can be represented as a vector in $\mathbb{R}^n$ for some fixed n, i.e., $\mathcal{C} \subseteq \mathbb{R}^n$. Then we consider the following protocol: For each trial $t = 1, \ldots, T$, (i) the player predicts $\boldsymbol{c}_t \in \mathcal{C}$, (ii) the adversary returns a loss vector $\boldsymbol{\ell}_t \in [0,1]^n$, and (iii) the player incurs loss $\boldsymbol{c}_t \cdot \boldsymbol{\ell}_t$. The goal of the player is to minimize the regret: $\sum_{t=1}^{T} \boldsymbol{c}_t \cdot \boldsymbol{\ell}_t - \min_{\boldsymbol{c} \in \mathcal{C}} \sum_{t=1}^{T} \boldsymbol{c} \cdot \boldsymbol{\ell}_t$.

There are some approaches to attack this type of problems. A naive approach to minimize the regret in the above problem is to apply Hedge algorithm [6]. Hedge algorithm combines experts predictions, where each expert corresponds to each concept in $\mathcal{C}$. In general, however, the size of $\mathcal{C}$ is exponentially large w.r.t. n. Therefore, a straightforward implementation of Hedge algorithm is inefficient. There are some efficient online prediction methods for combinatorial concepts, for example, PermELearn [9] and Component Hedge [13] and Comband in bandit setting [2]. These methods consist of abstract subroutines.

Among subroutines, projection and decomposition are important and used in many online learning algorithms (see, e.g., [19, 9, 13]). Here, the projection routine, given a point, outputs its projection onto the convex hull of combinatorial concepts and the decomposition routine, given finds a liner combination of combinatorial concepts given a point of the convex hull. So far, for particular combinatorial concepts, we need to design projection and decomposition subroutines individually.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 260–274, 2012.

In this paper, we investigate a unified and efficient projection and decomposition algorithms for a wide class of combinatorial concepts. The class we consider is the set of vertices (extreme points) of a polyhedron described by a submodular function f. In submodular function literature, the polyhedron is called (submodular) *base polyhedron* and denoted as $B(f)$ (we will give the definition later). That is, we consider the situation where $\mathcal{C}$ is the set of extreme points in $B(f)$. The base polyhedron $B(f)$ is defined using 2^n linear constraints and it is known that there are at most $n!$ vertices [7]. Examples of our problems include experts, k-sets [16], permutahedron [17], spanning trees [2, 13], truncated permutahedron, and k-forest. To the best of our knowledge, the last two problems are new for the online learning literature.

We propose projection and decomposition algorithms for the base polyhedron $B(f)$. The running times of the algorithms are both $O(n^6 + n^5 EO))$, where EO denotes the unit time to evaluate the submodular function. Furthermore, for cardinality-based submodular functions, we derive $O(n^2)$-time projection and decomposition algorithms. Such examples include k-sets and (truncated) permutahedron.

Our projection algorithms are designed for Euclidean distance and unnormalized relative entropy. So, we can combine them with Online Gradient Descent(OGD) [19] or Hedge [6], respectively. Combined with our projection and decomposition algorithms for $B(f)$, their regret bounds become $O(D_{euc}\sqrt{nT})$ and $O(\sqrt{L^* f([n]) \ln n} + f([n]) \ln n)$, respectively, where $D_{euc} = \max_{\boldsymbol{c},\boldsymbol{c}' \in B(f)} \|\boldsymbol{c} - \boldsymbol{c}'\|_2$, and $L^* = \min_{\boldsymbol{c} \in B(f)} \sum_{t=1}^{T} \boldsymbol{c} \cdot \ell_t$.

Our contribution is to provide a unified view and efficient prediction strategies for an online prediction problem with exponentially many candidates by using rich theory of submodular function. Further, our $O(n^2)$-time algorithms for cardinality-based submodular functions are non-trivial for submodular optimization as well.

We discuss the relationship between previous and our results. First, we compare Follow the perturbed leader (FPL, [12]) with our algorithms. FPL uses an algorithm which solves “offline” linear optimization. It is well known that linear optimization over the base polyhedron is tractable and solved in $O(n \log n)$ time [4, 7]. So, the running time of FPL for our problem is $O(n \log n)$ at each trial. On the other hand, the regret of FPL is $O(D_1\sqrt{nT})$, where $D_1 = \max_{\boldsymbol{c},\boldsymbol{c}' \in \mathcal{C}} \|\boldsymbol{c} - \boldsymbol{c}'\|_1$, which is worse than ours.

Next, we consider an algorithm proposed by [11] which converts an offline linear approximate optimization algorithm into the online one. This algorithm has an approximate projection subroutine. But the running time of the projection subroutine is $O(Tn \log n)$, which depends on T.

Component Hedge (CH, [13]) is also an efficient algorithm for predicting among exponentially many combinatorial concepts. CH represents a combinatorial concept as a matrix and solves an entropy minimization problem with linear constraints at each trial. CH has more known applications such as directed spanning trees, paths and so on. However, the class of concepts for which CH can deal with seems incomparable with ours. In an algorithmic sense, our algorithm has advantages for some concepts. For example, for permutahedron and its truncated one, it can be shown that CH requires $O(n^2)$ memory whereas ours uses $O(n)$ memory (see [17] for related discussion).

2 Preliminaries

For any fixed positive integer n, we denote by $[n]$ the set $\{1, \ldots, n\}$. A function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if for any $A, B \subset [n]$, $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$. For simplicity, we assume that $f(\emptyset) = 0$. Given a submodular function f, *the base polyhedron* is defined as

$$B(f) = \left\{ \boldsymbol{x} \in \mathbb{R}^n \mid \sum_{i \in S} x_i \leq f(S), \text{ for any } S \subset [n], \text{and } \sum_{i=1}^{n} x_i = f([n]) \right\}.$$

A point in $B(f)$ is an *extreme point* if it is not represented as a convex combination of other two points in $B(f)$. Let $\mathcal{C}$ be the set of extreme points in $B(f)$. In general, there can be exponentially many extreme points in $B(f)$. In this paper, for any submodular function f, we assume an oracle that returns the value $f(S)$ for any input S.

2.1 Examples

We illustrate some examples of our problems. In particular, the last two problems are new applications which are not previously studied.

Experts Problem. The classical expert problem [6] is an example of our problem. In the expert problem, we are given n experts and the player would like to predict as well as the best expert in hindsight. Here each expert i is represented as n-dimensional unit vector $\boldsymbol{e}_i$ whose i-th component is 1 and other components are 0. Then, the corresponding submodular function f is the constant function $f(S) = 1$, which is submodular.

k-Sets. The problem of k-sets is a generalization of Experts problem, where each combinatorial concept corresponds to a set of k experts among n experts. This problem was first considered by [16]. Each k-set is represented as a sum of k different unit vectors. Then, $\mathcal{C} = \{\boldsymbol{x} \in \{0, 1\}^n \mid \sum_i^n x_i = k\}$. Let $f : 2^{[n]} \to \mathbb{R}$ such that $f(S) = g(|S|)$, where $g(i) = i$, if $i \leq k$ and $g(i) = k$, if $i > k$. This function is submodular since any concave function of $|S|$ is submodular (see, e.g., [7]).

Spanning Trees. Online prediction problems of undirected or directed spanning trees are studied in [2] and [13]. In this paper, we consider undirected spanning trees. Let $G = (V, E)$ be an undirected graph. Let $f : 2^E \to R$ such that $f(A) = |V(A)| - s(A)$, where $V(A)$ is the set of vertices of the subgraph induced by the set A of edges, and $s(A)$ is the number of the connected components of the subgraph [5, 3]. Especially, the base polyhedron is called spanning tree polyhedron and $\mathcal{C} = \{\boldsymbol{x} \in \{0, 1\}^{|E|} \mid \text{the set of edges} \{e \mid x_e = 1\}$ forms a spanning tree of $G\}$.

Permutahedron. Let $\mathcal{C} = \{(i_1, \ldots, i_n) \mid (i_1, \ldots, i_n)$ is a permutation of $\{1, \ldots, n\}\}$. Each permutation corresponds to an element of S_n. The corresponding submodular function is $f(S) = \sum_{i=1}^{|S|}(n+1-i)$. The base polyhedron $B(f)$ is called *Permutahedron* (see, e.g., [18, 7]). This concept class relates to an online scheduling problem of n jobs with a single processor where the sum of flow time of each job is to be minimized [17] . A different representation of permutations and the related problem was also considered by [9].

Truncated Permutahedron. For $k < n$, let $\mathcal{C} = \{(i_1, \ldots, i_n) \mid (i_1, \ldots, i_n)$ is a permutation of $1, 2, \ldots, n-k$,and k $(n-k)$s$\}$. For example, $(2, 2, 2, 1)$ is a member of $\mathcal{C}$ for $n = 4$ and $k = 2$. The corresponding function is $f(S) = (n-k)|S|$ if $|S| \le k$ and $f(S) = (n-k)k + \sum_{j=k+1}^{|S|}(n+1-j)$, otherwise. This concept class is also related to a generalized version of the online scheduling problem [17] where the flow times of the first k jobs are neglected.

k-Forest. Let $\mathcal{C}$ denote the set of k-forests in a graph $G = (V, E)$, where $F \subset E$ is a k-forest if $|F| = k$ and F does not contain a cycle in G. It is known that $\mathcal{C}$ is a bases family of a truncation of a graphic matroid, that is known to be another matroid. The corresponding function is $f(X) = \min\{k, \max\{|F| \mid F \subseteq X$ is a forest$\}\}$.

2.2 Extreme Points of the Base Polyhedron

In this subsection, we will see the correspondence between the permutations of $[n]$ and the extreme points of the base polyhedron $B(f)$.

Given a permutation $\boldsymbol{\sigma} = (i_1, \ldots, i_n)$ of $[n] = \{1, \ldots, n\}$, the greedy algorithm of [4] generates a point $\boldsymbol{c}^{\boldsymbol{\sigma}} \in \mathbb{R}^n$ determined by

$$c_j^{\boldsymbol{\sigma}} = f(\{j' \in [n] : i_{j'} \le i_j\}) - f(\{j' \in [n] : i_{j'} < i_j\}) \text{ for each } j \in [n].$$

Then $\boldsymbol{c}^{\boldsymbol{\sigma}}$ is an extreme point of $B(f)$. We will say that $\boldsymbol{c}^{\boldsymbol{\sigma}}$ is an extreme point of $B(f)$ generated by $\boldsymbol{\sigma}$. Conversely, for each extreme point $\boldsymbol{c}$ of $B(f)$, there is a permutation that generates $\boldsymbol{c}$.

2.3 Bregman Divergence

Let $\Phi : \Gamma \to \mathbb{R}$ be a strictly convex function defined on a closed convex set $\Gamma \subseteq \mathbb{R}^n$. The Bregman divergence Δ_Φ with respect to Φ is defined as $\Delta_\Phi(\boldsymbol{p}, \boldsymbol{q}) = \Phi(\boldsymbol{p}) - \Phi(\boldsymbol{q}) - \nabla\Phi(\boldsymbol{q}) \cdot (\boldsymbol{p} - \boldsymbol{q})$. The function Φ is *separable* if there exists functions $\phi_i : \Gamma_i \to \mathbb{R}$ for $i = 1, 2., \ldots, n$ such that $\Gamma = \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$ and for any $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \Gamma$, $\Phi(\boldsymbol{x}) = \sum_{i=1}^n \phi_i(x_i)$. In particular, if all ϕ_i's are the same, then the function Φ is said to be *uniformly separable*. In this paper, we will sometimes consider two particular uniformly separable convex functions, the *2-norm function*: $\Phi_{\mathrm{EUC}}(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{x}\|_2^2$ defined on $\mathbb{R}^n$, and the *unnormalized negative entropy function*: $\Phi_{\mathrm{URE}}(\boldsymbol{x}) = \sum_{i=1}^n x_i \ln x_i - \sum_{i=1}^n x_i$ defined on $\mathbb{R}_{>0}^n$.

It is well known that these functions define the Euclidean distance and the unnormalized relative entropy, respectively. That is, $\Delta_{\mathrm{EUC}}(\boldsymbol{x}, \boldsymbol{z}) \overset{\mathrm{def}}{=} \Delta_{\Phi_{\mathrm{EUC}}}(\boldsymbol{x}, \boldsymbol{z}) = \frac{1}{2}\sum_{i=1}^{n}(x_i - z_i)^2$, and $\Delta_{\mathrm{URE}}(\boldsymbol{x}, \boldsymbol{z}) \overset{\mathrm{def}}{=} \Delta_{\Phi_{\mathrm{URE}}}(\boldsymbol{x}, \boldsymbol{z}) = \sum_{i=1}^{n} x_i \ln \frac{x_i}{z_i} + \sum_{i=1}^{n} z_i - \sum_{i=1}^{n} x_i$.

3 Algorithm

In this section, we propose an algorithm that predicts extreme points of the base polyhedron $B(f)$ and prove its regret bounds.

3.1 Main Structure

The main structure of the algorithm we use is shown in Algorithm 1. The algorithm is Regularized Follow the Leader (RFTL) [8], which is a generalization of Hedge or Online Gradient Decent(OGD, [19]) using Bregman divergence, combined with our subroutines Projection$_\Phi$ and Decomposition.

At each trial t, our version of RFTL runs Decomposition and get $\boldsymbol{c}_t \in \mathcal{C}$ randomly so that $\mathbf{E}[\boldsymbol{c}_t] = \boldsymbol{x}_t$. Then, RFTL updates $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t+\frac{1}{2}}$. Finally, RFTL computes the projection $\boldsymbol{x}_{t+1}$ of $\boldsymbol{x}_{t+\frac{1}{2}}$ onto the base polyhedron $B(f)$ using the procedure Projection$_\Phi$. Using RFTL itself is standard, but we need to design efficient procedures for projection and decomposition.

Algorithm 1. RFTL with Projection and Decomposition

```
1. Let $\boldsymbol{x}_1$ be any point in $B(f)$.
2. For $t = 1, \ldots, T$
   (a) Run Decomposition($\boldsymbol{x}_t$) and get $\boldsymbol{c}_t \in \mathcal{C}$ randomly so that $\mathbf{E}[\boldsymbol{c}_t] = \boldsymbol{x}_t$.
   (b) Predict $\boldsymbol{c}_t$ and incur a loss $\boldsymbol{c}_t \cdot \boldsymbol{\ell}_t$.
   (c) Update $\boldsymbol{x}_{t+\frac{1}{2}}$ as $\boldsymbol{x}_{t+\frac{1}{2}} = \nabla\Phi^{-1}(\nabla\Phi(\boldsymbol{x}_t) - \eta\boldsymbol{\ell}_t)$.
   (d) Run Projection$_\Phi$($\boldsymbol{x}_{t+\frac{1}{2}}$) and get $\boldsymbol{x}_{t+1}$, the projection of $\boldsymbol{x}_{t+\frac{1}{2}}$ onto the base
       polyhedron $B(f)$. That is, $\boldsymbol{x}_{t+1} = \arg\inf_{\boldsymbol{x}\in B(f)} \Delta_\Phi(\boldsymbol{x}, \boldsymbol{x}_{t+\frac{1}{2}})$.
```

The following theorem is known.

Theorem 1 ([6, 19, 8]). *Let $\mathcal{C}$ be the set of extreme points in $B(f)$.*

1. *For $\Phi = \Phi_{EUC}$, the expected regret of RFTL is $O(D_{euc}\sqrt{nT})$ for some η, where $D_{euc} = \max_{\boldsymbol{c},\boldsymbol{c}'\in\mathcal{C}} \|\boldsymbol{c} - \boldsymbol{c}'\|_2$.*
2. *For $\Phi = \Phi_{URE}$, the expected regret of RFTL is $O(\sqrt{L^* D_{ure}} + D_{ure})$ for some η and $\boldsymbol{x}_1 \in B(f)$, where $D_{ure} = \max_{\boldsymbol{c}\in\mathcal{C}} \Delta_{\Phi_{URE}}(\boldsymbol{c}, \boldsymbol{x}_1)$ and $L^* = \min_{\boldsymbol{c}\in\mathcal{C}} \sum_{t=1}^{T} \boldsymbol{c} \cdot \boldsymbol{\ell}_t$.*

For particular combinatorial concepts, we summarize their regret bounds in Table 1.

Table 1. The regrets of combinatorial concepts obtained using our projection and decomposition algorithms

problem	Hedge	OGD
Experts	$O(\sqrt{L^* \ln n})$	$O(\sqrt{nT})$
k-sets	$O(\sqrt{L^* k \ln(n/k)} + k \ln(n/k))$	$O(\sqrt{knT})$
Spanning Trees	$O(\sqrt{L^* n \ln n} + n \ln n)$	$O(n\sqrt{T})$
Permutahedron	$O(n\sqrt{L^* \ln n} + n^2 \ln n)$	$O(n^2\sqrt{T})$
Truncated Perm.	$O(\sqrt{L^*(n^2-k^2)\ln n} + (n^2-k^2)\ln n)$	$O((n-k)\sqrt{n(n+k)T})$
k-forest	$O(\sqrt{L^* k \ln(n/k)} + k \ln(n/k))$	$O(\sqrt{knT})$

To complete our analysis, we specify the procedures Projection$_\Phi$ for separable strictly convex function Φ and Decomposition, respectively, in the following subsections. We will see that both of the two procedures are no harder than the submodular function minimization problem. For a submodular function $f : 2^{[n]} \to \mathbb{R}$ with $f(\emptyset) = 0$, the submodular function minimization (SFM) is a problem of finding a subset $S \subseteq [n]$ with $f(S)$ minimum. Many combinatorial SFM algorithms are known (see [10]), and the fastest known strongly polynomial algorithm of [15] runs in $O(n^6 + n^5 EO)$ time, where EO is the unit time to evaluate the value of the submodular function. We will show that both of the procedures Projection$_\Phi$ and Decomposition can be implemented to run in $O(n^6 + n^5 EO)$ time.

3.2 Projection

For any given point $\boldsymbol{z} \in \mathbb{R}^n$, the procedure Projection$_\Phi$ in Algorithm 1 computes the projection of $\boldsymbol{z}$ onto the base polyhedron $B(f)$. We propose an efficient construction of this procedure. Formally, the projection problem is stated as follows:

$$\text{Projection}_\Phi(\boldsymbol{z}) = \arg \inf_{\boldsymbol{x} \in B(f)} \Delta_\Phi(\boldsymbol{x}, \boldsymbol{z})$$

$$\text{sub. to: } \sum_{j \in S} x_j \leq f(S),\ \forall S \subset [n],\ \text{and } \sum_{j=1}^{n} x_j = f([n]), \quad (1)$$

where $\Phi(\boldsymbol{x})$ is separable. This convex optimization problem with exponentially many constraints can be solved efficiently using the parametric submodular algorithm of [14], which is a parametric extension of the SFM algorithm of [15].

Theorem 2 ([14]). *There is an algorithm that solves problem (1) for separable strictly convex functions Φ in time $O(n^6 + n^5 EO)$.*

3.3 Decomposition

For any given point $\boldsymbol{x}$ in the base polyhedron $B(f) \subseteq \mathbb{R}^n$, the procedure Decomposition in Algorithm 1 finds extreme points $\boldsymbol{c}^{\boldsymbol{\sigma}^1}, \ldots, \boldsymbol{c}^{\boldsymbol{\sigma}^K}$ in $B(f)$ and $\lambda_1, \ldots, \lambda_K \in \mathbb{R}_{>0}$ such that $\sum_{i=1}^{K} \lambda_i \boldsymbol{c}^{\boldsymbol{\sigma}^i} = \boldsymbol{x}$ and $\lambda_1 + \cdots + \lambda_K = 1$, where each $\boldsymbol{c}^{\boldsymbol{\sigma}^i}$ is an extreme point of $B(f)$ generated by a permutation $\boldsymbol{\sigma}^i$ of $[n]$ via the greedy algorithm of [4]. In other words, this procedure represents $\boldsymbol{x}$ as a convex combination of extreme points of $B(f)$. Carathéodory's Theorem guarantees that $\boldsymbol{x} \in B(f)$ can be represented as a convex combination of at most n extreme points of $B(f)$.

To describe the procedure Decomposition, let us briefly review a common framework of algorithms for SFM. For a submodular function $f' : 2^{[n]} \to \mathbb{R}$ with $f'(\emptyset) = 0$, the result of [4] implies

$$\min_S \{f'(S) : S \subseteq [n]\} = \max_{\boldsymbol{z}} \{ \sum_{j=1}^n \min\{0, z_j\} : \boldsymbol{z} \in B(f')\}. \tag{2}$$

In many combinatorial SFM algorithms, including Orlin's algorithm ([15]), we finally obtain a minimizer $S^* \subseteq [n]$ and a maximizer $\boldsymbol{z}^* \in B(f')$ of (2). Moreover, we obtain $\boldsymbol{z}^* \in B(f')$ as a convex combination of at most n extreme points of $B(f')$. By the use of this fact, we can give an efficient construction of the procedure Decomposition.

For a given point $\boldsymbol{x} \in B(f)$, the function $f_{\boldsymbol{x}} : 2^{[n]} \to \mathbb{R}$ defined by $f_{\boldsymbol{x}}(S) = f(S) - \sum_{j \in S} x_j$ $(S \subseteq [n])$ is submodular and satisfies $f_{\boldsymbol{x}}(\emptyset) = 0$. For each permutation $\boldsymbol{\sigma}$ of $[n]$, let $\boldsymbol{c}^{\boldsymbol{\sigma}}$ be extreme points in $B(f)$ generated by $\boldsymbol{\sigma}$, and let $\boldsymbol{c}_{\boldsymbol{x}}^{\boldsymbol{\sigma}}$ be extreme points in $B(f_{\boldsymbol{x}})$ generated by $\boldsymbol{\sigma}$. Then it holds that $\boldsymbol{c}_{\boldsymbol{x}}^{\boldsymbol{\sigma}} = \boldsymbol{c}^{\boldsymbol{\sigma}} - \boldsymbol{x}$. In view of the definition of the base polyhedron, we have that $\min_{S \subseteq [n]} f_{\boldsymbol{x}}(S) = 0$ and the n-dimensional zero vector $\mathbf{0}_n$ is in $B(f_{\boldsymbol{x}})$. Therefore, $\boldsymbol{z} = \mathbf{0}_n$ is the unique optimal solution to the right hand side of (2) with $f' = f_{\boldsymbol{x}}$.

Now we describe the procedure Decomposition. Initially, we apply some combinatorial SFM algorithm, e. g. Orlin's algorithm ([15]), to the submodular function $f_{\boldsymbol{x}}$. Then we obtain permutations $\boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^K$ of $[n]$ and $\lambda_1, \ldots, \lambda_K \in \mathbb{R}_{>0}$ such that $\sum_{i=1}^K \lambda_i \boldsymbol{c}_{\boldsymbol{x}}^{\boldsymbol{\sigma}^i} = \mathbf{0}$, $\lambda_1 + \cdots + \lambda_K = 1$, and $K \leq n$. As for the function f, these permutations $\boldsymbol{\sigma}^1, \ldots, \boldsymbol{\sigma}^K$ and positive coefficients $\lambda_1, \ldots, \lambda_K$ generate another point $\sum_{i=1}^K \lambda_i \boldsymbol{c}^{\boldsymbol{\sigma}^i}$. For this point, we obtain

$$\sum_{i=1}^K \lambda_i \boldsymbol{c}^{\boldsymbol{\sigma}^i} = \sum_{i=1}^K \lambda_i (\boldsymbol{c}_{\boldsymbol{x}}^{\boldsymbol{\sigma}^i} + \boldsymbol{x}) = \sum_{i=1}^K \lambda_i \boldsymbol{c}_{\boldsymbol{x}}^{\boldsymbol{\sigma}^i} + \sum_{i=1}^K \lambda_i \boldsymbol{x} = \boldsymbol{x}.$$

Thus we have a required representation of $\boldsymbol{x}$. This gives the following.

Theorem 3. *For any $\boldsymbol{x} \in B(f)$, there is an algorithm that gives a convex combination representation of $\boldsymbol{x}$ using at most n extreme points of $B(f)$ in $O(n^6 + n^5 EO)$ time.*

4 Algorithm for Cardinality-Based Submodular Functions

In this section, we propose more efficient projection and decomposition algorithms when the underlying submodular function f is cardinality-based, i.e., $f(S) = g(|S|)$ for some $g : \mathbb{N} \to \mathbb{R}$. For projection, however, we only consider the Euclidean distance and the unnormalized relative entropy, rather than any Bregman divergence ∇_{Φ} for a separable function Φ as in the previous section.

A cardinality-based submodular function f has the following nice property: For any point $\boldsymbol{x} \in B(f)$ and any $i, j \in [n]$, the vector $\boldsymbol{x}'$ obtained by exchanging x_i and x_j in $\boldsymbol{x}$ is also contained in $B(f)$. A submodular function having this property is said to be *exchangeable.*

The following lemma says that for any exchangeable submodular function f, the projection onto $B(f)$ preserves the order of indices of vector with respect to the inequality relation.

Lemma 1. *Let $\boldsymbol{x}^*$ be the projection of $\boldsymbol{z}$ in (1) under the Bregman divergence ∇_{Φ} for a strictly convex and uniformly separable function Φ. Assume that the submodular function f is exchangeable and $z_1 \geq \cdots \geq z_n$. Then, it holds that $x_1^* \geq x_2^* \geq \cdots \geq x_n^*$.*

Proof. Suppose on the contrary that $x_i^* < x_j^*$ for some $i < j$. Let $\widehat{\boldsymbol{x}}$ be the point obtained by exchanging x_i^* and x_j^* in $\boldsymbol{x}^*$. Then, by definition, we have $\widehat{\boldsymbol{x}} \in B(f)$. Furthermore, observe that

$$\begin{aligned}
\Delta_{\Phi}(\boldsymbol{x}^*, \boldsymbol{z}) - \Delta_{\Phi}(\hat{\boldsymbol{x}}, \boldsymbol{z}) =& \Phi(\boldsymbol{x}^*) - \Phi(\hat{\boldsymbol{x}}) - \nabla\Phi(\boldsymbol{z}) \cdot (\boldsymbol{x}^* - \hat{\boldsymbol{x}}) \\
=& \phi(x_i^*) + \phi(x_j^*) - \phi(x_i^*) - \phi(x_j^*) \\
& - (\phi'(z_i)(x_i^* - x_j^*) - \phi'(z_j)(x_j^* - x_i^*)) \\
=& (x_j^* - x_i^*) \cdot (\phi'(z_i) - \phi'(z_j)) \geq 0,
\end{aligned}$$

which contradicts the assumption that $\boldsymbol{x}^*$ is the projection. □

In the following, we assume that $z_1 \geq \cdots \geq z_n$ without loss of generality (this can be achieved by sorting). Lemma 1 implies that for any $S \subseteq [n]$, $\sum_{i \in S} x_i^* \leq \sum_{j=1}^{|S|} x_j^*$, which means that, if the right hand side is bounded by $f(S) = g(|S|)$, the left hand side is also bounded by $g(|S|)$. Therefore, the projection problem (1) is equivalent to the following problem with only n constraints:

$$\begin{aligned}
& \min_{\boldsymbol{x}} \Delta_{\Phi}(\boldsymbol{x}, \boldsymbol{z}) \\
\text{sub.to: } & \sum_{i=1}^{j} x_i \leq g(j), \text{ (for } j = 1, \ldots, n-1), \text{ and } \sum_{i=1}^{n} x_i = g(n).
\end{aligned} \tag{3}$$

Now we propose an efficient implementation of Projection$_{\Phi}$ that solves the problem (3).

4.1 Projection under Euclidean Distance

First we give an algorithm which computes Projection$_{\Phi_{\mathrm{EUC}}}$ under Euclidean distance. We show the algorithm in Algorithm 2. Then we prove the following.

Algorithm 2. Projection under Euclidean distance

```
Input: $\boldsymbol{z} \in \mathbb{R}^n$ satisfying that $z_1 \geq z_2 \geq \cdots \geq z_n$.
Output: projection $\boldsymbol{x}$ of $\boldsymbol{z}$ onto $B(f)$.

1. Let $i_0 = 0$.
2. For $t = 1, \ldots,$
   (a) Let $C^t(i) = \frac{g(i) - g(i_{t-1}) - \sum_{j=i_{t-1}+1}^{i} z_j}{i - i_{t-1}}$, for $i = 1, \ldots, n$
       and $i_t = \arg\min_{i: i_{t-1}+1 \leq i \leq n} C^t(i)$,
       if there are multiple minimizers, choose the largest one as $i_t$.
   (b) Set $x_i = z_i + C^t(i_t)$, for $i_{t-1} + 1 \leq i \leq i_t$.
   (c) If $i_t = n$, then break.
3. Output $\boldsymbol{x}$.
```

Theorem 4. *(i) Given $\boldsymbol{z}$, Algorithm 2 outputs the projection of $\boldsymbol{x}$ onto the base polyhedron $B(f)$. (ii) The time complexity of Algorithm 2 is $O(n^2)$.*

Proof. By KKT condition(see, e.g., [1]), $\boldsymbol{x}^*$ is the solution of the problem (3) if and only if there exists $\alpha_1, \ldots, \alpha_{n-1}$ and η such that

$$x_i^* = z_i - \sum_{j=1}^{i} \alpha_j - \eta, \ \text{(for } i = 1, \ldots, n-1), \ \text{and } x_n^* = z_n - \eta,$$

$$\sum_{i=1}^{n} x_i^* = g(n),$$

$$\alpha_i (\sum_{j=1}^{i} x_j^* - g(i)) = 0, \ \alpha_i \geq 0, \ \sum_{j=1}^{i} x_j^* \leq g(i) \ \text{(for } i = 1, \ldots, n-1). \tag{4}$$

Now we show that there indeed exists $\alpha_1, \ldots, \alpha_{n-1}$ such that the output $\boldsymbol{x}$ of Projection$_{\Phi_{EUC}}(\boldsymbol{z})$ satisfies the optimality conditions (4), which suffices to prove the first statement. To do so, first we show that $C^{t-1}(i_{t-1}) \leq C^t(i_t)$ for each iteration t. By the definition of $C^{t-1}(i_{t-1})$, we have $C^{t-1}(i_{t-1}) \leq C^{t-1}(i_t)$. Observe that

$$\begin{aligned} C^{t-1}(i_t) &= \frac{g(i_t) - g(i_{t-2}) - \sum_{j=i_{t-2}+1}^{i_t} z_j}{i_t - i_{t-2}} \\ &= \frac{g(i_t) - g(i_{t-1}) - \sum_{j=i_{t-1}+1}^{i_t} z_j + g(i_{t-1}) - g(i_{t-2}) - \sum_{j=i_{t-2}+1}^{i_{t-1}} z_j}{(i_t - i_{t-1}) + (i_{t-1} - i_{t-2})} \\ &= \frac{(i_t - i_{t-1})(C^t(i_t)) + (i_{t-1} - i_{t-2})(C^{t-1}(i_{t-1}))}{(i_t - i_{t-1}) + (i_{t-1} - i_{t-2})}. \end{aligned}$$

Since $C^{t-1}(i_{t-1}) \leq C^{t-1}(i_t)$,

$$\frac{(i_t - i_{t-1})(C^{t-1}(i_{t-1}))}{(i_t - i_{t-1}) + (i_{t-1} - i_{t-2})} \leq \frac{(i_t - i_{t-1})(C^t(i_t))}{(i_t - i_{t-1}) + (i_{t-1} - i_{t-2})}.$$

By simplifying this, we get $C^{t-1}(i_{t-1}) \leq C^t(i_t)$, as desired.

Then we determine each α_{i_t} so that $-\alpha_{i_t} + C^{t+1}(i_{t+1}) = C^t(i_t)$, i.e., $\alpha_{i_t} = C^{t+1}(i_{t+1}) - C^t(i_t)$ and fix η to be $C^T(n)$, where T satisfies $i_T = n$. Note that since $C^t(i_t) \leq C^{t+1}(i_{t+1})$, each α_{i_t} is strictly positive. For other $i \notin \{i_1, \ldots, i_T\}$, we set $\alpha_i = 0$. Then, each x_{i_t} can be expressed as

$$x_{i_t} = z_i + C^t(i_t) = z_i - \alpha_{i_t} - \alpha_{i_{t+1}} - \cdots - \alpha_{i_T} - \eta = z_i - \alpha_{i_t} - \alpha_{i_t+1} - \cdots - \alpha_{i_{n-1}} - \eta.$$

Similarity, for other i such that $i_{t-1} < i < i_t$, we have

$$x_i = z_i + C^t(i_t) = z_i - \alpha_{i_t} - \alpha_{i_t+1} - \cdots - \alpha_{n-1} - \eta = z_i - \alpha_i - \alpha_{i+1} - \cdots - \alpha_{n-1} - \eta.$$

To check if the specified α_is and η satisfies the optimality conditions (4), observe that (i) for each i_t,

$$\sum_{j=1}^{i_t} x_j = \sum_{j=1}^{i_{t-1}} x_j + \sum_{j=i_{t-1}+1}^{i_t} (z_j + C^t(i_t)) = g(i_{t-1}) + (g(i_t) - g(i_{t-1})) = g(i_t)$$

and $\alpha_{i_t} > 0$, and (ii) for each i such that $i_{t-1} < i < i_t$,

$$\begin{aligned} \sum_{j=1}^{i} x_j = \sum_{j=1}^{i_{t-1}} x_j + \sum_{j=i_{t-1}+1}^{i} (z_j + C^t(i_t)) &\leq \sum_{j=1}^{i_{t-1}} x_j + \sum_{j=i_{t-1}+1}^{i} (z_j + C^t(i)) \\ &= g(i_{t-1}) + (g(i) - g(i_{t-1})) = g(i) \end{aligned}$$

and $\alpha_i = 0$.

Finally, the algorithm terminates in time $O(n^2)$ since the number of iteration is at most n and each iteration takes $O(n)$ time, which proves the second statement of the lemma. □

4.2 Projection under Unnormalized Relative Entropy

Next we propose an algorithm for Projection$_{\Phi_{URE}}$. We construct the projection algorithm by generalizing the one used for permutahedron [17] . Note that the algorithm is also a generalization of the capping algorithm in [16]. The algorithm shown in Algorithm 3 outputs the solution which satisfies the optimality conditions, and following theorem holds.

Theorem 5. *(i) Given $\boldsymbol{z}$, the Algorithm 3 outputs the projection of $\boldsymbol{x}$ onto the base polyhedron $B(f)$. (ii) The time complexity of Algorithm 3 is $O(n^2)$.*

The proof is also a generalization of the proof in [17] and omitted due to the space constraints.

Algorithm 3. Projection under unnormalized relative entropy

```
Input: $\boldsymbol{z} \in \mathbb{R}^n$ satisfying that $z_1 \geq z_2 \geq \cdots \geq z_n$.
Output: projection $\boldsymbol{x}$ of $\boldsymbol{z}$ onto $B(f)$.

1. Let $i_0 = 0$.
2. For $t = 1, \ldots,$
   (a) Let $C^t(i) = \frac{g(i)-g(i_{t-1})}{\sum_{j=i_{t-1}+1}^{i} z_j}$, for $i = 1, \ldots, n$
       and $i_t = \arg\min_{i:i_{t-1}+1 \leq i \leq n} C^t(i)$,
       if there are multiple minimizers, choose the largest one as $i_t$.
   (b) Set $x_i = z_i C^t(i_t)$, for $i_{t-1} + 1 \leq i \leq i_t$.
   (c) If $i_t = n$, then break.
3. Output $\boldsymbol{x}$.
```

4.3 Decomposition

In this subsection, we describe how to represent a point $\boldsymbol{x} \in B(f)$ by a convex combination of extreme points of $B(f)$. More precisely, we are concerned with the following randomized rounding problem; given a point $\boldsymbol{x} \in B(f)$, output an extreme point X of $B(f)$ with a probability such that $E[X] \overset{\text{def}}{=} \sum_{j=1}^{k} \Pr\left[X = \boldsymbol{c}^j\right] \cdot \boldsymbol{c}^j = \boldsymbol{x}$ for an appropriate $k > 0$.

As a preliminary step, we explain the following Propositions 6, 7, and 8, which are well-known facts (see e.g., [7]). Let $a \in \mathbb{R}_{>0}$ be a constant satisfying $a > g(n-1) - g(n)$, and we define $\tilde{f} \colon 2^{[n]} \to \mathbb{R}$ by $\tilde{f}(S) \overset{\text{def}}{=} f(S) + a|S|$ for any $S \subseteq [n]$. Notice that $\tilde{f}$ is clearly a cardinality based function; let $\tilde{g}(z) \overset{\text{def}}{=} g(z) + a \cdot z$ then $\tilde{f}(S) = \tilde{g}(|S|)$ holds.

Proposition 6. *The function $\tilde{f}$ is cardinality based* submodular *and* monotone increasing*, i.e., $\tilde{g}(i) < \tilde{g}(i+1)$ for each $i \in [n-1]$.*

Note that $\tilde{f}(\emptyset) = 0$, and $\tilde{f}(S) > 0$ hold for any S ($\emptyset \subset S \subseteq [n]$).

Proposition 7. *A point $\boldsymbol{x}$ is in $B(f)$ if and only if $\tilde{\boldsymbol{x}} \overset{\text{def}}{=} \boldsymbol{x} + a\mathbf{1}$ is in $B(\tilde{f})$. A point $\boldsymbol{c}$ is an extreme point of $B(f)$ if and only if $\tilde{\boldsymbol{c}} \overset{\text{def}}{=} \boldsymbol{c} + a\mathbf{1}$ is an extreme point of $B(\tilde{f})$.*

Proposition 8. *Suppose $\boldsymbol{x} \in B(f)$ satisfies $\boldsymbol{x} = \sum_{j=1}^{k} \lambda_j \boldsymbol{c}^j$ for $\lambda_j > 0$ ($j \in [k]$) satisfying $\sum_{j=1}^{k} \lambda_j = 1$ and $\boldsymbol{c}^j \in B(f)$ ($j \in [k]$). Then, $\tilde{\boldsymbol{x}} \overset{\text{def}}{=} \boldsymbol{x} + a\mathbf{1} \in B(\tilde{f})$ satisfies $\tilde{\boldsymbol{x}} = \sum_{j=1}^{k} \lambda_j \tilde{\boldsymbol{c}}^j$ where $\tilde{\boldsymbol{c}}^j \overset{\text{def}}{=} \boldsymbol{c}^j + a\mathbf{1} \in B(\tilde{f})$.*

Now, let $\tilde{f} \colon 2^{[n]} \to \mathbb{R}_{\geq 0}$ be a cardinality based submodular function which is *monotone increasing*, then we consider the randomized rounding problem; given a point $\tilde{\boldsymbol{x}} \in B(\tilde{f})$, output an extreme point X of $B(\tilde{f})$ with a probability such that $E[X] \overset{\text{def}}{=} \sum_{j=1}^{k} \Pr\left[X = \tilde{\boldsymbol{c}}^j\right] \cdot \tilde{\boldsymbol{c}}^j = \tilde{\boldsymbol{x}}$ for an appropriate $k > 0$. By Proposition 8, it is easily transformed into the case from a general cardinality based submodular function. Without loss of generality, we may assume that $\tilde{x}_1 \geq \cdots \geq \tilde{x}_n$

in the following. We remark that our randomized rounding algorithm is a generalization of [17] for the *permutahedron*, in a sense.

To begin with, we define special points in $B(\tilde{f})$, which we call *partially averaged points*. Suppose $\tilde{\boldsymbol{q}} \in B(\tilde{f})$ satisfies that $\tilde{q}_1 \geq \tilde{q}_2 \geq \cdots \geq \tilde{q}_n$, then, $\tilde{\boldsymbol{q}}$ is a partially averaged point if $\sum_{j=1}^{i} \tilde{q}_j = \tilde{g}(i)$ holds for each $i \in [n]$ satisfying $\tilde{q}_i > \tilde{q}_{i+1}$. Notice that if $\tilde{q}_i > \tilde{q}_{i+1} = \cdots = \tilde{q}_j > \tilde{q}_{j+1}$ hold for $i, j \in [n]$ then $q_{i+1} = \cdots = q_j = (\tilde{g}(j) - \tilde{g}(i))/(j-i)$. This means that the partially averaged point is uniquely determined only by a sequence of equalities(=)/inequalities(>). We simply say "a partially averaged point of $\tilde{\boldsymbol{x}}$" ($\tilde{\boldsymbol{x}} \in B(\tilde{f})$) as a partially averaged point determined by a sequence of equalities/inequalities derived from $\tilde{x}_1 \geq \tilde{x}_2 \geq \cdots \geq \tilde{x}_n$ of $\tilde{\boldsymbol{x}}$.

Proposition 9. *Suppose $\tilde{\boldsymbol{q}} \in B(\tilde{f})$ is a partially averaged point satisfying $\tilde{q}_1 \geq \tilde{q}_2 \geq \cdots \geq \tilde{q}_n$. Let $\Pi \overset{\text{def}}{=} \{\boldsymbol{\sigma} \in \mathrm{Sym}(n) \mid \tilde{q}_{\sigma(1)} \geq \tilde{q}_{\sigma(2)} \geq \cdots \geq \tilde{q}_{\sigma(n)}\}$, and let $\tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}} = (\tilde{c}_1^{\boldsymbol{\sigma}}, \ldots, \tilde{c}_n^{\boldsymbol{\sigma}})$ for $\boldsymbol{\sigma} \in \Pi$ denote the extreme point defined by hyperplanes $\sum_{j=1}^{i} \tilde{c}_{\sigma(j)}^{\boldsymbol{\sigma}} = \tilde{g}(i)$ for all $i \in [n]$. Note that $\boldsymbol{\sigma} \neq \boldsymbol{\sigma}'$ does not imply $\tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}} \neq \tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}'}$ in general. Then, $\tilde{\boldsymbol{q}} = \frac{1}{|\Pi|} \sum_{\boldsymbol{\sigma} \in \Pi} \tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}}$.*

Proof. Suppose $i \in [n-1]$ satisfies $\tilde{q}_i > \tilde{q}_{i+1}$. Since any $\boldsymbol{\sigma} \in \Pi$ satisfies $\tilde{q}_{\sigma(1)} \geq \tilde{q}_{\sigma(2)} \geq \cdots \geq \tilde{q}_{\sigma(n)}$, we see that $\{\sigma(1), \ldots, \sigma(i)\} = [i]$ holds for any $\boldsymbol{\sigma} \in \Pi$. This implies that $\sum_{j=1}^{i} \tilde{c}_j^{\boldsymbol{\sigma}} = \sum_{j=1}^{i} \tilde{c}_{\sigma(j)}^{\boldsymbol{\sigma}} = \tilde{g}(i)$. Since $\tilde{\boldsymbol{q}}$ is a partially averaged point, remember that $\sum_{j=1}^{i} \tilde{q}_j = \tilde{g}(i)$ holds, too.

Next, suppose $\tilde{q}_i > \tilde{q}_{i+1} = \cdots = \tilde{q}_j > \tilde{q}_{j+1}$ hold for $i, j \in [n]$. From the above arguments, we see that $\sum_{k=i+1}^{j} \tilde{c}_k^{\boldsymbol{\sigma}} = \tilde{g}(j) - \tilde{g}(i)$ holds for any $\boldsymbol{\sigma} \in \Pi$. For an arbitrary $\boldsymbol{\sigma} \in \Pi$, let $\boldsymbol{\sigma}' \in \mathrm{Sym}(n)$ satisfy $\sigma'(k) = \sigma(k)$ for each k ($k \leq i$ or $k > j$), then $\boldsymbol{\sigma}'$ is also in Π. Thus, let $\tilde{\boldsymbol{r}} \overset{\text{def}}{=} \frac{1}{|\Pi|} \sum_{\boldsymbol{\sigma} \in \Pi} \tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}}$ for convenience, then we see that $\tilde{r}_{i+1} = \cdots = \tilde{r}_j = (\tilde{g}(j) - \tilde{g}(i))/(j-i)$ hold. Since $\tilde{\boldsymbol{q}}$ is a partially averaged point, remember that $\tilde{q}_{i+1} = \cdots = \tilde{q}_j = (\tilde{g}(j) - \tilde{g}(i))/(j-i)$ hold, too. □

Proposition 9 and its proof immediately suggest an algorithm for randomized rounding of a partially averaged point; generate $\boldsymbol{\sigma} \in \Pi$ uniformly at random, and output $\tilde{\boldsymbol{c}}^{\boldsymbol{\sigma}}$. It's running time is $O(n)$, clearly.

Now, we explain our Algorithm 4, which provides a convex combination of partially average points representing $\tilde{\boldsymbol{x}} \in B(\tilde{f})$, i.e., given $\tilde{\boldsymbol{x}} \in B(\tilde{f})$, find partially average points $\tilde{\boldsymbol{q}}^1, \ldots, \tilde{\boldsymbol{q}}^K$ and $\lambda_1, \ldots, \lambda_K \in \mathbb{R}_{>0}$ such that $\sum_{i=1}^{K} \lambda_i \tilde{\boldsymbol{q}}^i = \tilde{\boldsymbol{x}}$ and $\sum_{i=1}^{K} \lambda_i = 1$. Once we obtain such a convex combination, it is clear to obtain an algorithm for randomized rounding into partially average points. Combining the above arguments concerning Proposition 9, we obtain a desired algorithm. We will prove the following lemma on Algorithm 4.

Theorem 10. *Algorithm 4 provides a convex combination of at most n partially averaged points representing an arbitrarily given $\tilde{\boldsymbol{x}} \in B(\tilde{f})$. Its running time is $O(n^2)$.*

To show Theorem 10, we show the following lemmas.

Algorithm 4. Decomposition by partially average points

```
Input: $\tilde{\boldsymbol{x}} \in B(\tilde{f})$ satisfying that $\tilde{x}_1 \geq \tilde{x}_2 \geq \cdots \geq \tilde{x}_n$.
Output: Partially average points $\tilde{\boldsymbol{q}}^1, \ldots, \tilde{\boldsymbol{q}}^K$ and $\lambda_1, \ldots, \lambda_K \in \mathbb{R}_{>0}$ s.t. $\sum_{i=1}^K \lambda_i \tilde{\boldsymbol{q}}^i = \tilde{\boldsymbol{x}}$, $\sum_{i=1}^K \lambda_i = 1$.

1. Let $\tilde{\boldsymbol{x}}^1 = \tilde{\boldsymbol{x}}$ and $\lambda = 1$.
2. For $t = 1, \ldots,$
   (a) Find a partially averaged point $\tilde{\boldsymbol{q}}^t$ for $\tilde{\boldsymbol{x}}^t$.
   (b) Let $\lambda_t = \min\left\{\lambda, \min_{i \in [n-1]} \left\{ \frac{\tilde{x}_i^t - \tilde{x}_{i+1}^t}{\tilde{q}_i^t - \tilde{q}_{i+1}^t} \mid \tilde{q}_i^t \neq \tilde{q}_{i+1}^t \right\}\right\}$.
   (c) Let $\tilde{\boldsymbol{x}}^{t+1} = \tilde{\boldsymbol{x}}^t - \lambda_t \tilde{\boldsymbol{q}}^t$ and let $\lambda = \lambda - \lambda_t$.
   (d) If $\lambda = 1$ then let $K = t$ and break.
3. Output $\tilde{\boldsymbol{q}}^1, \ldots \tilde{\boldsymbol{q}}^K$ and $\lambda_1, \ldots, \lambda_K$.
```

Lemma 2. *At any iteration t in Algorithm 4, $\tilde{\boldsymbol{x}}^t$ satisfies that $\tilde{x}_i^t \geq \tilde{x}_{i+1}^t$ for any $i \in [n-1]$.*

Proof. We give an inductive proof with respect to t. In case of $t = 1$, it is clear. In case of $t > 1$, we assume $\tilde{x}_i^{t-1} \geq \tilde{x}_{i+1}^{t-1}$ holds for any $i \in [n-1]$. If $\tilde{x}_i^{t-1} = \tilde{x}_{i+1}^{t-1}$, then $\tilde{q}_i^{t-1} = \tilde{q}_{i+1}^{t-1}$ holds, from the definition of $\tilde{\boldsymbol{q}}^{t-1}$. Thus

$$\tilde{x}_i^t = \tilde{x}_i^{t-1} - \lambda_{t-1}\tilde{q}_i^{t-1} = \tilde{x}_{i+1}^{t-1} - \lambda_{t-1}\tilde{q}_{i+1}^{t-1} = \tilde{x}_{i+1}^t$$

and we obtain the claim. If $\tilde{x}_i^{t-1} > \tilde{x}_{i+1}^{t-1}$, then $\tilde{q}_i^{t-1} > \tilde{q}_{i+1}^{t-1}$ holds, and

$$\tilde{x}_i^{t+1} - \tilde{x}_{i+1}^{t+1} = \tilde{x}_i^t - \tilde{x}_{i+1}^t - \lambda_t(\tilde{q}_i^t - \tilde{q}_{i+1}^t) = (\tilde{q}_i^t - \tilde{q}_{i+1}^t)\left(\frac{\tilde{x}_i^t - \tilde{x}_{i+1}^t}{\tilde{q}_i^t - \tilde{q}_{i+1}^t} - \lambda_t\right) \geq 0$$

where the last inequality comes from the definition of λ_t, followed by $\lambda_t \leq \min_{i \in [n-1]} \left\{ \left(\tilde{x}_{i+1}^t - \tilde{x}_i^t\right) / \left(\tilde{q}_{i+1}^t - \tilde{q}_i^t\right) \mid \tilde{q}_{i+1}^t \neq \tilde{q}_i^t \right\}$. □

Lemma 3. *In Algorithm 4, $\tilde{\boldsymbol{x}}^{K+1}$ $(= \tilde{\boldsymbol{x}}^K - \lambda^K \tilde{\boldsymbol{q}}^K) = 0$ holds.*

Proof. Without loss of generality, we may assume that $\tilde{x}_1 \geq \tilde{x}_2 \geq \cdots \geq \tilde{x}_n$, for simplicity of notations. First we show $\tilde{\boldsymbol{x}}^{K+1} \geq 0$. Since Lemma 2, if there exists $j \in [n]$ satisfying that $\tilde{x}_j^{K+1} < 0$, then $\tilde{x}_n^{K+1} < 0$ holds. Thus it is enough to show $\tilde{x}_n^{K+1} \geq 0$. Let $i^* = \min\{j \in [n] \mid \tilde{x}_j^K = \tilde{x}_n^K\}$. Then we have $\tilde{x}_{i^*}^K = \tilde{x}_{i^*+1}^K = \cdots = \tilde{x}_n^K$ and $\tilde{q}_{i^*}^K = \tilde{q}_{i^*+1}^K = \cdots = \tilde{q}_n^K$. Hence, we get $\tilde{x}_{i^*}^{K+1} = \tilde{x}_{i^*+1}^{K+1} = \cdots = \tilde{x}_n^{K+1}$. In case of $i^* \geq 2$, $\tilde{x}_{i^*-1}^t > \tilde{x}_{i^*}^t$ holds for any $t \in [K]$, meaning that $\tilde{q}_{i^*-1}^t > \tilde{q}_{i^*}^t$ holds for any $t \in [K]$. Thus we can see that $\sum_{j=i^*}^n \tilde{q}_j^t = \tilde{g}(n) - \tilde{g}(i^*-1)$ holds for any $t \in [K]$, from the definition of $\tilde{\boldsymbol{q}}^t$. Then we obtain

$$\sum_{j=i^*}^n \sum_{t=1}^K \lambda_t \tilde{q}_j^t = \sum_{t=1}^K \lambda_t \left(\tilde{g}(n) - \tilde{g}(i^*-1)\right) = \tilde{g}(n) - \tilde{g}(i^*-1) \leq \sum_{j=i^*}^n \tilde{x}_j$$

where the last inequality is due to constraints of $B(\tilde{f})$ $\sum_{j=1}^{i^*-1} \tilde{x}_j \leq \tilde{g}(i^*-1)$ and $\sum_{j=1}^{n} \tilde{x}_j = \tilde{g}(n)$. Thus we obtain that $\sum_{j=i^*}^{n} \tilde{x}_j^{K+1} = \sum_{j=i^*}^{n} \left(\tilde{x}_j - \sum_{t=1}^{K} \lambda_t \tilde{q}_j^t\right) \geq 0$. As discussed above, $\tilde{x}_{i^*}^{K+1} = \tilde{x}_{i^*+1}^{K+1} = \cdots = \tilde{x}_n^{K+1}$ holds, and we obtain $\tilde{x}_n^{T+1} \geq 0$. In case of $i^* = 1$, the proof is done in a similar way.

Now we show $\tilde{\boldsymbol{x}}^{K+1} = 0$. Since $\tilde{\boldsymbol{x}} \in B(\tilde{f})$, $\sum_{j=1}^{n} \tilde{x}_j^{K+1} = \tilde{g}(n)$ holds. In a similar way as the proof of $\tilde{\boldsymbol{x}}^{K+1} \geq 0$,

$$\sum_{j=1}^{n} \sum_{t=1}^{K} \lambda_t \tilde{q}_j^t = \sum_{t=1}^{K} \lambda_t \sum_{j=1}^{n} \tilde{q}_j^t = \sum_{t=1}^{K} \lambda_t \tilde{g}(n) = \tilde{g}(n).$$

Since $\boldsymbol{x}^{K+1} \geq 0$, $\tilde{\boldsymbol{x}}^{K+1} = \tilde{\boldsymbol{x}} - \sum_{t=1}^{K} \lambda_t \tilde{\boldsymbol{q}}^t = 0$. □

Lemma 4. *The number of iterations K is at most n.*

Proof. From the definition of λ_t, there is at least one $i \in [n]$ satisfying that $\tilde{x}_i^t > \tilde{x}_{i+1}^t$ and $\tilde{x}_i^{t+1} = \tilde{x}_{i+1}^{t+1}$. If $\tilde{x}_i^t = \tilde{x}_{i+1}^t$, then $\tilde{x}_i^{t+1} = \tilde{x}_{i+1}^{t+1}$ as discussed in the proof of Lemma 2. Now the claim is clear. □

Proof of Theorem 10. Since Lemma 3, it is clear that the output $\sum_{t=0}^{K} \lambda_t \tilde{\boldsymbol{q}}^t$ by Algorithm 4 is equal to an arbitrarily given $\tilde{\boldsymbol{x}} \in B(\tilde{f})$. It is not difficult to see that every lines in Algorithm 4 is done in $O(n)$. Hence, the running time of Algorithm 4 is $O(n^2)$ by Lemma 4. □

Note that, by modifying Algorithm 4, we can design an algorithm for randomized rounding of $\boldsymbol{x} \in B(f)$ using only $O(n)$ space, with the same time complexity of $O(n^2)$. We can also improve the algorithm with a time complexity of $O(n \log n)$ using a heap, with $O(n)$ space.

5 Conclusion

In this paper, we consider an prediction problem over the base polyhedron defined by a submodular function and propose efficient prediction algorithms. An open problem is to derive a tight lower bound of the regret of our problem.

Acknowledgements. We thank anonymous reviewers for their helpful comments. This work is supported in part by JSPS Grand-in-Aid for Young Scientists (B) 23700178, JSPS Grand-in-Aid for Scientific Research (B) 23300003, and Aihara Project, the FIRST program from JSPS.

References

[1] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)

[2] Cesa-Bianchi, N., Lugosi, G.: Combinatorial Bandits. In: Proceedings of the 22nd Conference on Learning Theory (COLT 2009) (2009)

[3] Chopra, S.: On the spanning tree polyhedron. Operations Research Letters 8(1), 25–29 (1989)

[4] Edmonds, J.: Submodular functions, matroids, and certain polyhedra. In: Combinatorial Structures and Their Applications, pp. 69–87 (1970)

[5] Edmonds, J.: Matroids and the greedy algorithm. Mathematical Programming 1(1), 127–136 (1971)

[6] Freund, Y., Schapire, R.E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)

[7] Fujishige, S.: Submodular functions and optimization, 2nd edn. Elsevier Science (2005)

[8] Hazan, E.: The convex optimization approach to regret minimization. In: Sra, S., Nowozin, S., Wright, S.J. (eds.) Optimization for Machine Learning, ch. 10, pp. 287–304. MIT Press (2011)

[9] Helmbold, D.P., Warmuth, M.K.: Learning Permutations with Exponential Weights. Journal of Machine Learning Research 10, 1705–1736 (2009)

[10] Iwata, S.: Submodular function minimization. Mathematical Programming, Ser. B 112, 45–64 (2008)

[11] Kakade, S., Kalai, A.T., Ligett, L.: Playing games with approximation algorithms. SIAM Journal on Computing 39(3), 1018–1106 (2009)

[12] Kalai, A., Vempala, S.: Efficient algorithms for online decision problems. Journal of Computer and System Sciences 71(3), 291–307 (2005)

[13] Koolen, W.M., Warmuth, M.K., Kivinen, J.: Hedging Structured Concepts. In: Proceedings of the 23rd Conference on Learning Theory (COLT 2010), pp. 93–105 (2010)

[14] Nagano, K.: A faster parametric submodular function minimization algorithm and applications. Technical Report METR 2007–43, Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo (2007)

[15] Orlin, J.B.: A Faster Strongly Polynomial Time Algorithm for Submodular Function Minimization. In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 240–251. Springer, Heidelberg (2007)

[16] Warmuth, M.K., Kuzmin, D.: Randomized Online PCA Algorithms with Regret Bounds that are Logarithmic in the Dimension. Journal of Machine Learning Research 9, 2287–2320 (2008)

[17] Yasutake, S., Hatano, K., Kijima, S., Takimoto, E., Takeda, M.: Online Linear Optimization over Permutations. In: Asano, T., Nakano, S.-I., Okamoto, Y., Watanabe, O. (eds.) ISAAC 2011. LNCS, vol. 7074, pp. 534–543. Springer, Heidelberg (2011)

[18] Ziegler, G.M.: Lectures on Polytopes. Graduate Texts in Mathematics, vol. 152. Springer (1995)

[19] Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), pp. 928–936 (2003)

Lower Bounds on Individual Sequence Regret[*]

Eyal Gofer and Yishay Mansour

Tel Aviv University,
Tel Aviv, Israel
{eyalgofe,mansour}@post.tau.ac.il

Abstract. In this work, we lower bound the individual sequence anytime regret of a large family of online algorithms. This bound depends on the quadratic variation of the sequence, Q_T, and the learning rate. Nevertheless, we show that any learning rate that guarantees a regret upper bound of $O(\sqrt{Q_T})$ necessarily implies an $\Omega(\sqrt{Q_T})$ anytime regret on *any* sequence with quadratic variation Q_T.

The algorithms we consider are linear forecasters whose weight vector at time $t+1$ is the gradient of a concave potential function of cumulative losses at time t. We show that these algorithms include all linear Regularized Follow the Leader algorithms. We prove our result for the case of potentials with negative definite Hessians, and potentials for the best expert setting satisfying some natural regularity conditions. In the best expert setting, we give our result in terms of the translation-invariant *relative* quadratic variation. We apply our lower bounds to Randomized Weighted Majority and to linear cost Online Gradient Descent.

We show that bounds on anytime regret imply a lower bound on the price of "at the money" call options in an arbitrage-free market. Given a lower bound Q on the quadratic variation of a stock price, we give an $\Omega(\sqrt{Q})$ lower bound on the option price, for $Q < 0.5$. This lower bound has the same asymptotic behavior as the Black-Scholes pricing and improves a previous $\Omega(Q)$ result given in [4].

1 Introduction

For any sequence of losses, it is trivial to tailor an algorithm that has no regret on that particular sequence. The challenge and the great success of regret minimization algorithms lie in achieving low regret for *every* sequence. This bound may still depend on a measure of sequence smoothness, such as the quadratic variation or variance. The optimality of such regret upper bounds may be demonstrated by proving the existence of some "difficult" loss sequences. Their existence may be implied, for example, by stochastically generating sequences and proving a

[*] This research was supported in part by the Google Inter-university center for Electronic Markets and Auctions, by a grant from the Israel Science Foundation, by a grant from United States-Israel Binational Science Foundation (BSF), by a grant from the Israeli Ministry of Science (MoS), and by The Israeli Centers of Research Excellence (I-CORE) program, (Center No. 4/11). This work is part of Ph.D. thesis research carried out by the first author at Tel Aviv University.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 275–289, 2012.

lower bound on the expected regret of any algorithm (see, e.g., [2]). This type of argument leaves open the possibility that the difficult sequences are, in some way, atypical or irrelevant to actual user needs. In this work we address this question by proving lower bounds on the regret of *any* individual sequence, in terms of its quadratic variation.

We first consider the related task of characterizing algorithms that have individual sequence *non-negative* regret. We focus our attention on linear forecasters that determine their next weight vector as a function of current cumulative losses. More specifically, if $\mathbf{L}_t \in \mathbb{R}^N$ is the cumulative loss vector at time t, and $\mathbf{x}_{t+1} \in \mathcal{K}$ is the next weight vector, then $\mathbf{x}_{t+1} = g(\mathbf{L}_t)$ for some continuous g. The algorithm then incurs a loss of $\mathbf{x}_{t+1} \cdot \mathbf{l}_{t+1}$, where $\mathbf{l}_t$ is the loss vector at time t. We show that such algorithms have individual sequence non-negative regret if and only if g is the gradient of a concave potential function. We then show that this characteristic is shared by all linear cost Regularized Follow the Leader regret minimization algorithms, which include Randomized Weighted Majority (RWM) and linear cost Online Gradient Descent (OGD).

As our main result, we prove a trade-off between the upper bound on an algorithm's regret and a lower bound on its anytime regret, namely, its maximal regret for any prefix of the loss sequence. In particular, if the algorithm has a regret upper bound of $O(\sqrt{Q})$ for any sequence with quadratic variation Q, then it must have an $\Omega(\sqrt{Q})$ anytime regret on any sequence with quadratic variation $\Theta(Q)$.

We prove our result for two separate classes of continuously twice-differentiable potentials. One class has negative definite Hessians in a neighborhood of $\mathbf{L} = \mathbf{0}$, and includes OGD. The other comprises potentials for the best expert setting whose Hessians in a neighborhood of $\mathbf{L} = \mathbf{0}$ have positive off-diagonal entries; in other words, such potentials increase the weights of experts as their performance relatively improves, which is a natural property of regret minimization algorithms. For the first class, we use the usual Euclidean quadratic variation, or $\sum_{t=1}^{T} \|\mathbf{l}_t\|_2^2$. For the best expert setting, however, we use the more appropriate *relative quadratic variation*, $\sum_{t=1}^{T}(\max_i\{l_{i,t}\} - \min_i\{l_{i,t}\})^2$.

Our proof is comprised of several steps. We add a learning rate η to any potential Φ by defining a new potential $\Phi_\eta(\mathbf{L}) = (1/\eta)\Phi(\eta\mathbf{L})$. We give an exact expression for the regret using the Taylor expansion of the potential, and use it to prove an $\Omega(\min\{1/\eta, \eta Q\})$ lower bound on the anytime regret of any sequence with quadratic variation Q. In addition, we construct two specific loss sequences with variation Q, one with an $\Omega(1/\eta)$ regret, and the other with $\Omega(\eta Q)$ regret. Thus, we must have $\eta = \Theta(1/\sqrt{Q})$ to ensure a regret of $O(\sqrt{Q})$ for every sequence with variation Q, and our lower bound on the anytime regret becomes $\Omega(\sqrt{Q})$. We demonstrate our result on RWM, as an example of a best expert potential, and on linear cost OGD, as an example of a potential with a negative definite Hessian.

We apply our bounds to the financial problem of pricing *at the money call options*.[1] In the financial setting, an online best expert algorithm may be applied

[1] A call option is a financial instrument that pays its holder at time T the sum of $\max\{S_T - K, 0\}$, where S_t is the price of a given stock at time t, and K is a set price. The option is termed "at the money" if $K = S_0$.

to decide how to allocate wealth among a set of assets. In the case of call options, the assets are stock and cash. As observed in [4], an upper bound on the ratio between the return of an online algorithm and the return of the best asset implies a lower bound on call option prices in an arbitrage-free market.[2] We apply our anytime regret lower bounds to this problem by modifying RWM to "lock in" transient regret. (As suggested in [4], regret at any time can be made permanent by moving all weight to the current best expert.) We then present and utilize a general result bounding an algorithm's return in terms of its loss and the relative quadratic variation. We obtain a price lower bound of $\exp(0.1\sqrt{Q}) - 1 \approx 0.1\sqrt{Q}$ for $Q < 0.5$, where Q is the assumed quadratic variation of the stock's log price ratios. This bound has the same asymptotic behavior as the Black-Scholes price, which is approximately $\sqrt{Q}/\sqrt{2\pi} \approx 0.4\sqrt{Q}$, and improves on a previous result by [4], who gave a $Q/10$ lower bound for $Q < 1$.

Related Work. There are numerous results providing worst case upper bounds for regret minimization algorithms and showing their optimality (see [2]). In the best expert setting, RWM has been shown to have an optimal regret of $O(\sqrt{T \log N})$. In the online convex optimization paradigm ([14]), upper bounds of $O(\sqrt{T})$ have been shown, along with $\Omega(\sqrt{T})$ lower bounds for linear cost functions ([8]). In both cases, regret lower bounds are proved by invoking a stochastic adversary and lower bounding the expected regret. Upper bounds based on various notions of quadratic variation and variance are given in [3] for the expert setting, and in [10] for linear cost online convex optimization and the expert setting in particular.

A trade-off result is given in [5], where it is shown that a best expert algorithm with $O(\sqrt{T})$ regret must have a worst case $\Omega(\sqrt{T})$ regret to any fixed average of experts. To the best of our knowledge, there are no results that lower bound the regret on loss sequences in terms of their individual quadratic variation.

Outline. The outline of the paper is as follows. In Section 2 we provide notation and definitions. Section 3 characterizes algorithms with non-negative individual sequence regret, proves they include linear cost RFTL, and provides basic regret lower bounds. Section 4 presents our main result on the trade-off between upper bounds on regret and lower bounds on individual sequence anytime regret. In Section 5 we apply our bounds to linear cost OGD and to RWM. In Section 6 we show how our results can be used to lower bound the price of at the money call options. Due to space limitations, some proofs are omitted. (See [6] for a full version of this paper.)

2 Preliminaries

2.1 Regret Minimization

In the *best expert* setting, there are N available experts, and at each time step $1 \leq t \leq T$, an online algorithm A selects a distribution $\mathbf{p}_t$ over the N experts.

[2] In an arbitrage-free market, no algorithm trading in financial assets can guarantee profit without any risk of losing money.

After the choice is made, an adversary selects a loss vector $\mathbf{l}_t = (l_{1,t}, \ldots, l_{N,t}) \in \mathbb{R}^N$, and the algorithm experiences a loss of $l_{A,t} = \mathbf{p}_t \cdot \mathbf{l}_t$. We denote $L_{i,t} = \sum_{\tau=1}^{t} l_{i,\tau}$ for the cumulative loss of expert i at time t, $\mathbf{L}_t = (L_{1,t}, \ldots, L_{N,t})$, and $L_{A,t} = \sum_{\tau=1}^{t} l_{A,\tau}$ for the cumulative loss of A at time t. The *regret* of A at time T is $R_{A,T} = L_{A,T} - \min_j\{L_{j,T}\}$. The aim of a regret minimization algorithm is to achieve small regret regardless of the loss vectors chosen by the adversary. The *anytime regret* of A is the maximal regret over time, namely, $\max_t\{R_{A,t}\}$. We will sometimes use the notation $m(t) = \arg\min_i\{L_{i,t}\}$, where we take the smallest such index in case of a tie. Similarly, we denote $M(t)$ for the expert with the maximal loss at time t. We denote also $\delta(\mathbf{v}) = \max_i\{v_i\} - \min_i\{v_i\}$ for any $\mathbf{v} \in \mathbb{R}^N$, so $\delta(\mathbf{L}_t) = L_{M(t),t} - L_{m(t),t}$.

The best expert setting is a special case of the more general setting of *linear forecasting*. In this setting, at time t the algorithm chooses a weight vector $\mathbf{x}_t \in \mathbb{R}^N$, and incurs a loss of $\mathbf{x}_t \cdot \mathbf{l}_t$. In this paper we assume that the weight vectors are chosen from a compact and convex set $\mathcal{K}$. The regret of a linear forecaster A is then defined as $R_{A,T} = L_{A,T} - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u} \cdot \mathbf{L}_T\}$. The best expert setting is simply the case where $\mathcal{K} = \Delta_N$, the probability simplex over N elements.

Quadratic Variation. We define the *quadratic variation* of the loss sequence $\mathbf{l}_1, \ldots, \mathbf{l}_T$ as $Q_T = \sum_{t=1}^{T} \|\mathbf{l}_t\|_2^2$. For the best expert setting, we will use the slightly different notion of *relative quadratic variation*, defined as $q_T = \sum_{t=1}^{T} \delta(\mathbf{l}_t)^2$. We denote Q for a known lower bound on Q_T and q for a known lower bound on q_T.

2.2 Convex Functions

We mention here some basic facts about convex and concave functions that we will need. For more on convex analysis, see [13], [1], and [12], among others.

We will discuss functions defined on $\mathbb{R}^N$. A function $f : C \to \mathbb{R}$ is *convex*, if C is a convex set and if for every $\lambda \in [0,1]$ and $\mathbf{x}, \mathbf{y} \in C$, $f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y})$. f is *concave* if $-f$ is convex. f is *strictly convex* if the inequality is strict for $\mathbf{x} \neq \mathbf{y}$ and $\lambda \in (0,1)$. f is *strongly convex* with parameter $\alpha > 0$, if for every $\mathbf{x}, \mathbf{y} \in C$ and $\lambda \in [0,1]$, $f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y}) - (\alpha/2)\lambda(1-\lambda)\|\mathbf{x} - \mathbf{y}\|_2^2$. If f is differentiable on a convex set C, f is convex iff for every $\mathbf{x}, \mathbf{y} \in C$, $\nabla f(\mathbf{y}) \cdot (\mathbf{y} - \mathbf{x}) \geq f(\mathbf{y}) - f(\mathbf{x}) \geq \nabla f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x})$; f is strictly convex iff the above inequalities are strict for $\mathbf{x} \neq \mathbf{y}$. If f is twice differentiable, then it is convex iff its Hessian is positive semi-definite: for every $\mathbf{x} \in C$, $\nabla^2 \Phi(\mathbf{x}) \succeq 0$. The *convex conjugate* of f (defined on $\mathrm{dom} f$) is the function $f^*(\mathbf{y}) = \sup_{\mathbf{x}\in\mathrm{dom} f}\{\mathbf{x} \cdot \mathbf{y} - f(\mathbf{x})\}$, which is convex, and its effective domain is $\mathrm{dom} f^* = \{\mathbf{y} : f^*(\mathbf{y}) < \infty\}$.

2.3 Seminorms

A *seminorm* on $\mathbb{R}^N$ is a function $\|\cdot\| : \mathbb{R}^N \to \mathbb{R}$ with the following properties:

- Positive homogeneity: for every $a \in \mathbb{R}$, $\mathbf{x} \in \mathbb{R}^N$, $\|a\mathbf{x}\| = |a|\|\mathbf{x}\|$.
- Triangle inequality: for every $\mathbf{x}$, $\mathbf{x}' \in \mathbb{R}^N$, $\|\mathbf{x} + \mathbf{x}'\| \leq \|\mathbf{x}\| + \|\mathbf{x}'\|$.

Clearly, every norm is a seminorm. A seminorm satisfies $\|\mathbf{x}\| \geq 0$ for every $\mathbf{x}$, and $\|\mathbf{0}\| = 0$. However, unlike a norm, $\|\mathbf{x}\| = 0$ does not imply $\mathbf{x} = 0$. We will not deal with the trivial all-zero seminorm. Thus, there always exists a vector with non-zero seminorm, and by homogeneity, there exists a vector with seminorm a for any $a \in \mathbb{R}^+$.

2.4 Miscellaneous Notation

For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$, we denote $[\mathbf{x}, \mathbf{y}]$ for the line segment between $\mathbf{x}$ and $\mathbf{y}$, namely, $\{a\mathbf{x} + (1-a)\mathbf{y} : 0 \leq a \leq 1\}$. We use the notation $\mathrm{conv}(A)$ for the convex hull of a set $A \subseteq \mathbb{R}^N$, that is, $\mathrm{conv}(A) = \{\sum_{i=1}^k \lambda_i \mathbf{x}_i : \mathbf{x}_i \in A,\ \lambda_i \geq 0,\ i = 1, \ldots, k, \sum_{i=1}^k \lambda_i = 1\}$.

3 Non-negative Individual Sequence Regret

Our ultimate goal is to prove strictly positive individual sequence regret lower bounds for a variety of algorithms. In this section, we will characterize algorithms for which this goal is achievable, and prove some basic regret lower bounds. This will be done by considering the larger family of algorithms that have *non-negative* regret for any loss sequence. This family, it turns out, can be characterized exactly, and includes the important class of linear cost Regularized Follow the Leader algorithms.

We focus on linear forecasters whose vector at time t is determined as $\mathbf{x}_t = g(\mathbf{L}_{t-1})$, for $1 \leq t \leq T$, where $g : \mathbb{R}^N \to \mathcal{K} \subseteq \mathbb{R}^N$ is continuous and $\mathcal{K}$ is compact and convex. For such algorithms we can write $L_{A,T} = \sum_{t=1}^T g(\mathbf{L}_{t-1}) \cdot (\mathbf{L}_t - \mathbf{L}_{t-1})$.

A *non-negative-regret algorithm* satisfies that $L_{A,T} \geq \min_{\mathbf{u} \in \mathcal{K}} \{\mathbf{u} \cdot (\mathbf{L}_T - \mathbf{L}_0)\}$ for every $\mathbf{L}_0, \ldots, \mathbf{L}_T \in \mathbb{R}^N$. We point out that we allow both positive and negative losses. Synonymously, we will also say that g has non-negative regret. Note that if $\mathbf{L}_0 = \mathbf{L}_T$ (a closed cumulative loss path), then for any $\mathbf{u} \in \mathcal{K}$, it holds that $\mathbf{u} \cdot (\mathbf{L}_T - \mathbf{L}_0) = 0$, and non-negative regret implies that $\sum_{t=1}^T g(\mathbf{L}_{t-1}) \cdot (\mathbf{L}_t - \mathbf{L}_{t-1}) \geq 0$. The following theorem gives an exact characterization of non-negative-regret forecasters as the gradients of concave potentials. The proof, which uses basic properties of concave functions and conservative vector fields, is given in the appendix of [6].[3]

Theorem 1. *A linear forecaster based on a continuous function g has individual sequence non-negative regret iff there exists a concave potential function $\Phi : \mathbb{R}^N \to \mathbb{R}$ s.t. $g = \nabla \Phi$.*

As a by-product of the proof, we get the following:

[3] See the closely related Theorem 24.8 in [13] regarding *cyclically monotone mappings*. The proof we give is different in that it involves the regret and relates the loss to the path integral.

Corollary 1. *If algorithm A uses a non-negative-regret function $g(\mathbf{L}) = \nabla\Phi(\mathbf{L})$, then it holds that $R_{A,T} \geq \Phi(\mathbf{L}_T) - \Phi(\mathbf{L}_0) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot(\mathbf{L}_T - \mathbf{L}_0)\} \geq 0$.*

If Φ is continuously twice-differentiable, its second order Taylor expansion may be used to derive a similar lower bound, which now includes a non-negative quadratic regret term.

Theorem 2. *Let A be an algorithm using a non-negative-regret function $g = \nabla\Phi$, and let $\mathbf{L}_0, \ldots, \mathbf{L}_T \in \mathbb{R}^N$. If Φ is continuously twice-differentiable on the set* $\mathrm{conv}(\{\mathbf{L}_0, \ldots, \mathbf{L}_T\})$*, then*

$$R_{A,T} = \Phi(\mathbf{L}_T) - \Phi(\mathbf{L}_0) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot(\mathbf{L}_T - \mathbf{L}_0)\} - \frac{1}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2\Phi(\mathbf{z}_t)\mathbf{l}_t,$$

where $\mathbf{z}_t \in [\mathbf{L}_{t-1}, \mathbf{L}_t]$.

The proof proceeds by expressing $\Phi(\mathbf{L}_t)$ using the Taylor expansion of Φ around $\mathbf{L}_{t-1}$, for every t, and summing up those expressions. The quantities $\mathbf{z}_t$ arise from the remainder elements.

Note that the regret is the sum of two non-negative terms. We will sometimes refer to the first one, $\Phi(\mathbf{L}_T) - \Phi(\mathbf{L}_0) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot(\mathbf{L}_T - \mathbf{L}_0)\}$, as *the first order regret term*, and to the second, $-\frac{1}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2\Phi(\mathbf{z}_t)\mathbf{l}_t$, as *the second order regret term*. The second order term is non-negative by the concavity of Φ. These two terms play a key role in our bounds.

3.1 Relation to Regularized Follow the Leader

The class of concave potential algorithms contains the important class of linear cost Regularized Follow the Leader (RFTL) algorithms. (For an in-depth discussion of the RFTL algorithm, see [9].) Following [9], let $\mathcal{K} \subseteq \mathbb{R}^N$ be compact, non-empty, and convex. $RFTL(\eta, \mathcal{R})$ is a linear forecaster with two parameters: the learning rate, $\eta > 0$, and a strongly convex and continuously twice-differentiable regularizing function, $\mathcal{R} : \mathcal{K} \to \mathbb{R}$. $RFTL(\eta, \mathcal{R})$ determines its weights according to the rule $\mathbf{x}_{t+1} = g(\mathbf{L}_t) = \arg\min_{\mathbf{x}\in\mathcal{K}}\{\mathbf{x}\cdot\mathbf{L}_t + \mathcal{R}(\mathbf{x})/\eta\}$.

The next theorem shows that linear RFTL is a concave potential algorithm, with a potential function that is directly related to the convex conjugate of the regularizing function. The proof uses standard calculus and is given in the appendix of [6] for completeness.

Theorem 3. *If $\mathcal{R} : \mathcal{K} \to \mathbb{R}$ is strongly convex and $\eta > 0$, then $\Phi(\mathbf{L}) = (-1/\eta)\mathcal{R}^*(-\eta\mathbf{L})$ is concave and continuously differentiable on $\mathbb{R}^N$, and for every $\mathbf{L} \in \mathbb{R}^N$, it holds that $\nabla\Phi(\mathbf{L}) = \arg\min_{\mathbf{x}\in\mathcal{K}}\{\mathbf{x}\cdot\mathbf{L} + \mathcal{R}(\mathbf{x})/\eta\}$ and $\Phi(\mathbf{L}) = \min_{\mathbf{x}\in\mathcal{K}}\{\mathbf{x}\cdot\mathbf{L} + \mathcal{R}(\mathbf{x})/\eta\}$.*

It is now possible to lower bound the regret of $RFTL(\eta, \mathcal{R})$ by applying the lower bounds of Corollary 1 and Theorem 2.

Theorem 4. *The regret of RFTL*$(\eta, \mathcal{R})$ *satisfies*

$$R_{RFTL(\eta,\mathcal{R}),T} \geq \frac{1}{\eta}\left(\mathcal{R}(\mathbf{x}_{T+1}) - \mathcal{R}(\mathbf{x}_1)\right) + \left(\mathbf{x}_{T+1} \cdot \mathbf{L}_T - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u} \cdot \mathbf{L}_T\}\right) \geq 0,$$

and if R^* *is continuously twice-differentiable on* $\mathrm{conv}(\{-\eta\mathbf{L}_0, \ldots, -\eta\mathbf{L}_T\})$, *then*

$$R_{RFTL(\eta,\mathcal{R}),T} \geq \frac{1}{\eta}\left(\mathcal{R}(\mathbf{x}_{T+1}) - \mathcal{R}(\mathbf{x}_1)\right) + \left(\mathbf{x}_{T+1} \cdot \mathbf{L}_T - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u} \cdot \mathbf{L}_T\}\right) + \frac{\eta}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2 \mathcal{R}^*(-\eta\mathbf{z}_t)\mathbf{l}_t \geq 0,$$

where $\mathbf{z}_t \in [\mathbf{L}_{t-1}, \mathbf{L}_t]$.

Note that the first order regret term is split into two new non-negative terms, namely, $\left(\mathcal{R}(\mathbf{x}_{T+1}) - \mathcal{R}(\mathbf{x}_1)\right)/\eta$, and $\mathbf{x}_{T+1} \cdot \mathbf{L}_T - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u} \cdot \mathbf{L}_T\}$.

The first order regret lower bound given in Theorem 4 may be proved directly by extending the "follow the leader, be the leader" (FTL-BTL) Lemma ([11], see also [9]). See the appendix of [6] for the extension and proof.

4 Strictly Positive Individual Sequence Anytime Regret

In this section we give lower bounds on the anytime regret for two classes of potentials. These are the class of potentials with negative definite Hessians, and a rich class of best expert potentials that includes RWM.

We start by describing our general argument, which is based on the lower bounds of Theorem 2 and Corollary 1. Note first that these bounds hold for any $1 \leq T' \leq T$. Now, let C be some convex set on which Φ is continuously twice-differentiable. If $\mathbf{L}_0, \ldots, \mathbf{L}_T \in C$, then we may lower bound the regret by the second order term of Theorem 2, and further lower bound that term over any $\mathbf{L}_0, \ldots, \mathbf{L}_T \in C$. Otherwise, there is some $1 \leq T' \leq T$ s.t. $\mathbf{L}_{T'} \notin C$, and we may lower bound the regret at time T' by the first order term, using Corollary 1. We further lower bound that term over any $\mathbf{L}_{T'} \notin C$. The minimum of those two lower bounds gives a lower bound on the anytime regret. By choosing C properly, we will be able to prove that these two lower bounds are strictly positive.

We now present our analysis in more detail. Observe that for every $\eta > 0$ and concave potential $\Phi : \mathbb{R}^N \to \mathbb{R}$, $\Phi_\eta(\mathbf{L}) = (1/\eta)\Phi(\eta\mathbf{L})$ is also concave, with $\nabla\Phi_\eta(\mathbf{L}) = \nabla\Phi(\eta\mathbf{L})$. Let $\|\cdot\|$ be a non-trivial seminorm on $\mathbb{R}^N$. In addition, let $a > 0$ be such that Φ is continuously twice-differentiable on the set $\{\|\mathbf{L}\| \leq a\}$, and let $\mathbf{L}_0 = \mathbf{0}$.

Suppose algorithm A uses Φ_η and encounters the loss path $\mathbf{L}_0, \ldots, \mathbf{L}_T$. If there is some T' s.t. $\|\mathbf{L}_{T'}\| > a/\eta$, then applying Corollary 1 to Φ_η, we get

$$R_{A,T'} \geq \frac{1}{\eta}\left(\Phi(\eta\mathbf{L}_{T'}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u} \cdot \eta\mathbf{L}_{T'}\}\right).$$

Defining $\rho_1(a) = \inf_{\|\mathbf{L}\| \geq a}\{\Phi(\mathbf{L}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\}\}$, we have that $R_{A,T'} \geq \rho_1(a)/\eta$.

We next assume that $\|\mathbf{L}_t\| \leq a/\eta$ for every t. It is easily verified that the set $\{\|\mathbf{L}\| \leq a/\eta\}$ is convex, and since it contains $\mathbf{L}_t$ for every t, it also contains $\text{conv}(\{\mathbf{L}_0, \ldots, \mathbf{L}_T\})$. This means that $\Phi_\eta(\mathbf{L}) = (1/\eta)\Phi(\eta\mathbf{L})$ is continuously twice-differentiable on $\text{conv}(\{\mathbf{L}_0, \ldots, \mathbf{L}_T\})$. Applying Theorem 2 to Φ_η and dropping the non-negative first order term, we have

$$R_{A,T} \geq -\frac{1}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2 \Phi_\eta(\mathbf{z}_t)\mathbf{l}_t = -\frac{\eta}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2 \Phi(\eta\mathbf{z}_t)\mathbf{l}_t ,$$

where $\mathbf{z}_t \in [\mathbf{L}_{t-1}, \mathbf{L}_t]$. We now define $\rho_2(a) = \inf_{\|\mathbf{L}\|\leq a, \|\mathbf{l}\|=1}\{-\mathbf{l}^\top \nabla^2 \Phi(\mathbf{L})\mathbf{l}\}$. If $\|\mathbf{l}_t\| \neq 0$, then

$$-\mathbf{l}_t^\top \nabla^2 \Phi(\eta\mathbf{z}_t)\mathbf{l}_t = -(\mathbf{l}_t/\|\mathbf{l}_t\|)^\top \nabla^2 \Phi(\eta\mathbf{z}_t)(\mathbf{l}_t/\|\mathbf{l}_t\|)\|\mathbf{l}_t\|^2 \geq \rho_2(a)\|\mathbf{l}_t\|^2 ,$$

where we used the fact that $\|\eta\mathbf{z}_t\| \leq a$, which holds since $\mathbf{z}_t \in \text{conv}(\{\mathbf{L}_0, \ldots, \mathbf{L}_T\})$. Otherwise, $-\mathbf{l}_t^\top \nabla^2 \Phi(\eta\mathbf{z}_t)\mathbf{l}_t \geq 0 = \rho_2(a)\|\mathbf{l}_t\|^2$, so in any case,

$$R_{A,T} \geq -\frac{\eta}{2}\sum_{t=1}^{T} \mathbf{l}_t^\top \nabla^2 \Phi(\eta\mathbf{z}_t)\mathbf{l}_t \geq \frac{\eta}{2}\sum_{t=1}^{T}\rho_2(a)\|\mathbf{l}_t\|^2 = \frac{\eta}{2}\rho_2(a)\sum_{t=1}^{T}\|\mathbf{l}_t\|^2 .$$

Thus, we have

Lemma 1. *If $\|\mathbf{L}_t\| \leq a/\eta$ for every t, then $R_{A,T} \geq \frac{\eta}{2}\rho_2(a)\sum_{t=1}^{T}\|\mathbf{l}_t\|^2$. Otherwise, for any t s.t. $\|\mathbf{L}_t\| > a/\eta$, $R_{A,t} \geq \rho_1(a)/\eta$. Therefore,*

$$\max_t\{R_{A,t}\} \geq \min\left\{\frac{\rho_1(a)}{\eta}, \frac{\eta}{2}\rho_2(a)\sum_{t=1}^{T}\|\mathbf{l}_t\|^2\right\} .$$

Note that $\rho_1(a)$ is non-decreasing, $\rho_2(a)$ is non-increasing, and $\rho_1(a), \rho_2(a) \geq 0$. Lemma 1 therefore highlights a trade-off in the choice of a.

It still remains to bound ρ_1 and ρ_2 away from zero, and that will be done in two different ways for the cases of negative definite Hessians and best expert potentials. Nevertheless, the next technical lemma, which is instrumental in bounding ρ_1 away from zero, still holds in general. Essentially, it says that in the definition of $\rho_1(a)$, it suffices to take the infimum over $\{\|\mathbf{L}\| = a\}$ instead of $\{\|\mathbf{L}\| \geq a\}$.

Lemma 2. *It holds that $\rho_1(a) = \inf_{\|\mathbf{L}\|=a}\{\Phi(\mathbf{L}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\}\}$.*

We now present the main result of this section, assuming we have shown $\rho_1, \rho_2 > 0$. In what follows we denote $Q_T = Q_T(\mathbf{l}_1, \ldots, \mathbf{l}_T) = \sum_{t=1}^{T}\|\mathbf{l}_t\|^2$ for the generic quadratic variation w.r.t. the seminorm $\|\cdot\|$ (not to be confused with specific notions of quadratic variation). We denote $Q > 0$ for a given lower bound on Q_T.

Theorem 5. *Let $a > 0$ satisfy $\rho_1(a), \rho_2(a) > 0$.*

(i) For every $\eta > 0$, $\max_t\{R_{A,t}\} \geq \min\{\frac{\rho_1(a)}{\eta}, \frac{\eta}{2}\rho_2(a)Q\}$, and for $\eta = \sqrt{\frac{2\rho_1(a)}{\rho_2(a)Q}}$,

$$\max_t\{R_{A,t}\} \geq \sqrt{\rho_1(a)\rho_2(a)/2} \cdot \sqrt{Q}.$$

(ii) If for any sequence with quadratic variation $Q_T \leq Q'$ we have $R_{A,T} \leq c\sqrt{Q'}$, then for any such sequence,

$$\max_t\{R_{A,t}\} \geq \frac{\rho_1(a)\rho_2(a)}{2c} \cdot \frac{Q_T}{\sqrt{Q'}}.$$

In particular, if $Q_T = \Theta(Q')$, then $\max_t\{R_{A,t}\} = \Omega(\sqrt{Q_T})$.

Proof. (i) By Lemma 1,

$$\max_t\{R_{A,t}\} \geq \min\left\{\frac{\rho_1(a)}{\eta}, \frac{\eta}{2}\rho_2(a)\sum_{t=1}^{T}\|\mathbf{l}_t\|^2\right\} \geq \min\left\{\frac{\rho_1(a)}{\eta}, \frac{\eta}{2}\rho_2(a)Q\right\}.$$

Picking $\eta = \sqrt{\frac{2\rho_1(a)}{\rho_2(a)Q}}$ implies that $\frac{\rho_1(a)}{\eta} = \frac{\eta}{2}\rho_2(a)Q = \sqrt{\frac{1}{2}Q\rho_1(a)\rho_2(a)}$.

(ii) Given a and η, let $0 < \epsilon < a/\eta$ satisfy that $T = Q'/\epsilon^2$ is an integer. In addition, let $\mathbf{x} \in \mathbb{R}^N$ be such that $\|\mathbf{x}\| = 1$. The loss sequence $\mathbf{l}_t = (-1)^{t+1}\epsilon\mathbf{x}$, for $1 \leq t \leq T$, satisfies that $Q_T = \epsilon^2 T = Q'$ and that $\|\mathbf{L}_t\| = \|(1-(-1)^t)\epsilon\mathbf{x}/2\| \leq \epsilon < a/\eta$ for every t. Therefore,

$$c\sqrt{Q'} \geq R_{A,T} \geq \frac{\eta}{2}\rho_2(a)\sum_{t=1}^{T}\|\mathbf{l}_t\|^2 = \frac{\eta}{2}\rho_2(a)Q',$$

where the second inequality is by Lemma 1. This implies that $\eta \leq \frac{2c}{\rho_2(a)\sqrt{Q'}}$.

On the other hand, let $T > \frac{a^2}{\eta^2 Q'}$, define $\epsilon = \sqrt{Q'/T}$, and consider the loss sequence $\mathbf{l}_t = \epsilon\mathbf{x}$. Then $Q_T = \epsilon^2 T = Q'$, and $\|\mathbf{L}_T\| = \epsilon T = \sqrt{Q'T} > a/\eta$. Thus, again using Lemma 1, we have $c\sqrt{Q'} \geq R_{A,T} \geq \rho_1(a)/\eta$, which means that $\eta \geq \frac{\rho_1(a)}{c\sqrt{Q'}}$. Together, we have that $\frac{\rho_1(a)}{c\sqrt{Q'}} \leq \eta \leq \frac{2c}{\rho_2(a)\sqrt{Q'}}$, implying that $c \geq \sqrt{\rho_1(a)\rho_2(a)/2}$.[4] Given any sequence with $Q_T \leq Q'$, we have that $\frac{\rho_1(a)}{\eta} \geq \frac{\rho_1(a)\rho_2(a)\sqrt{Q'}}{2c}$ and $\frac{\eta}{2}\rho_2(a)Q_T \geq \frac{\rho_1(a)\rho_2(a)Q_T}{2c\sqrt{Q'}}$, so by Lemma 1, $\max_t\{R_{A,t}\} \geq \frac{\rho_1(a)\rho_2(a)}{2c} \cdot \frac{Q_T}{\sqrt{Q'}}$, concluding the proof. □

4.1 Potentials with Negative Definite Hessians

For this case, we pick $\|\cdot\|_2$ as our seminorm. Let $a > 0$ and let $\nabla^2\Phi(\mathbf{L}) \prec 0$ for $\mathbf{L}$ s.t. $\|\mathbf{L}\|_2 \leq a$. In this setting, the infimum in the definitions of $\rho_1(a)$ and $\rho_2(a)$ is equivalent to a minimum, using continuousness and the compactness of L^2 balls and spheres.

[4] Note that this means we cannot guarantee a regret upper bound of $c\sqrt{Q'}$ for $c < \sqrt{\rho_1(a)\rho_2(a)/2}$.

Lemma 3. *If* $\|\cdot\| = \|\cdot\|_2$*, then* $\rho_1(a) = \min_{\|\mathbf{L}\|=a}\{\Phi(\mathbf{L}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\}\}$ *and* $\rho_2(a) = \min_{\|\mathbf{L}\|\le a, \|\mathbf{l}\|=1}\{-\mathbf{l}^\top \nabla^2\Phi(\mathbf{L})\mathbf{l}\}$.

By Lemma 3, $\rho_2(a) = \min_{\|\mathbf{L}\|\le a, \|\mathbf{l}\|=1}\{-\mathbf{l}^\top \nabla^2\Phi(\mathbf{L})\mathbf{l}\} > 0$, where the inequality is true since the Hessians are negative definite, so we are taking the minimum of positive values. In addition, if $\mathbf{L} \ne \mathbf{0}$, then $\Phi(\mathbf{L}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\} > \nabla\Phi(\mathbf{L})\cdot\mathbf{L} - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\} \ge 0$, since Φ is strictly concave. Thus, again by Lemma 3, $\rho_1(a) = \min_{\|\mathbf{L}\|=a}\{\Phi(\mathbf{L}) - \Phi(\mathbf{0}) - \min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\}\} > 0$. The following statement is an immediate consequence of Theorem 5:

Theorem 6. *If* $\nabla^2\Phi(\mathbf{L}) \prec 0$ *for every* $\mathbf{L}$ *s.t.* $\|\mathbf{L}\|_2 \le a$*, for some* $a > 0$*, then*

(i) For every $\eta > 0$*, it holds that* $\max_t\{R_{A,t}\} \ge \min\{\frac{\rho_1(a)}{\eta}, \frac{\eta}{2}\rho_2(a)Q\}$*, and for* $\eta = \sqrt{\frac{2\rho_1(a)}{\rho_2(a)Q}}$*,* $\max_t\{R_{A,t}\} \ge \sqrt{\rho_1(a)\rho_2(a)/2}\cdot\sqrt{Q}$.

(ii) If for any sequence with quadratic variation $Q_T \le Q'$ *we have* $R_{A,T} \le c\sqrt{Q'}$*, then for any such sequence,* $\max_t\{R_{A,t}\} \ge \frac{\rho_1(a)\rho_2(a)}{2c}\cdot\frac{Q_T}{\sqrt{Q'}}$*. In particular, if* $Q_T = \Theta(Q')$*, then* $\max_t\{R_{A,t}\} = \Omega(\sqrt{Q_T})$.

4.2 The Best Expert Setting

In the best expert setting, where $\mathcal{K} = \Delta_N$, potentials can never be strictly concave, let alone have negative definite Hessians. To see that, let $\mathbf{L} \in \mathbb{R}^N$, $c \in \mathbb{R}$, and define $\mathbf{L}' = \mathbf{L} + c\cdot\mathbf{1}$, where $\mathbf{1}$ is the all-one vector. We will say that $\mathbf{L}'$ is a *uniform translation* of $\mathbf{L}$. Then

$$c = \nabla\Phi(\mathbf{L})\cdot(\mathbf{L}'-\mathbf{L}) \ge \Phi(\mathbf{L}') - \Phi(\mathbf{L}) \ge \nabla\Phi(\mathbf{L}')\cdot(\mathbf{L}'-\mathbf{L}) = c,$$

where we use the concavity of Φ, the fact that $\nabla\Phi$ is a probability vector, and the fact that $\mathbf{L}' - \mathbf{L} = c\cdot\mathbf{1}$. For a strictly concave Φ, the above inequalities would be strict if $c \ne 0$, but instead, they are equalities. Thus, the conditions for strict concavity are not fulfilled at any point $\mathbf{L}$.

We will replace the negative definite assumption with the assumption that for every $i \ne j$, $\frac{\partial^2\Phi}{\partial L_i \partial L_j} > 0$. This condition is natural for regret minimization algorithms, because $\frac{\partial^2\Phi(\mathbf{L})}{\partial L_i \partial L_j} = \frac{\partial p_i(\mathbf{L})}{\partial L_j}$, where p_i is the weight of expert i. Thus, we simply require that an increase in the cumulative loss of expert j results in an increase in the weight of every other expert (and hence a decrease in its own weight). A direct implication of this assumption is that $\frac{\partial\Phi(\mathbf{L})}{\partial L_i} > 0$ for every i and $\mathbf{L}$. To see that, observe that $p_i(\mathbf{L}) = 1 - \sum_{j\ne i} p_j(\mathbf{L})$, so $\frac{\partial p_i(\mathbf{L})}{\partial L_i} = -\sum_{j\ne i}\frac{\partial p_j(\mathbf{L})}{\partial L_i} < 0$. Since $p_i(\mathbf{L}) \ge 0$ and it is strictly decreasing in L_i, it follows that $p_i(\mathbf{L}) > 0$, or $\frac{\partial\Phi(\mathbf{L})}{\partial L_i} > 0$.

Using the above assumption we proceed to bound ρ_1 and ρ_2 away from zero. We first state some general properties of best expert potentials (proof in the appendix of [6]).

Lemma 4. *(i) Every row and column of* $\nabla^2\Phi$ *sum up to zero. (ii)* $\Phi(\mathbf{L})-\Phi(\mathbf{0})-\min_{\mathbf{u}\in\mathcal{K}}\{\mathbf{u}\cdot\mathbf{L}\}$ *is invariant w.r.t. a uniform translation of* $\mathbf{L}$*. (iii)* $\mathbf{l}^\top\nabla^2\Phi(\mathbf{L})\mathbf{l}$ *is invariant w.r.t. a uniform translation of either* $\mathbf{l}$ *or* $\mathbf{L}$*.*

We now consider $\rho_1(a)$ and $\rho_2(a)$ where we use the seminorm $\|\mathbf{v}\|=\delta(\mathbf{v})$. (The fact that $\delta(\mathbf{v})=\max_i\{v_i\}-\min_i\{v_i\}$ is a seminorm is easily verified.) Under this seminorm, $\sum_{t=1}^T\|\mathbf{l}_t\|^2$ becomes q_T, the relative quadratic variation. Note that $\delta(\mathbf{v})$ is invariant to uniform translation. In particular, for every $\mathbf{v}\in\mathbb{R}^N$ we may consider its "normalized" version, $\hat{\mathbf{v}}=\mathbf{v}-\min_i\{v_i\}\cdot\mathbf{1}$. We have that $\delta(\hat{\mathbf{v}})=\delta(\mathbf{v})$, $\hat{\mathbf{v}}\in[0,\delta(\mathbf{v})]^N$, and there exist entries i and j s.t. $\hat{\mathbf{v}}_i=0$ and $\hat{\mathbf{v}}_j=\delta(\mathbf{v})$. Denoting $\mathcal{N}(a)$ for the set of normalized vectors with seminorm a, we thus have that $\mathcal{N}(a)=\{\mathbf{v}\in[0,a]^N:\exists i,j \text{ s.t. } v_i=a,\ v_j=0\}$. The set $\mathcal{N}(a)$ is bounded and also closed, as a finite union and intersection of closed sets.

Using invariance to uniform translation, we can now show that the infima in the expressions for ρ_1 and ρ_2 may be taken over compact sets, and thus be replaced with minima. Using the requirement that $\frac{\partial^2\Phi}{\partial L_i\partial L_j}>0$ for every $i\neq j$, we can then show that the expressions inside the minima are positive. This is summarized in the next lemma.

Lemma 5. *For the best expert setting, it holds that* $\rho_1(a)=\min_{\mathbf{L}\in\mathcal{N}(a)}\{\Phi(\mathbf{L})\}-\Phi(\mathbf{0})>0$ *and* $\rho_2(a)=\min_{\mathbf{L}\in[0,a]^N,\mathbf{l}\in\mathcal{N}(1)}\{-\mathbf{l}^\top\nabla^2\Phi(\mathbf{L})\mathbf{l}\}>0$*.*

We can now apply Theorem 5 to the best expert setting.

Theorem 7. *If* $\frac{\partial^2\Phi}{\partial L_i\partial L_j}>0$ *for every* $i\neq j$ *and every* $\mathbf{L}$ *s.t.* $\delta(\mathbf{L})\le a$*, then*

(i) For every $\eta>0$*, it holds that* $\max_t\{R_{A,t}\}\ge\min\{\frac{\rho_1(a)}{\eta},\frac{\eta}{2}\rho_2(a)q\}$*, and for* $\eta=\sqrt{\frac{2\rho_1(a)}{\rho_2(a)q}}$, $\max_t\{R_{A,t}\}\ge\sqrt{\rho_1(a)\rho_2(a)/2}\cdot\sqrt{q}$*.*

(ii) If for any sequence with relative quadratic variation $q_T\le q'$ *we have* $R_{A,T}\le c\sqrt{q'}$*, then for any such sequence,* $\max_t\{R_{A,t}\}\ge\frac{\rho_1(a)\rho_2(a)}{2c}\cdot\frac{q_T}{\sqrt{q'}}$*. In particular, if* $q_T=\Theta(q')$*, then* $\max_t\{R_{A,t}\}=\Omega(\sqrt{q_T})$*.*

5 Application to Specific Regret Minimization Algorithms

5.1 Online Gradient Descent with Linear Costs

In this subsection, we deal with the Lazy Projection variant of the OGD algorithm ([14]) with a fixed learning rate η and linear costs. In this setting, for each t, OGD selects a weight vector according to the rule $\mathbf{x}_{t+1}=\arg\min_{\mathbf{x}\in\mathcal{K}}\{\|\mathbf{x}+\eta\mathbf{L}_t\|\}$, where $\mathcal{K}\subseteq\mathbb{R}^N$ is compact and convex. As observed in [10] and [9], this algorithm is equivalent to $RFTL(\eta,\mathcal{R})$, where $\mathcal{R}(\mathbf{x})=\frac{1}{2}\|\mathbf{x}\|_2^2$, namely, setting $\mathbf{x}_{t+1}=\arg\min_{\mathbf{x}\in\mathcal{K}}\{\mathbf{x}\cdot\mathbf{L}_t+(1/2\eta)\|\mathbf{x}\|_2^2\}$. In what follows we will make the assumption that $\mathcal{K}\supseteq B(\mathbf{0},a)$, where $B(\mathbf{0},a)$ is the closed ball with radius a centered at $\mathbf{0}$, for some $a>0$.

Note that solving the above minimization problem without the restriction $\mathbf{x} \in \mathcal{K}$ yields $\mathbf{x}'_{t+1} = -\eta \mathbf{L}_t$. However, if $\|\mathbf{L}_t\|_2 \le a/\eta$, then $\mathbf{x}'_{t+1} \in \mathcal{K}$, and then, in fact, $\mathbf{x}_{t+1} = -\eta \mathbf{L}_t$. By Theorem 3,

$$\begin{aligned}\Phi_\eta(\mathbf{L}_t) &= \mathbf{x}_{t+1} \cdot \mathbf{L}_t + (1/\eta)\mathcal{R}(\mathbf{x}_{t+1}) = -\eta \mathbf{L}_t \cdot \mathbf{L}_t + (1/2\eta)\| - \eta \mathbf{L}_t\|_2^2 \\ &= -(\eta/2)\|\mathbf{L}_t\|_2^2.\end{aligned}$$

Thus, if $\|\mathbf{L}\|_2 \le a$, then $\Phi(\mathbf{L}) = -(1/2)\|\mathbf{L}\|_2^2$ and also $\nabla^2 \Phi(\mathbf{L}) = -I$, where I is the identity matrix. By Lemma 3,

$$\rho_1(a) = \min_{\|\mathbf{L}\|_2 = a} \{-\frac{1}{2}\|\mathbf{L}\|_2^2 - \min_{\mathbf{u} \in \mathcal{K}}\{\mathbf{u} \cdot \mathbf{L}\}\} \ge \min_{\|\mathbf{L}\|_2 = a} \{-\frac{1}{2}\|\mathbf{L}\|_2^2 - (-\mathbf{L}) \cdot \mathbf{L}\} = \frac{1}{2}a^2,$$

where we used the fact that $-\mathbf{L} \in \mathcal{K}$ if $\|\mathbf{L}\|_2 = a$. In addition, by Lemma 3,

$$\rho_2(a) = \min{}_{\|\mathbf{L}\|_2 \le a, \|\mathbf{l}\|_2 = 1}\{-\mathbf{l}^\top(-I)\mathbf{l}\} = 1.$$

By Theorem 6, we have that $\max_t\{R_{A,t}\} \ge \min\{a^2/(2\eta), (\eta/2)Q\}$, and for $\eta = \frac{a}{\sqrt{Q}}$, $\max_t\{R_{A,t}\} \ge \frac{a}{2}\sqrt{Q}$.

5.2 Randomized Weighted Majority

RWM is the most notable regret minimization algorithm for the expert setting. We have $\mathcal{K} = \Delta_N$, and the algorithm gives a weight $p_{i,t+1} = \frac{p_{i,0} e^{-\eta L_{i,t}}}{\sum_{j=1}^N p_{j,0} e^{-\eta L_{j,t}}}$ to expert i at time $t+1$, where the initial weights $p_{i,0}$ and the learning rate η are parameters.

It is easy to see that for the potential $\Phi_\eta(\mathbf{L}) = -(1/\eta)\ln(\sum_{i=1}^N p_{i,0} e^{-\eta L_i})$, we have that $\mathbf{p} = (p_1, \ldots, p_N) = \nabla \Phi_\eta(\mathbf{L})$. The Hessian $\nabla^2 \Phi_\eta$ has the following simple form:

Lemma 6. *Let $\mathbf{L} \in \mathbb{R}^N$ and denote $\mathbf{p} = \nabla \Phi_\eta(\mathbf{L})$. Then $\nabla^2 \Phi_\eta(\mathbf{L}) = \eta \cdot (\mathbf{p}\mathbf{p}^\top - diag(\mathbf{p})) \preceq 0$, where $diag(\mathbf{p})$ is the diagonal matrix with $\mathbf{p}$ as its diagonal.*

We will assume $p_{1,0} = \ldots = p_{N,0} = 1/N$, and write $RWM(\eta)$ for RWM with parameters η and the uniform distribution. Thus, $\Phi(\mathbf{L}) = -\ln((1/N)\sum_{i=1}^N e^{-L_i})$, and we have by Lemma 6 that $\frac{\partial^2 \Phi}{\partial L_i \partial L_j} > 0$ for every $i \ne j$. Therefore, by Lemma 5, $\rho_1, \rho_2 > 0$. We now need to calculate ρ_1 and ρ_2. This is straightforward in the case of ρ_1, but for ρ_2 we give the value only for $N = 2$ (proof in [6]).

Lemma 7. *For any $N \ge 2$, $\rho_1(a) = \ln \frac{N}{N-1+e^{-a}}$. For $N = 2$, it holds that $\rho_2(a) = \left(e^{a/2} + e^{-a/2}\right)^{-2}$.*

Picking $a = 1.2$, we have by Theorem 7 that

Theorem 8. *For $N = 2$, there exists η s.t.*

$$\max_t\{R_{RWM(\eta),t}\} \ge \sqrt{\rho_1(a)\rho_2(a)q/2} \ge 0.195\sqrt{q}.$$

This bound will be used in the next section to lower bound the price of call options. We comment that using different techniques, we can derive a bound for any N (proof omitted):

Theorem 9. *For any* $0 < \alpha < 1$ *and* $\eta > 0$*, it holds that* $\max_t\{R_{RWM(\eta),t}\} \geq \min\{\frac{1}{\eta}\ln(\frac{N(1-\alpha)}{N-1}), \frac{1}{4}\eta\alpha q\}$*, and for* $\alpha = 1/(2N)$ *and* $\eta = \sqrt{(8N/q)\ln\frac{2N-1}{2N-2}}$*,* $\max_t\{R_{RWM(\eta),t}\} \geq \sqrt{q}/(4N)$*.*

6 Application to Call Option Pricing

6.1 The Investment Setting and Regret Minimization

In an investment setting, we encounter a multiplicative version of the best expert setting of regret minimization. There are N assets (experts), $\mathbf{X}_1, \ldots, \mathbf{X}_N$, where the value of $\mathbf{X}_i$ at time $0 \leq t \leq T$ is denoted by $X_{i,t}$, and we assume that $X_{i,t} > 0$ for every i and t. We define the *single period return* of $\mathbf{X}_i$ at time $1 \leq t \leq T$ by $r_{i,t} = X_{i,t}/X_{i,t-1} - 1$. A trading algorithm invests in the above assets by allocating on each day t a fraction $p_{i,t}$ of the total assets, V_{t-1}, to be invested in $\mathbf{X}_i$. The return of the algorithm on day t is defined as $r_{A,t} = V_t/V_{t-1} - 1$, and we have, by definition of V_t, that $r_{A,t} = \sum_i p_{i,t}r_{i,t}$. We aim to bound the quantity $V_T/\max_i\{X_{i,T}\}$ for all possible *price paths* $\{r_{i,t}\}$.

It is natural to translate the multiplicative scenario of regret minimization to the additive one by taking logarithms. Defining $l_{i,t} = -\ln(1 + r_{i,t})$, the cumulative losses of the assets translate directly to minus the logarithms of their total returns, that is, $L_{i,t} = \sum_{\tau=1}^{t} l_{i,\tau} = -\sum_{\tau=1}^{t}\ln(X_{i,\tau}/X_{i,\tau-1}) = -\ln(X_{i,t}/X_{i,0})$. In addition, we have $q_T = \sum_{t=1}^{T}\ln^2\left(\frac{1+\max_i\{r_{i,t}\}}{1+\min_i\{r_{i,t}\}}\right)$. However, the loss of an online algorithm does not translate seamlessly. We have that $L_{A,t} = \sum_{\tau=1}^{t} l_{A,\tau} = -\sum_{\tau=1}^{t}\sum_{i=1}^{N} p_{i,\tau}\ln(1 + r_{i,\tau})$, whereas

$$-\ln(V_t/V_0) = -\sum_{\tau=1}^{t}\ln(V_\tau/V_{\tau-1}) = -\sum_{\tau=1}^{t}\ln\sum_{i=1}^{N} p_{i,\tau}(1 + r_{i,\tau})$$
$$= -\sum_{\tau=1}^{t}\ln\left(1 + \sum_{i=1}^{N} p_{i,\tau}r_{i,\tau}\right).$$

Nevertheless, we can show that up to a factor of $q_T/8$, the multiplicative and additive notions of the loss of an online algorithm are the same.

Theorem 10. *The additive loss and the return satisfy* $0 \leq L_{A,T} + \ln(V_T/V_0) \leq q_T/8$*.*

6.2 Lower Bound on the Price of “at the Money” Call Options

We consider a scenario with two assets, where one is a stock $\mathbf{X}$, and the other is one unit of cash. A call option $\mathbf{C}(K, T)$ is a security that pays its holder at some future time T a sum of $\max\{X_T - K, 0\}$, where $K \geq 0$ is the pre-fixed *strike price*. The option is said to be "at the money" if $K = X_0$. W.l.o.g., the price of the stock at time 0 is $X_0 = 1$. We denote $C(K,T)$ for the price of $\mathbf{C}(K,T)$ at time 0. We assume the same basic market conditions as [4, 7].[5]

We may obtain a lower bound on $C(K,T)$ by showing that an online algorithm is always dominated by the option, as formalized in the next lemma.

Lemma 8. ([4]) *Let A be a trading algorithm with $V_0 = 1$. If for every price path and some $\beta > 0$, $V_T \leq \beta \max\{X_T, K\}$, then $C(K,T) \geq 1/\beta - K$.*

We next use the lower bound on the anytime regret of RWM to lower bound $C(1,T)$. Since our assets are stock and cash, q is a lower bound on $q_T = \sum_{t=1}^{T} \ln^2 \left(\frac{1+\max_i\{r_{i,t}\}}{1+\min_i\{r_{i,t}\}}\right) = \sum_{t=1}^{T} \ln^2 (1 + r_t)$, where r_t is the return of the stock at time t. The proof idea is as follows. First, we note that given a lower bound on the anytime regret of an algorithm, it may be modified to "lock in" that regret from the moment the bound is exceeded until time T. (This idea of locking in regret is suggested in [4].) Modifying RWM this way, we obtain an upper bound on $V_T/\max\{X_T, 1\}$ by Theorems 8 and 10. This leads to a lower bound on $C(1,T)$ by Lemma 8.

Theorem 11. *Assuming that $q_T \in [q, \gamma q]$, where $\gamma \geq 1$, it holds that*

$$C(1,T) \geq \exp(0.195\sqrt{q} - \gamma q/8) - 1.$$

If $q_T = q$ is assumed, then $\gamma = 1$, and we have

Corollary 2. *If $q < 0.5$, then $C(1,T) \geq \exp(0.1\sqrt{q}) - 1 \geq 0.1\sqrt{q}$.*

In comparison, the Black-Scholes pricing has an asymptotic value that corresponds to $\sqrt{q}/\sqrt{2\pi} \sim 0.4\sqrt{q}$ for small values of q (see the appendix of [6]). We comment that our bound allows for a fully adversarial choice of the sequence of stock returns, while the Black-Scholes model assumes a stochastic setting. For this reason, it is expected that our lower bound falls below the Black-Scholes price.

References

1. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
2. Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press (2006)
3. Cesa-Bianchi, N., Mansour, Y., Stoltz, G.: Improved second-order bounds for prediction with expert advice. Machine Learning 66(2-3), 321–352 (2007)
4. DeMarzo, P., Kremer, I., Mansour, Y.: Online trading algorithms and robust option pricing. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, pp. 477–486 (2006)

[5] Namely, that the market is arbitrage-free, the risk-free interest rate is zero, and it is possible to trade any real quantity of stocks with no transaction costs.

[5] Even-Dar, E., Kearns, M., Mansour, Y., Wortman, J.: Regret to the best vs. regret to the average. Machine Learning 72(1-2), 21–37 (2008)

[6] Gofer, E., Mansour, Y.: Lower bounds on individual sequence regret, http://www.cs.tau.ac.il/~eyalgofe/papers/indivseq_full.pdf

[7] Gofer, E., Mansour, Y.: Pricing Exotic Derivatives Using Regret Minimization. In: Persiano, G. (ed.) SAGT 2011. LNCS, vol. 6982, pp. 266–277. Springer, Heidelberg (2011)

[8] Hazan, E.: Efficient algorithms for online convex optimization and their applications. Ph.D. thesis, Princeton University (2006)

[9] Hazan, E.: The convex optimization approach to regret minimization. In: Sra, S., Nowozin, S., Wright, S.J. (eds.) Optimization for Machine Learning. MIT Press (2011)

[10] Hazan, E., Kale, S.: Extracting certainty from uncertainty: regret bounded by variation in costs. Machine Learning 80(2-3), 165–188 (2010)

[11] Kalai, A., Vempala, S.: Efficient algorithms for online decision problems. J. Comput. Syst. Sci. 71(3), 291–307 (2005); Special issue Learning Theory 2003

[12] Nesterov, Y.: Introductory Lectures on Convex Optimization: A Basic Course. Kluwer Academic Publishers (2004)

[13] Rockafellar, R.T.: Convex Analysis. Princeton University Press (1970)

[14] Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: ICML, pp. 928–936 (2003)

A Closer Look at Adaptive Regret

Dmitry Adamskiy[1], Wouter M. Koolen[1], Alexey Chernov[2], and Vladimir Vovk[1]

[1] Computer Learning Research Centre and Department of Computer Science,
Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK
[2] Department Mathematical Sciences, Durham University, Durham, DH1 3LE, UK

Abstract. For the prediction with expert advice setting, we consider methods to construct algorithms that have low adaptive regret. The adaptive regret of an algorithm on a time interval $[t_1, t_2]$ is the loss of the algorithm there minus the loss of the best expert. Adaptive regret measures how well the algorithm approximates the best expert locally, and it is therefore somewhere between the classical regret (measured on all outcomes) and the tracking regret, where the algorithm is compared to a good sequence of experts.

We investigate two existing intuitive methods to derive algorithms with low adaptive regret, one based on specialist experts and the other based on restarts. Quite surprisingly, we show that both methods lead to the same algorithm, namely Fixed Share, which is known for its tracking regret. Our main result is a thorough analysis of the adaptive regret of Fixed Share. We obtain the exact worst-case adaptive regret for Fixed Share, from which the classical tracking bounds can be derived. We also prove that Fixed Share is optimal, in the sense that no algorithm can have a better adaptive regret bound.

Keywords: Online learning, adaptive regret, Fixed Share, specialist experts.

1 Introduction

This paper deals with the prediction with expert advice setting. Nature generates outcomes step by step. At every step Learner tries to predict the outcome. Then the actual outcome is revealed and the quality of Learner's prediction is measured by a loss function.

No assumptions are made about the nature of the data. Instead, at every step Learner is presented with the predictions of a pool of experts and he may base his predictions on these. The goal of Learner in the classical setting is to guarantee small regret, that is, to suffer cumulative loss that is not much larger than that of the best (in hindsight) expert from the pool. Several classical algorithms exist for this task, including the Aggregating Algorithm [13] and the Exponentially Weighted Forecaster [3]. In the standard log-loss game the regret incurred by those algorithms when competing with N experts is at most $\ln N$.

A common extension of the framework takes into account the fact that the best expert could change with time. In this case we may be interested in competing

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 290–304, 2012.

with the best *sequence* of experts from the pool. Known algorithms for this task include Fixed Share [8] and Mixing Past Posteriors [1].

In this paper we focus on the related task of obtaining small *adaptive* regret, a notion first considered in [11] and later studied in [7]. The adaptive regret of an algorithm on a time interval $[t_1, t_2]$ is the loss of the algorithm there, minus the loss of the best expert for that interval:

$$R_{[t_1,t_2]} := L_{[t_1,t_2]} - \min_j L^j_{[t_1,t_2]}$$

The goal is now to ensure small regret on all intervals simultaneously. Note that adaptive regret was defined in [7] with a maximum over intervals, but we need the fine-grained dependence on the endpoint times to be able to prove matching upper and lower bounds.

Our results. The contribution of our paper is twofold.

1. We study two constructions to get adaptive regret algorithms from existing classical regret algorithms. The first one is a simple construction which originates in [5] and [4] and involves so called sleeping (specialist) experts, and the second one uses restarts, as proposed in [7]. Although conceptually dissimilar, we show that both constructions reduce to the Fixed Share algorithm with variable switching rate.
2. We compute the exact adaptive worst-case regret of Fixed Share and show that no algorithm can have better adaptive regret. We also derive the tracking regret bounds from the adaptive regret bounds, showing that the latter are in fact more fundamental.

Here is a sneak preview of the adaptive bounds we obtain, presented in a slightly relaxed form for simplicity. The refined statement can be found in Theorem 4 below. In the log-loss game for each of the following adaptive regret bounds there is an algorithm achieving it, simultaneously for all the intervals $[t_1, t_2]$:

$$\ln N + \ln t_2 \,, \tag{1a}$$

$$\ln N + \ln t_1 + \ln\ln t_2 + 2 \,, \tag{1b}$$

$$\ln N + 2\ln t_1 + 1 \,, \tag{1c}$$

where $\ln\ln 1$ is interpreted as 0.

Outline. The structure of the paper is as follows. In Section 2 we give the description of the protocol and review the standard algorithms. In Section 3 we study two intuitive ways of obtaining adaptive regret algorithms from classical algorithms. We show that curiously both these algorithms turn out to be Fixed Share. In Section 4 we study in detail the adaptive regret of Fixed Share.

2 Setup

We phrase our results in the setting defined in Protocol 1 which, for lack of a standard name, we call *mix loss*. We choose this fundamental setting because

Protocol 1. Mix loss prediction

```
for $t = 1, 2, \dots$ do
    Learner announces probability vector $\boldsymbol{u}_t \in \triangle_N$
    Reality announces loss vector $\boldsymbol{\ell}_t \in [-\infty, \infty]^N$
    Learner suffers loss $\ell_t := -\ln \sum_n u_t^n e^{-\ell_t^n}$
end for
```

it is universal, in the sense that many other common settings reduce to it. For example probability forecasting, sequential investment and data compression are straightforward instances [3]. In addition, mix loss is the baseline for the wider class of *mixable loss functions*, which includes e. g. square loss [14]. Classical regret bounds transfer from mix loss to mixable losses almost by definition, and the same reasoning extends to adaptive regret bounds. In addition, mix loss results carry over in the usual modular ways (via Hoeffding and related bounds) to non-mixable games, which include the Hedge setting [6] and Online Convex Optimisation [16].

Let us introduce two standard algorithms in this setup. The *Aggregating Algorithm* [15] is parametrised by a prior distribution $\boldsymbol{u}_1$ on $[N]$ (where $[N]$ denotes the set $\{1, \dots, N\}$). It predicts in trial t with

$$u_t^n \;:=\; \frac{u_1^n e^{-\sum_{s<t} \ell_s^n}}{\sum_n u_1^n e^{-\sum_{s<t} \ell_s^n}}, \tag{2a}$$

which we may also maintain incrementally using the update rule

$$u_{t+1}^n \;=\; \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}}. \tag{2b}$$

For this algorithm with uniform prior $u_1^n = 1/N$, the classical regret bound states that for each expert j

$$\sum_{t=1}^T \ell_t - \sum_{t=1}^T \ell_t^j \;\leq\; \ln N.$$

Note that AA is minimax for classical mix loss regret since $\geq \ln N$ can be inflicted on any algorithm. The second algorithm, *Fixed Share* [8], in addition to a prior $\boldsymbol{u}_1$ requires a sequence of switching rates $\alpha_2, \alpha_3, \dots$ Intuitively, α_t is the probability of a switch in the sequence of "best-at-the-step" experts *before* trial t. The weights are now updated as

$$u_{t+1}^n \;:=\; \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}}\,. \tag{3}$$

(We see that the Aggregating Algorithm is the special case when all α_t are 0.) The tracking regret bound for Fixed Share with uniform prior $\boldsymbol{u}_1$ and constant

$\alpha_t = \alpha$ switching rate states that for any reference sequence $j_1, \ldots, j_T$ of experts with m blocks (and hence $m-1$ switches)

$$\sum_{t=1}^T \ell_t - \sum_{t=1}^T \ell_t^{j_t} \;\leq\; \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha).$$

Having introduced the standard classical and tracking regret algorithms, we now turn to adaptive regret.

3 Intuitive Algorithms with Low Adaptive Regret

Two methods have been proposed in the literature that can be used to obtain adaptive regret bounds: specialist experts and restarts. We discuss both and show that each of them yields Fixed Share with a particular choice of time-dependent switching rate α_t.

3.1 Specialist Experts

One way of getting an adaptive algorithm is the following. We create a pool of virtual experts. For each real expert n and time t, we include a virtual expert that mimics Learner's behaviour for the first $t-1$ trials (which is another way to say that this expert is a specialist [5] that abstains from prediction, or *sleeps*, during the first $t-1$ trials), and predicts as expert n from trial t onward. Then the classical regret w.r.t. this virtual expert on $[1, T]$ is the same as the adaptive regret w.r.t. the real expert n on $[t, T]$ because on the first $t-1$ steps the loss of the virtual expert equals Learner's loss. The natural idea is to feed all those virtual experts into the existing algorithm capable of obtaining good classical regret, the AA. For fixed t_2, the uniform prior on wake-up time $t_1 \leq t_2$ and expert n would lead to adaptive regret $\ln(Nt_2)$. It turns out that the same holds even without knowledge of t_2.

There is a snag, namely that in the prediction step you need to know the losses of the sleeping virtual specialists which are equal to the yet unknown loss of the Learner. However, it is possible to find a fixed point prediction which makes the AA loss exactly the same as if it took into account the sleeping experts making the same prediction. To avoid dealing with equations involving an infinite number of sleeping experts let us fix a time horizon $T > t$. Later we will see that this time horizon plays no role.

Let us denote by $w_t^{n,s}$ the probability assigned by the AA in trial t to the virtual specialist parametrised by real expert n and wake-up time s. Learner then will predict with weights $\boldsymbol{u}_t$ where $u_t^n = \sum_{s=1}^t w_t^{n,s} \Big/ \sum_{j=1}^N \sum_{\tau=1}^t w_t^{j,\tau}$. The desired fixed point property is achieved for this prediction:

$$\ell_t \;\coloneqq\; -\ln\left(\sum_{n=1}^N u_t^n e^{-\ell_t^n}\right) \;=\; -\ln\left(\sum_{n=1}^N \sum_{s=1}^t w_t^{n,s} e^{-\ell_t^n} + \sum_{n=1}^N \sum_{s=t+1}^T w_t^{n,s} e^{-\ell_t}\right).$$

That is, the loss ℓ_t of the prediction $\boldsymbol{u}_t$ in the game with N real experts equals the loss of the prediction $\boldsymbol{w}_t$ in the game with TN virtual specialists, where specialists that are still asleep are assumed to suffer Learner's loss ℓ_t.

At first glance, it is very inefficient to maintain weights of TN specialists. However, we do not need to, since we may merge the weights of all awake specialists associated to the same real expert, resulting in Algorithm 1. To verify this, denote this merged (unnormalised) weight in trial t by v_t^n for each real expert n. The merged (unnormalised) weight v_{t+1}^n of this real expert n in the next trial $t+1$ consists of the prior weight of the newly awaken virtual specialist plus v_t^n, the sum of the old weights, each multiplied by the same factor $e^{(\ell^t - \ell_t^n)}$ (as they were all awake). Thus we can update the sum directly, and this is reflected by our update rule.

Note that for simplicity, we have taken the prior on experts and wake-up times independent, i. e.

$$p^{(n,t)} \; = \; p(t)\,.$$

Also note that there is no need for the priors $p^{(n,t)}$ to normalise.

Algorithm 1. Adaptive Aggregating Algorithm

```
Input: Prior nonnegative weights $p(t)$, $t = 1, 2, \ldots$
  $v_1^n := p(1), n = 1, \ldots, N$
  for $t = 1, 2, \ldots$ do
    Play weights $u_t^n := \frac{v_t^n}{\sum_{k=1}^N v_t^k}$
    Read the experts losses $\ell_t^n$, $n = 1, \ldots, N$
    Set $v_{t+1}^n := p(t+1) + v_t^n \frac{e^{-\ell_t^n}}{\sum_{k=1}^N u_t^k e^{-\ell_t^k}}$, $n = 1, \ldots, N$
  end for
```

Now we will see that Algorithm 1 turns out to be Fixed Share with variable switching rate. In the rest of this section we derive this. Let $P(t) = \sum_{s=1}^t p(s)$.

Fact 1. *The update step of Algorithm 1 preserves the following:*

$$\sum_n v_t^n = \sum_n \sum_{s \le t} p(s) = NP(t)\,.$$

Proof. This follows immediately from expanding the one-step update rule:

$$\begin{aligned}
\sum_n v_{t+1}^n \;&=\; \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&=\; \sum_n p(t+1) + \sum_n v_t^n \frac{e^{-\ell_t^n}}{\sum_k \frac{v_t^k}{\sum_j v_t^j} e^{-\ell_t^k}} \\
&=\; Np(t+1) + \sum_n v_t^n \overset{\text{Induction}}{=} NP(t+1)\,.
\end{aligned}$$

□

We now show that Algorithm 1 can be seen as Fixed Share (and vice versa).

Lemma 2. *Say that α_t is the probability of a Fixed Share switch before trial t, and $p(t)$ is the prior weight of specialist waking up in trial t in Algorithm 1. Then the following conversion preserves behaviour*

$$p(t) = \frac{\frac{N}{N-1}\alpha_t}{\prod_{s=2}^{t}(1-\frac{N}{N-1}\alpha_s)}, \qquad \alpha_t = \frac{N-1}{N}\frac{p(t)}{\sum_{s=1}^{t} p(s)},$$

where we use the convention that $\alpha_1 = \frac{N-1}{N}$.

Proof. Let us rewrite the update step of Algorithm 1 for the normalised weights.

$$\begin{aligned}
u_{t+1}^n &= \frac{v_{t+1}^n}{\sum_k v_{t+1}^k} = \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} v_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{p(t+1)}{NP(t+1)} + \frac{1}{NP(t+1)} NP(t) u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{\alpha_{t+1}}{N-1} + \frac{P(t+1)-p(t+1)}{P(t+1)} u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}} \\
&= \frac{\alpha_{t+1}}{N-1} + \left(1 - \frac{N}{N-1}\alpha_{t+1}\right) u_t^n \frac{e^{-\ell_t^n}}{\sum_k u_t^k e^{-\ell_t^k}}.
\end{aligned}$$

We see that the weight update is the update of the Fixed Share algorithm with variable switching rate α_t. □

The idea to use specialist experts for obtaining adaptive bounds was introduced in [5]. There a virtual specialist is created for every interval $[t_1, t_2]$ which leads to redundancy and suboptimal bounds. Their adaptive regret bounds sport a term which exceeds $2 \ln t_2$ whereas our bounds (1) have at most a single $\ln t_2$.

3.2 Restarts

A second intuitive method to obtain adaptive regret bounds, called Follow the Leading History (FLH), was introduced in [7]. One starts with a base algorithm that ensures low classical regret. FLH then obtains low adaptive regret by restarting a copy of this base algorithm each trial, and aggregating the predictions of these copies. To get low adaptive regret w.r.t. N experts, it is natural to take the AA as the base algorithm. We now show that FLH with this choice equals Fixed Share with switching rate $\alpha_t = \frac{N-1}{Nt}$.

For each n, s and $t \geq s$, let $p_t^{n|s}$ denote the weight allocated to expert n by the copy of the AA started at time s. By definition $p_s^{n|s} = 1/N$, and these weights evolve according to (2b). We denote by p_t^s the weight allocated by FLH in trial $t \geq s$ to the copy of AA started at time s. In [7], these weights are defined as follows. Initially $p_1^1 = 1$ and subsequently

$$p_{t+1}^s = \left(1 - \frac{1}{t+1}\right) \frac{p_t^s e^{-\left(-\ln \sum_n p_t^{n|s} e^{-\ell_t^n}\right)}}{\sum_{r=1}^{t} p_t^r e^{-\left(-\ln \sum_n p_t^{n|r} e^{-\ell_t^n}\right)}}, \qquad p_{t+1}^{t+1} = \frac{1}{t+1}.$$

Lemma 3. *For mix loss, FLH with AA as the base algorithm issues the same predictions as Fixed Share with learning rate* $\alpha_t = \frac{N-1}{Nt}$.

Proof. We prove by induction on t that the FS and FLH weights coincide:

$$u_t^n \;=\; \sum_{s=1}^{t} p_t^{n|s} p_t^s.$$

The base case $t = 1$ is obvious. For the induction step we expand

$$\begin{aligned}
\sum_{s=1}^{t+1} p_{t+1}^{n|s} p_{t+1}^s \;&=\; \sum_{s=1}^{t} p_{t+1}^{n|s} p_{t+1}^s + p_{t+1}^{t+1}/N \\
&=\; \left(1 - \frac{1}{t+1}\right) \sum_{s=1}^{t} \left(\frac{p_t^{n|s} e^{-\ell_t^n}}{\sum_n p_t^{n|s} e^{-\ell_t^n}} \frac{p_t^s \left(\sum_n p_t^{n|s} e^{-\ell_t^n}\right)}{\sum_{r=1}^t p_t^r \left(\sum_n p_t^{n|r} e^{-\ell_t^n}\right)} \right) + \frac{1}{N(t+1)} \\
&=\; \left(1 - \frac{1}{t+1}\right) \frac{\sum_{s=1}^t p_t^s p_t^{n|s} e^{-\ell_t^n}}{\sum_{r=1}^t \sum_n p_t^r p_t^{n|r} e^{-\ell_t^n}} + \frac{1}{N(t+1)} \\
&\overset{\text{Induction}}{=}\; \left(1 - \frac{1}{t+1}\right) \frac{u_t^n e^{-\ell_t^n}}{\sum_n u_t^n e^{-\ell_t^n}} + \frac{1}{N(t+1)} \;=\; u_{t+1}^n,
\end{aligned}$$

and find the Fixed Share update equation (3) for switching rate $\alpha_t = \frac{N-1}{Nt}$. □

4 The Adaptive Regret of Fixed Share

We have seen in the previous section that both intuitive approaches to obtain algorithms with low adaptive regret result in Fixed Share. We take this convergence to mean that Fixed Share is the most fundamental adaptive algorithm. The tracking regret for Fixed Share is already well-studied. In this section we thoroughly analyse the adaptive regret of Fixed Share. We obtain the worst-case adaptive regret for mix loss. This result implies the known tracking regret bounds.

We also show an information-theoretic lower bound for mix loss that must hold for any algorithm, and which is tight for Fixed Share. This proves that Fixed Share is a Pareto-optimal algorithm for the mix loss game, in the sense that no other algorithm can guarantee essentially better adaptive regret.

4.1 The Exact Worst-Case Adaptive Regret for Mix Loss

In this section we first compute the exact worst-case adaptive regret of Fixed Share with arbitrary switching rate α_t. Then we obtain certain regret bounds of interest, including the tracking regret bound, for particular choices of α_t.

Theorem 4. *The worst-case adaptive regret of Fixed Share with N experts on interval $[t_1, t_2]$ equals*

$$-\ln\left(\frac{\alpha_{t_1}}{N-1}\prod_{t=t_1+1}^{t_2}(1-\alpha_t)\right). \tag{4}$$

Proof. The proof consists of two parts. First we claim that the worst-case data for the interval $[t_1, t_2]$ in the setting of Protocol 1 is rather simple: on the interval there is one *good* expert (all others get infinite losses) and on the single trial before the interval (if $t_1 > 1$) this expert suffers infinite loss while others do not. The proof of this can be found in Appendix A.

Now we will compute the regret on this data. The regret of Fixed Share on the interval $[t_1, t_2]$ is $-\ln$ of the product of the weights put on the good expert (say, j) on this interval:

$$R^{\mathrm{FS}}_{[t_1,t_2]} = -\ln\prod_{t_1\le t\le t_2} u_t^j\,.$$

It is straightforward to derive $u_{t_1}^j$ from (3):

$$u_{t_1}^j = \frac{\alpha_{t_1}}{N-1} \qquad \text{and} \qquad u_t^j = 1-\alpha_t \qquad \text{for } t\in[t_1+1, t_2]$$

from which the statement follows. □

Example 1: Constant Switching Rate. This is the original Fixed Share [8].

Corollary 5. *Fixed Share with constant switching rate $\alpha_t = \alpha$ for $t > 1$ (recall that $\alpha_1 = \frac{N-1}{N}$) has worst-case adaptive regret equal to*

$$\begin{aligned}&\ln(N-1) - \ln\alpha - (t_2-t_1)\ln\big(1-\alpha\big) && \textit{for } t_1 > 1,\ \textit{and}\\ &\ln N - (t_2-1)\ln\big(1-\alpha\big) && \textit{for } t_1 = 1.\end{aligned}$$

A slightly weaker upper bound was obtained in [2]. The clear advantage of our analysis with equality is that we can obtain the standard Fixed Share tracking regret bound by summing the above adaptive regret bounds on individual intervals. Comparing with the best sequence of experts S on the interval $[1, T]$ with m blocks, we obtain the bound

$$L^{\mathrm{FS}}_{[1,T]} - L^{S}_{[1,T]} \;\le\; \ln N + (m-1)\ln(N-1) - (m-1)\ln\alpha - (T-m)\ln(1-\alpha)\,,$$

which is exactly the Fixed Share standard bound. So we see that the reason why Fixed Share can effectively compete with switching sequences is that it can, in fact, effectively compete with an expert on any interval, that is, has small adaptive regret.

Example 2: Slowly Decreasing Switching Rate. The idea of slowly decreasing the switching rate was considered in [12] in the context of source coding, and later analysed for expert switching in [10]; we saw in Section 3.2 that it also underlies Follow the Leading History of [7]. It results in tracking regret bounds that are almost as good as the bounds for constant α *with optimally tuned* α. These tracking bounds are again implied by the following corresponding adaptive regret bound.

Corollary 6. *Fixed Share with switching rate* $\alpha_t = 1/t$ *(except for* $\alpha_1 = \frac{N-1}{N}$*) has worst-case adaptive regret*

$$-\ln\left(\frac{1}{(N-1)t_1}\prod_{t=t_1+1}^{t_2}\frac{t-1}{t}\right) = \ln(N-1) + \ln t_2 \qquad \textit{for } t_1 > 1, \textit{ and} \tag{5a}$$

$$-\ln\left(\frac{1}{N}\prod_{t=2}^{t_2}\frac{t-1}{t}\right) = \ln N + \ln t_2 \qquad \textit{for } t_1 = 1. \tag{5b}$$

Example 3: Quickly Decreasing Switching Rate. The bounds we have obtained so far depend on t_2 either linearly or logarithmically. To get bounds that depend on t_2 sub-logarithmically, or even not at all, one may instead decrease the switching rate faster than $1/t$, as analysed in [12, 9]. To obtain a controlled trade-off, we consider setting the switching rate to $\alpha_t = \frac{1}{t\ln t}$, except for $\alpha_1 = \frac{N-1}{N}$. This leads to adaptive regret at most

$$\ln(N-1) + \ln t_1 + \ln\ln t_1 - \sum_{t=t_1+1}^{t_2}\ln\left(1 - \frac{1}{t\ln t}\right) \\ \leq \ln(N-1) + \ln t_1 + \ln\ln t_2 + 1.28 \tag{6a}$$

when $t_1 > 1$ and

$$\ln N - \sum_{t=2}^{t_2}\ln\left(1 - \frac{1}{t\ln t}\right) \leq \ln N + \ln\ln t_2 + 1.65 \tag{6b}$$

when $t_1 = 1$ (remember that $\ln\ln 1$ is understood to be 0). The constant 1.28 in (6a) is needed because t_1 and t_2 can take small values; e.g., if we only consider $t_1 \geq 10$, we can replace 1.28 by 0.05, and we can replace 1.28 by an arbitrarily small $\delta > 0$ if we only consider $t_1 \geq c$ for a sufficiently large c.

The dependence on t_2 in (6) is extremely mild. We can suppress it completely by increasing the dependence on t_1 just ever so slightly. If we set $\alpha_t = t^{-1-\epsilon}$, where $\epsilon > 0$, then the sum of the series $\sum_{t=1}^{\infty}\alpha_t$ is finite and the bound becomes

$$\ln(N-1) + (1+\epsilon)\ln t_1 + c_\epsilon \qquad \text{for } t_1 > 1, \text{ and} \tag{7a}$$

$$\ln N + c_\epsilon \qquad \text{for } t_1 = 1, \tag{7b}$$

where $c_\epsilon = -\sum_{t=2}^{\infty} \ln(1 - t^{-1-\epsilon})$. It is clear that the bound (7a) is far from optimal when t_1 is large: c_ϵ can be replaced by a quantity that tends to 0 as $O(t_1^{-\epsilon})$ as $t_1 \to \infty$. In particular, for $\epsilon = 1$ we have the bound

$$\ln N + 2 \ln t_1 + \ln 2.$$

An interesting feature of this switching rate is that for the full interval $[t_1, t_2] = [1, T]$ the bound differs from the standard AA bound only by an additive term less than 1. In words, the overhead for small adaptive regret is negligible.

4.2 Lower Bounds on Adaptive Regret

One may wonder how good this worst-case adaptive regret bound for Fixed Share is, if we compare to some other algorithm. We now argue that it cannot be improved. First we show an information-theoretic lower bound on the adaptive regret of any algorithm. Then we show that Fixed Share meets this bound.

Theorem 7. *Let $\phi(t_1, t_2, N)$ be the worst-case adaptive regret of any algorithm. Then for all T and for all N*

$$\sum_{m=1}^{T} \sum_{1=t_1<\ldots<t_{m+1}=T+1} N(N-1)^{m-1} e^{-\sum_{j=1}^{m} \phi(t_j, t_{j+1}-1, N)} \;\leq\; 1. \tag{8}$$

Proof. Fix an algorithm, time horizon T and expert count N. For any sequence $\boldsymbol{e} \in \{1, \ldots, N\}^T$ we define the loss pattern $(\ell_t^n)_{t\in[T]}^{n\in[N]}$ by

$$\ell_t^n \;=\; -\ln \mathbf{1}_{\{n=e_t\}}$$

(where $\mathbf{1}_{\{n=e_t\}} = 1$ if $n = e_t$ and $\mathbf{1}_{\{n=e_t\}} = 0$ otherwise). Let $L(\boldsymbol{e})$ be the loss of the algorithm on this loss pattern. Define the weight $w(\boldsymbol{e}) = e^{-L(\boldsymbol{e})}$. Clearly w is a probability distribution on $[N]^T$. Now let $t_2 < \ldots < t_m$ enumerate the internal block start indices $\big\{t \in \{2, \ldots, T\} \;\big|\; e_{t-1} \neq e_t\big\}$, and for the boundary set $t_1 = 1$ and $t_{m+1} = T + 1$. Since ϕ is the worst-case adaptive regret, and the loss of the best expert on each block is 0, we must have

$$L(\boldsymbol{e}) \;\leq\; \sum_{j=1}^{m} \phi(t_j, t_{j+1} - 1, N).$$

The theorem is obtained by negating and exponentiating this inequality, summing it over $[N]^T$, and grouping the contributions of sequences that agree on their block start indices. □

This bound is worthwhile because it is tight as we will see momentarily. It is however somewhat esoteric to interpret. It may be readily relaxed to imply for example that the bounds in (1) are tight, to a certain accuracy.

We will be interested in the performance guarantees that are *separable*, i.e., in upper bounds on $\phi(t_1, t_2, N)$ of the form $A(t_1) + B(t_2)$ (the number N of experts is fixed and omitted from our notation).

Corollary 8. *Suppose $\phi(t_1, t_2, N) \le A(t_1) + B(t_2)$ for all t_1 and t_2. Then for all T,*

$$\ln N - A(1) - B(T) + \sum_{t=2}^{T} \ln\left(1 + (N-1)e^{-A(t)-B(t-1)}\right) \;\le\; 0. \tag{9}$$

Proof. Substitute the constraint on ϕ into (8). □

The following corollary shows that the stronger form (5) of (1a) is essentially tight: we cannot improve the right-hand side of (5a) by a constant, even for large t_1 and t_2, unless (5b) is relaxed drastically (it is not sufficient to replace $\ln N$ by D and ignore all $t_2 < T_0$ for arbitrarily large D and T_0).

Corollary 9. *Fix the number of experts N, a constant $C < \ln(N-1)$, and arbitrarily large positive integer constants D and T_0. No algorithm has worst-case adaptive regret*

$$\phi(t_1, t_2, N) \;\le\; C + \ln t_2 + \infty 1_{\{t_2 < T_0\}} + D 1_{\{t_1 = 1\}} + \infty 1_{\{1 < t_1 \le T_0\}}. \tag{10}$$

Proof. Setting

$$A(t) = \begin{cases} D & \text{if } t = 1 \\ 0 & \text{if } t > T_0 \\ \infty & \text{otherwise} \end{cases} \quad \text{and} \quad B(t) = \begin{cases} \ln t + C & \text{if } t \ge T_0 \\ \infty & \text{otherwise} \end{cases}$$

on the right-hand side of (9) we obtain

$$\ln N - D - \ln T - C + \sum_{t=T_0+1}^{T} \ln\left(1 + (N-1)\frac{e^{-C}}{t-1}\right) \ge -\ln T + \frac{N-1}{e^C}\ln T - O(1)$$

which tends to ∞ as $T \to \infty$ (the inequality follows from the inequality $\ln(1+x) \ge x - x^2$, where $x \ge -1/2$). This contradicts (9). □

Our next corollary is about the tightness of (1b) and its elaboration (6) (see also the discussion following (6)).

Corollary 10. *Fix the number of experts N, a constant $C < \ln(N-1)$, and positive integer D and T_0. No algorithm has worst-case adaptive regret*

$$\phi(t_1, t_2, N) \;\le\; C + \ln t_1 + \ln\ln t_2 + \infty 1_{\{t_2 < T_0\}} + D 1_{\{t_1 = 1\}} + \infty 1_{\{1 < t_1 \le T_0\}}. \tag{11}$$

Proof. Setting

$$A(t) = \begin{cases} D & \text{if } t = 1 \\ \ln t & \text{if } t > T_0 \\ \infty & \text{otherwise} \end{cases} \quad \text{and} \quad B(t) = \begin{cases} \ln\ln t + C & \text{if } t \ge T_0 \\ \infty & \text{otherwise} \end{cases}$$

on the right-hand side of (9) we now have

$$\begin{aligned}\ln N - D - \ln\ln T - C + \sum_{t=T_0+1}^{T} \ln\left(1 + (N-1)\frac{e^{-C}}{t\ln(t-1)}\right) \\ \geq -\ln\ln T + \frac{N-1}{e^C}\ln\ln(T-1) - O(1) \to \infty \qquad (T \to \infty). \qquad \square\end{aligned}$$

Finally, we explore the tightness of (1c) (and its elaboration given later in the paper: see (7) and the discussion afterwards).

Corollary 11. *Fix the number of experts N, a constant $\epsilon > 0$, and a constant $a < \sum_{t=2}^{\infty} \ln(1 + t^{-1-\epsilon})$. No algorithm has worst-case adaptive regret*

$$\phi(t_1, t_2, N) \;\leq\; \begin{cases} \ln N + a & \textit{if } t_1 = 1 \\ \ln(N-1) + (1+\epsilon)\ln t_1 & \textit{otherwise.} \end{cases} \tag{12}$$

Proof. Setting

$$A(t) = \begin{cases} \ln N + a & \text{if } t = 1 \\ \ln(N-1) + (1+\epsilon)\ln t & \text{otherwise} \end{cases}$$

and $B(t) = 0$ on the right-hand side of (9) now gives

$$\ln N - \ln N - a + \sum_{t=2}^{T} \ln\left(1 + (N-1)e^{-\ln(N-1) - (1+\epsilon)\ln t}\right) > 0$$

for a sufficiently large T. □

4.3 Fixed Share Has Optimal Adaptive Worst-Case Regret

We now prove that Fixed Share is optimal, in the sense that no algorithm can have a worst-case adaptive regret that is nowhere worse.

Corollary 12. *Fix any switching rate $(\alpha_t)_{t\geq 1}$, and let $\phi(t_1, t_2, N)$ be the worst-case adaptive regret of FS. Then (8) holds with equality.*

Proof. Plug the worst-case adaptive regret (4) into the sum (8). □

5 Conclusion

We examined the problem of guaranteeing small adaptive regret for the setting of prediction with expert advice. In the first part we considered two techniques to obtain adaptive algorithms: using virtual specialist experts and restarting classical algorithms. We showed that both can be viewed as Fixed Share with a variable switching rate. In the second part we computed the exact worst-case adaptive regret for Fixed Share, thus tightening the existing upper bounds. So much, in fact, that by summing these worst-case regrets over a partition of the interval $[1, T]$ we recover the standard Fixed Share tracking bound. This formally establishes the complete congruence between adaptive and tracking performance, which was intuitive but not apparent from previously obtained adaptive bounds.

We then showed that Fixed Share is Pareto-optimal, in the sense that no algorithm can ensure better adaptive regret. We presented an information-theoretic lower bound on the worst-case adaptive regret of any algorithm, and showed that it holds with equality for Fixed Share.

Open problem. Whereas upper bounds readily transfer to mixable losses, obtaining adaptive regret lower bounds for mixable losses is much more tricky. It is fair to call the lower bound argument in [15] for classical regret complicated, and this would be a special case for adaptive regret lower bounds.

Acknowledgments. First author supported by Veterinary Laboratories Agency (VLA) of Department for Environment, Food and Rural Affairs (Defra). Second author supported by NWO Rubicon grant 680-50-1010. Third author supported by EPSRC grant EP/I030328/1.

References

[1] Bousquet, O., Warmuth, M.K.: Tracking a small set of experts by mixing past posteriors. Journal of Machine Learning Research 3, 363–396 (2002)

[2] Cesa-Bianchi, N., Gaillard, P., Lugosi, G., Stoltz, G.: A new look at shifting regret. CoRR abs/1202.3323 (2012)

[3] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning, and Games. Cambridge University Press (2006)

[4] Chernov, A., Vovk, V.: Prediction with Expert Evaluators' Advice. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 8–22. Springer, Heidelberg (2009)

[5] Freund, Y., Schapire, R.E., Singer, Y., Warmuth, M.K.: Using and combining predictors that specialize. In: Proc. 29th Annual ACM Symposium on Theory of Computing, pp. 334–343. ACM (1997)

[6] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139 (1997)

[7] Hazan, E., Seshadhri, C.: Efficient learning algorithms for changing environments. In: ICML, p. 50 (2009)

[8] Herbster, M., Warmuth, M.K.: Tracking the best expert. Machine Learning 32(2), 151–178 (1998)

[9] Koolen, W.M.: Combining Strategies Efficiently: High-quality Decisions from Conflicting Advice. Ph.D. thesis, Institute of Logic, Language and Computation (ILLC), University of Amsterdam (January 2011)

[10] Koolen, W.M., de Rooij, S.: Combining expert advice efficiently. In: Servedio, R., Zang, T. (eds.) Proceedings of the 21st Annual Conference on Learning Theory (COLT 2008), pp. 275–286 (June 2008)

[11] Littlestone, N., Warmuth, M.K.: The weighted majority algorithm. Inf. Comput. 108(2), 212–261 (1994)

[12] Shamir, G.I., Merhav, N.: Low complexity sequential lossless coding for piecewise stationary memoryless sources. IEEE Trans. Info. Theory 45, 1498–1519 (1999)

[13] Vovk, V.: Aggregating strategies. In: Proceedings of the Third Annual Workshop on Computational Learning Theory, pp. 371–383. Morgan Kaufmann (1990)

[14] Vovk, V.: Competitive on-line statistics. International Statistical Review 69, 213–248 (2001)

[15] Vovk, V.: A game of prediction with expert advice. Journal of Computer and System Sciences 56, 153–173 (1998)

[16] Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: Proc. 20th Int. Conference on Machine Learning (ICML 2003), pp. 928–936 (2003)

A Worst-Case Adaptive Regret Data for Fixed Share

In this subsection we prove that the worst-case data for Fixed Share has the following form. On the interval $[t_1, t_2]$ we are interested in all but one expert suffer infinite loss and on the step preceding t_1 (if $t_1 \neq 1$) this one expert suffers infinite loss himself. The construction is iterative and we start constructing the data from the end of the interval.

Lemma 13. *For any history prior to the step t_2 the adaptive regret $R^j_{[t_1,t_2]}$ w.r.t. expert j on the interval $[t_1, t_2]$ is maximised with $\ell^k_{t_2} = \infty$ for $k \neq j$.*

Proof. Let us differentiate the adaptive regret w.r.t. $\ell^k_{t_2}$:

$$\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^k_{t_2}} = \frac{u^k_{t_2} e^{-\ell^k_{t_2}}}{\sum u^i_{t_2} e^{-\ell^i_{t_2}}} - \mathbf{1}_{\{j=k\}}$$

□

We can see that it is positive for all $k \neq j$ and becomes zero for $k = j$ when we plug in $\ell^k_{t_2} = \infty$ for those.

Lemma 14. *Fix an comparator expert j. Let $t \in [t_1, t_2]$. Suppose that the losses for steps $s = t+1, \ldots, t_2$ satisfy $\ell^k_s = \infty$ for $k \neq j$. Then the adaptive regret $R^j_{[t_1,t_2]}$ is maximised with $\ell^k_t = \infty$ for $k \neq j$.*

Proof. Let us start with showing that the if on the steps $t+1$ and $t+2$ the data is organised as we want to, that is j-th expert is good and all others suffer infinite loss, then Learner's loss on step $t+2$ is not dependent on what happens at time t and before. This follows immediately from (3), as

$$\ell_{t+2} = -\ln(1 - \alpha_{t+2}) .$$

Now let us differentiate the adaptive regret $R^j_{[t_1,t_2]}$ w.r.t. ℓ^k_t assuming that the future losses are set up as we want. Let us show that the derivatives w.r.t. ℓ^k_t where $k \neq j$ are all positive. For those,

$$\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^k_t} = \frac{\partial \ell_t}{\partial \ell^k_t} + \frac{\partial \ell_{t+1}}{\partial \ell^k_t}$$

Expanding the second one gives (as before, $k \neq j$):

$$\begin{aligned}\frac{\partial \ell_{t+1}}{\partial \ell^k_t} &= \frac{\partial}{\partial \ell^k_t} - \ln\left(\frac{\alpha_{t+1}}{N-1} + (1 - \frac{N}{N-1}\alpha_{t+1})u^j_t e^{\ell_t - \ell^j_t}\right)\\ &= -\frac{(1-\frac{N}{N-1}\alpha_{t+1})u^j_t e^{\ell_t-\ell^j_t}\frac{\partial}{\partial \ell^k_t}\ell_t}{\frac{\alpha_{t+1}}{N-1} + (1-\frac{N}{N-1}\alpha_{t+1})u^j_t e^{\ell_t-\ell^j_t}}\end{aligned}$$

So we see that $\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^k_t} = \frac{\partial \ell_t}{\partial \ell^k_t}\left(\frac{\frac{\alpha_{t+1}}{N-1}}{\frac{\alpha_{t+1}}{N-1}+(1-\frac{N}{N-1}\alpha_{t+1})u^j_t e^{\ell_t-\ell^j_t}}\right) > 0$. So our worst-case pattern of losses extends one trial backwards. □

Finally, we need to state the almost obvious fact that in order to maximise the adaptive regret we need to insert an infinite loss for the comparator expert right before the start of the interval, thus killing all the previous weight on him.

Lemma 15. *Fix a comparator expert j. Suppose that the losses for steps $s = t_1, \ldots, t_2$ satisfy $\ell^k_s = \infty$ for $k \neq j$. Then the adaptive regret $R^j_{[t_1,t_2]}$ is maximised with $\ell^j_{t-1} = \infty$.*

Proof. As before, the adaptive regret on steps starting from $t_1 + 1$ does not depend on $\ell^k_{t_1-1}$. So let us show that $\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^j_{t_1-1}} > 0$. We can reuse the proofs of previous lemmas for that:

$$\begin{aligned}\frac{\partial R^j_{[t_1,t_2]}}{\partial \ell^j_{t_1-1}} &= \frac{\partial \ell_{t_1}}{\partial \ell^j_{t_1-1}}\\ &= -\frac{(1-\frac{N}{N-1}\alpha_{t_1})u^j_{t_1-1}e^{\ell_{t_1-1}-\ell^j_{t_1-1}}}{\frac{\alpha_{t_1}}{N-1}+(1-\frac{N}{N-1}\alpha_{t_1})u^j_{t_1-1}e^{\ell_{t_1-1}-\ell^j_{t_1-1}}}\frac{\partial\left(\ell_{t_1-1}-\ell^j_{t_1-1}\right)}{\partial \ell^j_{t_1-1}} > 0,\end{aligned}$$

since $\frac{\partial\left(\ell_{t_1-1}-\ell^j_{t_1-1}\right)}{\partial \ell^j_{t_1-1}}$ is negative as follows from the proof of Lemma 13. □

Partial Monitoring with Side Information

Gábor Bartók and Csaba Szepesvári

University of Alberta
Edmonton, Canada

Abstract. In a partial-monitoring problem in every round a learner chooses an action, simultaneously an opponent chooses an outcome, then the learner suffers some loss and receives some feedback. The goal of the learner is to minimize his (unobserved) cumulative loss. In this paper we explore a variant of this problem where in every round, before the learner makes his decision, he receives some side-information. We assume that the outcomes are generated randomly from a distribution that is influenced by the side-information. We present a "meta" algorithm scheme that reduces the problem to that of the construction of an algorithm that is able to estimate the distributions of observations while producing confidence bounds for these estimates. Two specific examples are shown for such estimators: One uses linear estimates, the other uses multinomial logistic regression. In both cases the resulting algorithm is shown to achieve $\widetilde{O}(\sqrt{T})$ minimax regret for locally observable partial-monitoring games.

1 Introduction

Partial monitoring is a framework to model online learning games with arbitrary feedback structure. In every time step, a learner chooses an *action* and simultaneously an opponent chooses an *outcome*. Then, the learner suffers some loss and receives some feedback, both of which are deterministic functions of the action and the outcome. The loss and feedback functions are both known to the learner and the opponent and together they define the *partial monitoring game*. The goal of the learner is to keep his cumulative loss as low as possible. His performance is measured in terms of the *regret*: the learner's excess cumulative loss compared to that of the best fixed action in hindsight.

Canonical examples of partial-monitoring include *product testing* and *dynamic pricing*. In the case of product testing, the learner has to decide to test or not test products arriving on a production line. The learner receives feedback about the quality of the product only if he decided to test the product. On the other hand, he suffers a constant loss in every time step when either a good product was tested (unable to sell, e.g., when the test means the destruction of the product) or a bad product was not tested (complaining costumers). In the case of dynamic pricing, a vendor (learner/he) sets the price of a product while the consumer (opponent/she) secretly chooses a maximum price she is willing to buy the product for. In case the sale price is below the consumer-chosen price,

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 305–319, 2012.

the product is sold. The information received by the learner is the single bit whether this happens. The loss suffered in a round when the product is sold is the difference between the consumer-chosen prices and the sale price, while in a round when the product is not sold a fixed storage cost is incurred.

In this paper we extend the basic partial monitoring problem to allow the learner to use some *side information* to make a more informed decision. For example, in product testing, before deciding about whether to use a potentially destructive testing procedure the learner can take a look at the product. Similarly, in dynamic pricing, the learner may use information available about the customer (gender, age, etc.) for determining a more competitive price. Formally, the assumption is that in each round the learner receives the so-called side information (sometimes also called "a context") before making a decision. The side information is not subject to any restrictions, but in this paper we assume that the outcome for the given round is a *stochastically* function of the side information shown to the learner. Then, instead of competing with the single best action, the learner competes with the oracle that knows the mapping that maps the side information to the outcome distributions and who makes optimal decisions given this knowledge.

1.1 Related Work

The model of partial monitoring was introduced by Piccolboni and Schindelhauer [2001]. They designed the algorithm FeedExp and showed for any game, either the worst-case expected regret is linear in the time horizon T, or the algorithm achieves expected regret of $O(T^{3/4})$ for any outcome sequence. This upper bound was later improved to $O(T^{2/3})$ by Cesa-Bianchi et al. [2006]. In the same paper, Cesa-Bianchi et al. show that there exists a game whose *minimax regret*—the worst case regret of the best possible algorithm—scales as $\Omega(T^{2/3})$. However, they noted that some games enjoy minimax regret growth rate of $\Theta(\sqrt{T})$, and posed the problem of determining exactly which games have minimax regret rate better than $\Theta(T^{2/3})$. This problem was solved in the works of Bartók et al. [2011] against stochastic opponents, while by providing a new algorithm Foster and Rakhlin [2011] showed that the classification of games worked out by Bartók et al. [2011] continues to hold even against adversarial opponents. According to the solution, partial-monitoring games with a finite number of actions and outcomes can be classified into four categories based on the growth rate of the minimax regret: *trivial* games with minimax regret 0, *easy* games with minimax regret[1] of $\widetilde{\Theta}(\sqrt{T})$, *hard* games with minimax regret $\Theta(T^{2/3})$, and *hopeless* games with linear minimax regret. The condition that separates easy games from hard games is the *local observability condition* (see Definition 2). In the bandit literature learning with side-information has been considered before under various conditions, see Auer [2003]; Dudík et al. [2011] and references therein, while Helmbold et al. [2000] considered a special case of our framework when both the number of actions and outcomes is two, with one action revealing

[1] The notations $\widetilde{O}(\cdot)$ and $\widetilde{\Theta}(\cdot)$ hide polylogarithmic terms.

the actual outcome, while the other action not yielding any information about the outcome, the hidden relationship between the side information and hidden information is deterministic and the loss is the zero-one loss.

2 Problem Definition

An instance of a partial-monitoring game with side-information is described by the tuple $\mathbf{G} = (\mathbf{L}, \mathbf{H}, \mathcal{F})$, where $\mathbf{L} \in \mathbb{R}^{N \times M}$ is the *loss matrix*, $\mathbf{H} \in \Sigma^{N \times M}$ is the *feedback matrix* (Σ is the set of feedback symbols), and $\mathcal{F} \subseteq \{f \mid f : \mathcal{X} \to \Delta_M\}$ is a subset of all functions that map elements from some side-information set $\mathcal{X}$ to the set of outcome distributions. For convenience, we assume that $\max_{i \in \underline{N}, j \in \underline{M}}(\mathbf{L}_{i,j}) - \min_{i \in \underline{N}, j \in \underline{M}}(\mathbf{L}_{i,j}) \leq 1$, where for a natural number $n \in \mathbb{N}$ we used $\underline{n}$ to denote the set $\{1, 2, \ldots, n\}$. The partial-monitoring game proceeds in turns. Before the first turn, both the learner and the opponent is given $\mathbf{G}$ and the opponent secretly chooses a function $f \in \mathcal{F}$. In turn t ($t = 1, 2, \ldots$), first the learner receives the side-information $x_t \in \mathcal{X}$. Then, the learner chooses an action $I_t \in \underline{N}$, while at the same time the opponent draws an outcome J_t from the distribution $f(x_t)$. No stochastic assumption is made about the side information sequence, $\{x_t\}$ and, in fact, we also allow x_t to be chosen based on the history $\mathcal{H}_{t-1} = (x_1, I_1, J_1, \ldots, x_{t-1}, I_{t-1}, J_{t-1})$. After the learner and the opponent made their decisions, the learner receives the feedback $\mathbf{H}_{I_t,J_t}$ and suffers the loss $\mathbf{L}_{I_t,J_t}$. It is important to emphasize that the loss is not revealed to the learner.

The goal of the learner is to minimize his cumulative loss given the knowledge of the game $\mathbf{G}$. His performance is measured in terms of the regret, defined as the excess cumulative loss he suffers as compared to the expected cumulative loss of the oracle that knows f and chooses the action with the smallest expected loss as a function of the side-information in every round. In other words,

$$R_T = \sum_{t=1}^{T} \mathbf{L}_{I_t,J_t} - \min_{g \in \underline{N}^{\mathcal{X}}} \sum_{t=1}^{T} \mathbb{E}[\mathbf{L}_{g(x_t),J_t} | \mathcal{H}_{t-1}, x_t] \,.$$

3 Preliminaries

In this section we introduce the necessary notations and definitions that we will need. Most of the definitions presented here are taken from Bartók et al. [2011].

Let $\mathbf{G} = (\mathbf{L}, \mathbf{H}, \mathcal{F})$ be a partial-monitoring game. For an action i, the column vector ℓ_i consisting of the elements of the i^{th} row of $\mathbf{L}$ is called the *loss vector* of action i. Let the probability simplex of dimension n be denoted by $\mathcal{K}_n \subseteq \mathbb{R}^n$. Thus, the set of all outcome distributions is $\mathcal{K}_M$. It is easy to see that the expected loss of action i at time step t given the past and x_t equals $\mathbb{E}[\mathbf{L}_{i,J_t} | H_{t-1}, x_t] = \ell_i^\top f(x_t)$.

For an action i, let the cell of i be the set of outcome distributions under which action i is optimal:

$$\mathcal{C}_i = \{p \in \mathcal{K}_M \mid \forall j \in \underline{N} : (\ell_i - \ell_j)^\top p \leq 0\} \,.$$

It is easy to see that for every $i \in \underline{N}$, $\mathcal{C}_i$ is either empty or a closed convex polytope, with $\bigcup_{i \in \underline{N}} \mathcal{C}_i = \mathcal{K}_M$. We call $\mathbb{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N\}$ the *cell decomposition* of $\mathcal{K}_M$. For clarity of presentation, in this paper we only deal with games that are non-degenerate: for every action i, $\mathcal{C}_i$ is $M-1$ dimensional and for $i \neq j$, $\mathcal{C}_i \neq \mathcal{C}_j$. We remark that our results generalize to degenerate games, but the algorithm and its analysis are somewhat more involved.

For an action $i \in \underline{N}$, we define the *signal matrix* of i as follows:

Definition 1. *For an action i, let $\alpha_1, \alpha_2, \ldots, \alpha_{\sigma_i} \in \Sigma$ be the distinct symbols in the i^{th} row of the feedback matrix $\mathbf{H}$. The signal matrix $S_i \in \{0,1\}^{\sigma_i \times M}$ is defined as the incidence matrix of the i^{th} row of the feedback matrix $\mathbf{H}$:*

$$(S_i)_{k,l} = \mathbb{I}_{\{H_{i,l} = \alpha_k\}} \, .$$

An important property of the signal matrix S_i is that if $p \in \mathcal{K}_M$ is the outcome distribution chosen by the opponent then $S_i p$ is the probability distribution over the set of observations $\{\alpha_1, \ldots, \alpha_k\}$ induced by p under action i. From now on, without loss of generality, we assume that the feedback at time step t is presented as the unit vector corresponding to the received symbol $\mathbf{H}_{I_t, J_t}$. We shall denote this unit vector by Y_t.

If for two actions i and j, $\dim(\mathcal{C}_i \cap \mathcal{C}_j) = M - 2$ we say that i and j are *neighbors*. The set of neighboring action pairs is denoted by $\mathcal{N}$. Now we are ready to recall the *local observability condition* from Bartók et al. [2011]:

Definition 2. *Let $\{i,j\} \in \mathcal{N}$ be two neighboring actions. We say that $\{i,j\}$ is* locally observable *if $\ell_i - \ell_j \in \mathrm{Im}(S_i^\top) \oplus \mathrm{Im}(S_j^\top)$. The game is called* locally observable *(or we say that it satisfies the local observability condition) if every neighboring action pair is locally observable. For a pair of distinct action $\{i,j\} \in \mathcal{N}$, a pair of vectors, $v_{i,j}$, $v_{j,i}$ is called* observer vectors *for $\{i,j\}$ if*

$$\ell_i - \ell_j = S_i^\top v_{i,j} - S_j^\top v_{j,i} \, .$$

If a neighboring action pair is locally observable then the local observability condition yields the existence of these observer vectors. From now on, for locally observable neighboring action pairs we shall fix such observer vectors. Note that the observer vectors are *not* uniquely defined. We will discuss good choices of the observer vectors later on.

4 The Algorithm

Bartók et al. [2011] proved that if a game is locally observable then a minimax regret of $\widetilde{O}(\sqrt{T})$ is achievable against a stochastic opponent. Now we extend their result to partial monitoring with side-information. In particular, we show that the $\widetilde{O}(\sqrt{T})$ regret bound remains true in this richer model.

In this section we describe the algorithm scheme CBP-Side for "Confidence Bound Partial monitoring with Side-information" that when fed with a method that estimates the outcome distributions and their uncertainty defines a learning strategy. In Section 5 we give a bound on the expected regret as a function of how fast the uncertainty of the outcome distribution estimates decays. Then, in Section 6 we present two examples that illustrate how this general bound translates into actual regret bounds for two different classes of functions $\mathcal{F}$.

The algorithm is a generalization of the algorithm "Confidence Bound Partial monitoring" (CBP) from Bartók et al. [2012]. Pseudocode for the algorithm is given in Algorithm 1.

Throughout the algorithm, some statistics $\mathcal{S}$ is maintained that is used by the functions getObsEst and getConfWidth (which are left generic for now). The statistics might be the whole sequence of observations and actions up to time step $t-1$, or just some average of the observations and maybe the number of times each action was chosen. After receiving the side-information for time step t, estimates for the observation probabilities and their confidence widths are obtained by calling the functions getObsEst and getConfWidth. Then the algorithm calculates estimates of the loss differences (denoted by $\tilde{\Delta}_{i,j}$) for neighboring action pairs, along with their confidence widths $c_{i,j}$. If, for some pair $i, j \in \mathcal{N}$ the absolute value of the loss-difference estimate is greater than its confidence width, we know that, with high probability, $p_t = f(x_t)$ lies in the half space $\{p \in \mathbb{R}^M \mid \operatorname{sgn}(\tilde{\Delta}_{i,j})(\ell_i - \ell_j)^\top p \geq 0\}$. Thus, the intersection of all these half spaces and the probability simplex determines the convex polytope $\mathcal{K}_t$ that p_t belongs to (with high probability), giving rise to the set of admissible actions Q. To compute this set the method getNeighbors computes $\mathcal{N}(t) = \{\{i,j\} \in \mathcal{N} : \mathcal{C}_i \cap \mathcal{C}_j \cap \operatorname{int}(K_t) \neq \emptyset\}$. Then, $Q = \cup\mathcal{N}(t)$. Finally, the action I_t from Q that has the greatest potential of reducing the confidence width for the next rounds is chosen and based on the information received the statistics $\mathcal{S}$ is updated.

5 Analysis of CBP-Side

In this section we provide an upper bound on the expected regret suffered by the algorithm on any given game with any plugged-in estimate and confidence width functions. Note that the upper bound contains the expectation of some random values that depend on the outcomes drawn randomly at every time step. In the next sections, we will see how these can be upper-bounded by some (small) deterministic quantities in some specific cases.

From now on, we use the convention that for any variable v, we denote by $v(t)$ the value assigned to v in time step t.

Theorem 1. *Assume that there exist numbers $\delta_1, \delta_2, \ldots, \delta_T \in [0,1]$ and a norm $\|\cdot\|$ such that for every time step t it holds that*

$$\mathbb{P}\left(\|\hat{q}_i(t) - S_i f(x_t)\| > w_i(t)\right) \leq \delta_t \tag{1}$$

for every $i \in \underline{N}$. Then, the expected regret of CBP-Side *on game $\mathbf{G} = (\mathbf{L}, \mathbf{H}, \mathcal{F})$ can be upper bounded as*

$$\mathbb{E}[R_T] \leq \sum_{t=1}^{T} N\delta_t + \sum_{t=1}^{T} \mathbb{E}\left[\min\left\{4NW_{I_t} w_{I_t}(t), 1\right\}\right],$$

Algorithm 1. The algorithm CBP-Side

```
Input: $\mathbf{L}$, $\mathbf{H}$, $\alpha$
Calculate $\mathcal{P}$, $\mathcal{N}$, $v_{i,j}$, $W_k$
$\mathcal{S} \leftarrow$initStatistic()                      {Some statistics as needed}
for $t = 1$ to $T$ do
    Receive side information $x_t$
    for each $i \in \underline{N}$ do
        $\tilde{q}_i \leftarrow$getObsEst($\mathcal{S}, x_t$)    {Observation distribution estimate}
        $w_i \leftarrow$getConfWidth($\mathcal{S}, x_t$)         {Confidence}
    end for
    for each $\{i,j\} \in \mathcal{N}$ do
        $\tilde{\Delta}_{i,j} \leftarrow v_{i,j}^\top \tilde{q}_i - v_{j,i}^\top \tilde{q}_j$      {Loss diff. estimate}
        $c_{i,j} \leftarrow \|v_{i,j}\|_* w_i + \|v_{j,i}\|_* w_j$     {Confidence}
        if $|\tilde{\Delta}_{i,j}| \geq c_{i,j}$ then
            $halfSpace(i,j) \leftarrow \operatorname{sgn} \tilde{\Delta}_{i,j}$
        else
            $halfSpace(i,j) \leftarrow 0$
        end if
    end for
    $\mathcal{N}(t) \leftarrow$ getNeighbors($\mathcal{P}, \mathcal{N}, halfSpace$)
    $Q \leftarrow \bigcup \mathcal{N}(t)$                       {Admissible actions}
    Choose $I_t = \operatorname{argmax}_{i \in Q}(W_i w_i)$     {$W_i = \max_j \|v_{i,j}\|_*$}
    Observe $Y_t$
    $\mathcal{S} \leftarrow$updateStatistic($\mathcal{S}, x_t, I_t, Y_t$)
end for
```

where $W_i = \max_j \|v_{i,j}\|_*$ *with* $\|\cdot\|_*$ *being the dual norm of* $\|\cdot\|$.

Proof. For any $i, j \in \underline{N}$ and $x \in \mathcal{X}$, let $\Delta_{i,j}(x)$ denote the expected loss difference of actions i and j given side-information x, written as $\Delta_{i,j}(x) \overset{\triangle}{=} (\ell_i - \ell_j)^\top f(x)$. Further, let $\Delta_i(x) \overset{\triangle}{=} \max_j \Delta_{i,j}(x)$ be the "gap" between the expected loss of action i and that of an optimal action given side-information x. It is easy to see that the expected regret of an algorithm can be rewritten as $\mathbb{E}[R_T] = \sum_{t=1}^T \mathbb{E}[\Delta_{I_t}(x_t)]$. Let $\mathcal{E}_t$ be the event that some confidence width fails at time step t. Then, $\mathbb{E}[R_T] = \sum_{t=1}^T \mathbb{E}[\Delta_{I_t}(x_t)] \leq \sum_{t=1}^T N\delta_t + \sum_{t=1}^T \mathbb{E}[\Delta_{I_t}(x_t)\mathbb{I}_{\{\mathcal{E}_t^c\}}]$, where we used that $\Delta_i(x) \leq 1$. Thus, it remains to bound $\Delta_{I_t}(x_t)$ assuming that for all $i \in \underline{N}$, $\|\tilde{q}_i(t) - S_i f(x_t)\| \leq w_i(t)$ holds.

If i and j are in $\mathcal{N}(t)$ (that is, they are neighbors at time step t), then $\tilde{\Delta}_{i,j}(t)$ is a "good" approximation of $\Delta_{i,j}(x_t)$:

$$\begin{aligned} |\Delta_{i,j}(x_t) - \tilde{\Delta}_{i,j}(t)| &= \left|(\ell_i - \ell_j)^\top f(x_t) - \left(v_{i,j}^\top \tilde{q}_i(t) - v_{j,i}^\top \tilde{q}_j(t)\right)\right| \\ &\leq \|v_{i,j}\|_* \|S_i f(x_t) - \tilde{q}_i(t)\| + \|v_{j,i}\|_* \|S_j f(x_t) - \tilde{q}_j(t)\| \\ &\leq \|v_{i,j}\|_* w_i(t) + \|v_{j,i}\|_* w_j(t) \\ &= c_{i,j}(t) \, . \end{aligned} \tag{2}$$

We know from line 12 of the algorithm that if $\{i,j\} \in \mathcal{N}(t)$ then $\tilde{\Delta}_{i,j}(t) \le c_{i,j}$. This together with Equation (2) gives

$$\Delta_{i,j}(x_t) \le 2c_{i,j}\,. \tag{3}$$

Let i^* be an optimal action at time step t (that is, $\min_i \ell_i^\top f(x_t) = \ell_{i^*}^\top f(x_t)$). Then

$$\Delta_{I_t,i^*}(t) = \sum_{s=1}^{r} \Delta_{k_{s-1},k_s}(t)\,,$$

where $I_t = k_0, k_1, \ldots, k_r = i^*$ is a sequence of actions such that $\{k_{s-1}, k_s\} \in \mathcal{N}(t)$ for all $1 \le s \le r$. This sequence always exists thanks to how the algorithm constructs the set of admissible actions[2]. With the help of Equation (3) we get

$$\begin{aligned}\Delta_{I_t,i^*}(t) &\le 2\sum_{s=1}^{r} c_{k_{s-1},k_s}(t) = 2\sum_{s=1}^{r}\Big(\left\|v_{k_{s-1},k_s}\right\|_q w_{k_{s-1}}(t) + \left\|v_{k_s,k_{s-1}}\right\|_q w_{k_s}(t)\Big)\\ &\le 4NW_{I_t}w_{I_t}(t)\,,\end{aligned}$$

where in the last line we used line 20 of the algorithm and the fact that $r \le N$, thus finishing the proof. □

Remark 1 (On the choice of the observer vectors $v_{i,j}$.). We mentioned earlier that the choice of the observer vectors is not unique and thus we have some freedom in choosing them. Theorem 1 indicates that for different estimators, the best choice of the observer vectors might differ. In particular, it depends on the norm the estimate uses: to optimize the bound of Theorem 1, we should choose the vectors that minimize $\|v_{i,j}\|_*$. If the norm used is the 2-norm then there is a closed form solution for the best $v_{i,j}$:

$$\begin{pmatrix} v_{i,j} \\ -v_{j,i} \end{pmatrix} = \left(S_i^\top\ S_j^\top\right)^+ (\ell_i - \ell_j)\,,$$

where A^+ denotes the pseudo-inverse of the matrix A.

6 Examples

In this section we demonstrate the power of Theorem 1 through specific examples.

6.1 Linear Side-Information, Least-Squares Estimate

In the first example, the side-information set is the probability simplex $\mathcal{K}_d$ of some dimension $d > 0$ while the function set $\mathcal{F}$ is the set of all linear maps where the underlying matrix is a stochastic matrix of size $M \times d$. The estimator we

[2] For a thorough proof of this statement, we refer the reader to Bartók et al. [2012].

use is regularized least squares. We introduce the following notations. For every action i, let $\theta_i^* = S_i K \in \mathbb{R}^{\sigma_i \times d}$, where K is the matrix underlying the the linear map f chosen by the opponent (thus, $f(x) = Kx$). Let $t_i(s)$ be the time step when action i is chosen by the algorithm the s^{th} time. Let $n_i(t)$ be the number of times action i is chosen up to time step t. Then the regularized least squares estimator is defined by the equation

$$\tilde{\theta}_i(t) = \min_{\theta \in \mathbb{R}^{M \times d}} \sum_{s=1}^{n_i(t-1)} \left(Y_{t_i(s)} - \theta x_{t_i(s)}\right)^2 + \lambda_i \|\theta\|_2^2 .$$

For the closed form solution we define the matrices

$$X_{i,t} = \left(x_{t_i(1)}\ x_{t_i(2)}\ \cdots\ x_{t_i(n_i(t-1))}\right), \quad \mathcal{Y}_{i,t} = \left(Y_{t_i(1)}\ Y_{t_i(2)}\ \cdots\ Y_{t_i(n_i(t-1))}\right) .$$

Then,

$$\tilde{\theta}_i(t) = \mathcal{Y}_{i,t} X_{i,t}^\top \left(\lambda_i I_d + X_{i,t} X_{i,t}^\top\right)^{-1} ,$$

where I_d is the $d \times d$ identity matrix. Let $V_{i,t} = \lambda_i I_d + X_{i,t} X_{i,t}^\top$.

For some positive definite matrix S, let $\|\cdot\|_S$ denote the S-weighted 2-norm: $\|v\|_S^2 = v^\top S v$. In the rest of the paper, we will need a number of results, which, for the sake of completeness, we recite here.

Theorem 2 (Abbasi-Yadkori et al. [2011, Theorem 1]). *Let $\{F_t\}_{t=1}^\infty$ a filtration. Let $\{\eta_t\}_{t=1}^T$ be a real-valued stochastic process such that η_t is F_t-measurable and η_t is conditionally R-sub-Gaussian for some $R \geq 0$. Let $\{x_t\}_{t=1}^\infty$ be an $\mathbb{R}^d$-valued stochastic process such that x_t is F_{t-1}-measurable. Let $\lambda > 0$. For any $t \geq 0$, define*

$$V_t = \lambda I + \sum_{s=1}^t x_s x_s^\top, \qquad S_t = \sum_{s=1}^t \eta_s x_s .$$

Then, for any $\delta > 0$, with probability at least $1 - \delta$, for all $t \geq 0$,

$$\|S_t\|_{V_t^{-1}}^2 \leq 2R^2 \log \left(\frac{\det(V_t)^{1/2} \det(\lambda I)^{-1/2}}{\delta} \right) .$$

Theorem 3 (Abbasi-Yadkori and Szepesvári [2011, Theorem 1]). *Let $(x_0, Y_1), \ldots, (x_t, Y_{t+1}), x_i \in \mathbb{R}^d, Y_i \in \mathbb{R}^n$ satisfy the linear model Assumption*[3] *A1 with some $L > 0$, $\Theta_* \in \mathbb{R}^{d \times n}$, $\operatorname{tr}(\Theta_*^\top \Theta_*) \leq S^2$ and let $\mathcal{F} = (\mathcal{F}_t)$ be the associated filtration. Consider the ℓ^2-regularized least-squares parameter estimate $\hat{\Theta}_t$ with regularization coefficient $\lambda > 0$. Let*

$$V_t = \lambda I + \sum_{i=0}^{t-1} x_i x_i^\top$$

[3] Reciting this assumption is beyond the scope of this paper. In a nutshell, it says that x_t and Y_t are $\mathcal{F}_t$-measurable, $\mathbb{E}[Y_{t+1}|\mathcal{F}_t] = \Theta^\top x_t$ for some matrix Θ, the noise $Y_{t+1} - \mathbb{E}[Y_{t+1}|\mathcal{F}_t]$ is componentwise sub-Gaussian with parameter L.

be the regularized design matrix underlying the covariates. Define

$$\beta_t(\delta) = \left(nL\sqrt{2\log\frac{\det(V_t)^{1/2}\det(\lambda I)^{-1/2}}{\delta}} + \lambda^{1/2}S \right)^2 .$$

Then, for any $0 < \delta < 1$ and stopping time N, with probability at least $1-\delta$,

$$\operatorname{tr}\left((\hat{\Theta}_N - \Theta_*)^\top V_N (\hat{\Theta}_N - \Theta_*) \right) \le \beta_N(\delta) .$$

Lemma 1 (Abbasi-Yadkori et al. [2011, Lemma 10]). *Let $x_1, \dots, x_t \in \mathbb{R}^d$ be such that for any $1 \le s \le t$, $\|x_s\|_2 \le L$. Let $V_t = \lambda I + \sum_{s=1}^t x_s x_s^\top$ for some $\lambda > 0$. Then,*

$$\det(V_t) \le (\lambda + tL^2/d)^d .$$

In the following lemma $z_1, z_2, \dots \in \mathbb{R}^d$ is an arbitrary sequence of d-dimensional vectors and $V_t = \lambda I + \sum_{s=1}^t z_s z_s^\top$ for some $\lambda > 0$.

Lemma 2 (Abbasi-Yadkori and Szepesvári [2011, Lemma 10]). *The following holds for any $t \ge 1$:*

$$\sum_{k=0}^{t-1} \min\left(\|z_k\|^2_{V_k^{-1}}, 1) \right) \le 2\log\frac{\det(V_t)}{\det(\lambda I)} .$$

Further, when the covariates satisfy $\|z_t\| \le c_m, t \ge 0$ with some $c_m > 0$ w.p.1 then

$$\log\frac{\det(V_t)}{\det(\lambda I)} \le (n+d)\log\frac{\lambda(n+d) + tc_m^2}{\lambda(n+d)} .$$

With the help of Theorem 1 of Abbasi-Yadkori and Szepesvári [2011] we get that for any $0 < \delta_t < 1$,

$$\operatorname{tr}((\tilde{\theta}_i(t) - \theta_i^*)V_{i,t}(\tilde{\theta}_i(t) - \theta_i^*)^\top) \le d^2 \left(\sqrt{2\log\frac{\det(V_{i,t})^{1/2}}{\delta_t \lambda_i^{d/2}}} + \sigma_i \lambda_i^{1/2} \right)^2$$

with probability at least $1 - \delta_t$. Lemma 10 of Abbasi-Yadkori et al. [2011] gives

$$\det(V_{i,t}) \le (\lambda_i + n_i(t-1))^d .$$

Using the above two inequalities together with $\operatorname{tr}(A^\top A) \ge \|A\|_2^2$ and plugging in $\lambda_i = 1$ we arrive at

$$\|(\tilde{\theta}_i(t) - \theta_i^*)V_{i,t}^{1/2}\|_2 \le d\left(\sqrt{d\log t + 2\log(1/\delta_t)} + \sigma_i \right) .$$

Now, we are ready to derive the confidence width for the estimate $\tilde{q}_i(t)$:

$$\begin{aligned} \|\tilde{q}_i(t) - q_i(t)\|_2 &= \|(\tilde{\theta}_i(t) - \theta_i^*)x_t\|_2 \\ &\le \|(\tilde{\theta}_i(t) - \theta_i^*)V_{i,t}^{1/2}\|_2 \|V_{i,t}^{-1/2}x_t\|_2 \\ &\le d\left(\sqrt{d\log t + 2\log(1/\delta_t)} + \sigma_i \right) \|x_t\|_{V_{i,t}^{-1}} \overset{\triangle}{=} w_i(t) . \qquad (4) \end{aligned}$$

With these definitions we get the following result from Theorem 1:

Theorem 4. *Let* $\mathbf{G} = (\mathbf{L}, \mathbf{H}, \mathcal{F})$ *be a partial-monitoring game with* $\mathcal{X} = \mathcal{K}_d$ *and* $\mathcal{F} = \{x \mapsto Kx : K \in \mathbb{R}^{M \times d}, K \text{ stochastic}\}$*. Then, the regret of* CBP-Side *run with the least-squares estimator and confidence widths defined above satisfies*

$$\mathbb{E}[R_T] \le C_1 N + C_2 N^{3/2} d^2 \sqrt{T} \log T$$

with some $\mathbf{G}$*-dependent constants* $C_1, C_2 > 0$*.*

Proof. Plugging in the confidence widths from Equation (4) gives

$$\begin{aligned}
&\sum_{t=1}^{T} \min\{4NW_{I_t} w_{I_t}(t), 1\} \\
&\le 4N \sum_{i=1}^{N} V_i \sum_{s=1}^{n_i(T)} \min\{w_i(t_i(s)), 1\} \qquad (5) \\
&\le 4N \max_{i \in \underline{N}} W_i \\
&\qquad \sum_{i=1}^{N} \sqrt{n_i(T) \sum_{s=1}^{n_i(T)} d \left(\sqrt{d \log t_i(s) + 2\log(1/\delta_{t_i(s)})} + \sigma_i\right) \min\left\{\|x_t\|^2_{V^{-1}_{i,t_i(s)}}, 1\right\}} \\
&\le 4Nd \max_{i \in \underline{N}} W_i \left(\sqrt{d \log T + 2\log(1/\delta_T)} + \sum_{i=1}^{N} \sigma_i\right) \sum_{i=1}^{N} \sqrt{n_i(T) 2d \log T} \qquad (6) \\
&\le 4N^{3/2} d^{3/2} \max_{i \in \underline{N}} W_i \left(\sqrt{d \log T + 2\log(1/\delta_T)} + \sum_{i=1}^{N} \sigma_i\right) \sqrt{T 2 \log T}\,,
\end{aligned}$$

where in (6) we used Lemma 10 from Abbasi-Yadkori and Szepesvári [2011]. Setting $\delta_t = 1/t^2$ gives the regret bound $\mathbb{E}[R_T] \le C_1 N + C_2 N^{3/2} d^2 \sqrt{T} \log T$. □

6.2 Multinomial Logistic Regression

In this section we will consider the case when for any given action the observations follow a multinomial logit model. A σ-dimensional multinomial logit model $q^\theta : \mathcal{X} \to \mathcal{K}_\sigma$ is defined using a feature map $\Phi : \mathcal{X} \to \mathbb{R}^{\sigma \times D}$. Here, $\theta \in \mathbb{R}^D$ is the parameter vector of the model and the dependence of q_k^θ on x is given by

$$q_k^\theta(x) = \frac{\exp(\eta_k^\theta(x))}{N^\theta(x)}, \quad \eta_k^\theta(x) = \phi_k(x)^\top \theta\,, \quad \text{where } N^\theta(x) = \sum_{k=1}^{\sigma} \exp(\eta_k^\theta(x))\,,$$

and the feature-vectors $(\phi_k^\top(x))_{k=1,\dots,K}$ are the rows of matrix $\Phi(x)$:

$$\Phi(x) = \begin{pmatrix} \phi_1^\top(x) \\ \vdots \\ \phi_K^\top(x) \end{pmatrix}.$$

The set $\mathcal{F}$ is implicitly defined as the set of maps such that the observations, for all actions, follow some multinomial logit model. More precisely, let $\mathcal{Q}_i$ be the set of admissible symbol-distribution models; in this section these will be some subset of all σ_i-dimensional multinomial logit models with some feature maps $\Phi_i : \mathcal{X} \to \mathbb{R}^{\sigma \times D_i}$. Define $\mathcal{F}_i = \{f : \mathcal{X} \to \mathcal{K}_M \, : \, S_i f \in \mathcal{Q}_i\}$, where $S_i f : X \to \mathcal{K}_{\sigma_i}$ is given by $(S_i f)(x) = S_i f(x)$, $x \in \mathcal{X}$. Then, $\mathcal{F} = \cap_{i \in \underline{N}} \mathcal{F}_i$. In what follows we shall assume that $\mathcal{F}$ is non-empty. This holds, for example, when the features underlying all actions correspond to a common underlying discretization of the side-information set.

As in the previous section, for each action i, the parameters θ_i of the i^{th} model are estimated using (constrained) maximum likelihood based on the observation available for that action. To simplify the presentation of the following developments, from here on we fix an action i and we will suppress the indexing of the features, parameters, etc. by the action i. Thus, Φ will denote the feature map for action i, θ will denote the underlying parameter to be tuned, etc. Thus, the set of admissible models is $\mathcal{Q} = \{q^\theta \, : \, \theta \in \Theta\}$, where $q^\theta = (q_k^\theta)_{1 \le k \le sn}$ and Θ is the set of admissible parameters.

The log-likelihood of the data available for the selected action is given by

$$\ell_t(\theta) = \sum_{s=1}^{n_i(t-1)} \sum_{k=1}^{\sigma} Z_{t_i(s),k} \log q_k^\theta(x_{t_i(s)}) \, , \quad \text{where } Z_{t,k} = \mathbb{I}_{\{Y_t = k\}}$$

and $n_i(\cdot)$, $t_i(\cdot)$ are as in the previous section. To simplify the presentation we will reindex the variables $(Z_{t_i(s),k}, x_{t_i(s)}, Y_{t_i(s)})$ as $(Z_\tau, x_\tau, Y_\tau; \tau = 1, 2, \ldots)$ (*e.g.*, $Z_{t_i(1)}$ is identified with Z_τ with $\tau = 1$). Note that the reindexing does not impact the dependence structure of the variables. In particular, by our assumption, for any $\tau > 0$ we have $Y_\tau \sim q_k^{\theta^*}(x_\tau)$ for some $\theta^* \in \Theta$. We will also drop the i subindex of $n_i(t)$.

Let us first derive the estimator that we wish to use. A simple calculation shows that

$$\frac{\partial}{\partial \theta} q_k^\theta(x) = \sum_{j=1}^{\sigma} \left\{ \mathbb{I}_{\{k=j\}} - p_j^\theta(x) \right\} \phi_j^\top(x).$$

Using $\sum_{k=1}^{\sigma} Z_{\tau,k} = 1$, from this we get that

$$\frac{\partial}{\partial \theta} \ell_t(\theta) = D_t - g_t(\theta), \text{ where}$$

$$D_t = \sum_{k=1}^{\sigma} \sum_{\tau=1}^{n(t-1)} Z_{\tau,k} \phi_k(x_\tau), \quad g_t(\theta) = \sum_{k=1}^{\sigma} \sum_{\tau=1}^{n(t-1)} q_k^\theta(x_\tau) \phi_k(x_\tau) \, .$$

Let $\hat{\theta}_t$ be the maximum likelihood solution: $D_t = g_t(\hat{\theta}_t)$. We will show below that $\hat{\theta}_t$, the maximizer of the likelihood $\ell_t(\theta)$ is uniquely defined. Since $\hat{\theta}_t$ might be outside of the set of admissible parameters Θ, we "project it back" to Θ. Our final estimator $\hat{\theta}_t$ is defined as the

$$\tilde{\theta}_t = \operatorname{argmin}_{\theta \in \Theta} \| g_t(\theta) - g_t(\hat{\theta}_t) \|^2_{V_t^{-1}} \, .$$

Here and in what follows, for a positive definite matrix $S \succ 0$. Further,

$$V_t = \sum_{\tau=1}^{n(t-1)} \sum_{k=1}^{\sigma} \phi_k(x_\tau)\phi_k(x_\tau)^\top .$$

The role of V_t will become clear in the analysis. Note that in a practical implementation first one should check $\hat{\theta}_t \in \Theta$ because if this holds then $\tilde{\theta}_t = \hat{\theta}_t$.

To ensure that V_t is invertible we assume that the algorithm generates $D\sigma$ "virtual data points" $(x_\tau)_{\tau=1,\dots,D\sigma}$ such that

$$V_{D\sigma,k} \overset{\triangle}{=} \sum_{\tau=1}^{D\sigma} \phi_k(x_\tau)\phi_k(x_\tau)^\top \succeq \lambda_0 I \succ 0, \quad 1 \le k \le \sigma . \tag{7}$$

Note that this must be done for each action, independently of each other. The corresponding observations $(Y_\tau)_{\tau=1,\dots,D\sigma}$ are arbitrarily assigned to one of the available features. (This initialization allows one to encode prior information about the models, too.)

In what follows we shall assume that the following holds:

Assumption A1. The following are assumed to hold:

(i) The set Θ is such that for all $1 \le k \le \sigma$ it holds that $0 < \inf_{\theta\in\Theta, x\in X} q_k^\theta(x) \le \sup_{\theta\in\Theta, x\in X} q_k^\theta(x) < 1$.

(ii) The constant $C_L > 0$ is known such that for any $x \in X$, $\theta, \theta' \in \Theta$, $1 \le k \le \sigma$, $|p_k^\theta(x) - p_k^{\theta'}(x)| \le C_L \|\Phi(x)(\theta - \theta')\|$, *i.e.*, $p_k^\cdot(x)$ is C_L-Lipschitzian.

Now, we are ready to state our first result:

Lemma 3. *Let Assumption A1 hold. Define*

$$\varepsilon_{\tau,k} = Z_{\tau,k} - q_k^{\theta_*}(x_\tau), \quad \xi_t = \sum_{\tau=1}^{n(t-1)} \sum_{k=1}^{\sigma} \varepsilon_{\tau,k}\phi_k(x_\tau) .$$

Then, if (7) *holds for some* $\lambda_0 > 0$ *then there exists some constant* $C > 0$ *such that for any* $1 \le j \le \sigma$, $x \in X$, $t \ge 1$,

$$|q_j^{\theta_*}(x) - q_j^{\tilde{\theta}_t}(x)| \le C \|\xi_t\|_{V_t^{-1}} \sqrt{\sum_{k=1}^{\sigma} \|\phi_k(x)\|_{V_t^{-1}}^2} .$$

Note that the constant can be computed as a function of the upper and lower bounds for the logit model values in Assumption A1(i) and λ_0.

Proof. We follow the constructions from Filippi et al. [2010]. The Hessian of the log-likelihood takes the form

$$H_t(\theta) \overset{\triangle}{=} \frac{\partial}{\partial\theta} g_t(\theta) = \sum_{j,k=1}^{\sigma} \sum_{\tau=1}^{n(t-1)} \left[(\mathbb{I}_{\{k=j\}} - q_j^\theta(x_\tau)) q_k^\theta(x_\tau) \right] \phi_k(x_\tau)\phi_j^\top(x_\tau) .$$

Using (A1)(i), one can prove that there exists some constant $C_H > 0$ such that for any $\theta \in \Theta$, $H_t(\theta) \succeq C_H V_t \succeq C_H V_D \succeq C_H \lambda_0 I \succ 0$ holds. Now define

$$\hat{H}_t = \int_0^1 \frac{\partial}{\partial\theta} g_t\big(u\theta_* + (1-u)\tilde{\theta}_t\big)\, du\,.$$

Since g_t is continuous, by the Fundamental Theorem of Calculus,

$$g_t(\theta_*) - g_t(\tilde{\theta}_t) = \hat{H}_t(\theta_* - \tilde{\theta}_t). \tag{8}$$

Now, since $H_t(\theta) \succeq C_H V_t \succ 0$, $\hat{H}_t$ is non-singular and in particular

$$\hat{H}_t^{-1} \preceq \frac{1}{C_H} V_t^{-1}\,. \tag{9}$$

By Assumption A1(ii) and (8),

$$\begin{aligned}|q_j^{\theta_*}(x) - q_j^{\tilde{\theta}_t}(x)|^2 &\le C_L^2 \sum_{k=1}^{\sigma} \Big|\langle \phi_k(x), \theta_* - \tilde{\theta}_t\rangle\Big|^2 \\ &\le C_L^2 \sum_{k=1}^{\sigma} \Big|\langle \phi_k(x), \hat{H}_t^{-1}(g_t(\theta_*) - g_t(\tilde{\theta}_t))\rangle\Big|^2\,.\end{aligned}$$

Applying Cauchy-Schwartz and (9) gives

$$\begin{aligned}\langle \phi_k(x), \hat{H}_t^{-1}(g_t(\theta_*) - g_t(\tilde{\theta}_t))\rangle &\le \|\phi_k(x)\|_{\hat{H}_t^{-1}} \, \|g_t(\theta_*) - g_t(\tilde{\theta}_t)\|_{\hat{H}_t^{-1}} \\ &\le \frac{1}{C_H} \|\phi_k(x)\|_{V_t^{-1}} \, \|g_t(\theta_*) - g_t(\tilde{\theta}_t)\|_{V_t^{-1}}\,.\end{aligned}$$

Let us now bound the second term on the right-hand side:

$$\begin{aligned}\|g_t(\theta_*) - g_t(\tilde{\theta}_t)\|_{V_t^{-1}} &\le \|g_t(\theta_*) - g_t(\hat{\theta}_t)\|_{V_t^{-1}} + \|g_t(\hat{\theta}_t) - g_t(\tilde{\theta}_t)\|_{V_t^{-1}} \\ &\le 2\|g_t(\theta_*) - g_t(\hat{\theta}_t)\|_{V_t^{-1}}\end{aligned}$$

Here, the second inequality follows from the optimizer property of $\tilde{\theta}_t$ and because $\theta_* \in \Theta$ by assumption. Now, it remains to put together the inequalities and to notice that $\xi_t = g_t(\hat{\theta}_t) - g_t(\theta_*)$. □

Now, we use the result of Lemma 3 to construct the confidence widths $w_i(t)$. First, we upper bound the term $\|\xi_t\|_{V_t^{-1}}$. Define $V_{t,k} = \sum_{\tau=1}^{n(t-1)} \phi_k(x_\tau)\phi_k(x_\tau)^\top$ to get

$$\begin{aligned}\|\xi_t\|_{V_t^{-1}} &\le \sum_{k=1}^{\sigma} \left\| \sum_{\tau=1}^{n(t-1)} \varepsilon_{\tau,k}\phi_k(x_\tau)\right\|_{V_t^{-1}} \\ &\le \sum_{\tau=1}^{D\sigma}\sum_{k=1}^{\sigma} \|\phi_k(x_\tau)\|_{V_{D\sigma,k}^{-1}} + \sum_{k=1}^{\sigma} \left\| \sum_{\tau=D\sigma+1}^{n(t-1)} \varepsilon_{\tau,k}\phi_k(x_\tau)\right\|_{V_{t,k}^{-1}}\,.\end{aligned}$$

Here we separated the terms that are obtained during the initialization as for those terms $\varepsilon_{\tau,k}$ are arbitrary (they do not posses the martingale property possesed by $\varepsilon_{\tau,k}$ coming after the initialization phase). Assuming $\lambda_0 = 1$ and that the 2-norm of $\phi_k(x_\tau)$ for any k and τ is upper bounded by the $R > 0$, we get

$$\|\xi_t\|_{V_t^{-1}} \le RD\sigma^2 + \sum_{k=1}^{\sigma} \left\| \sum_{\tau=D\sigma+1}^{n(t-1)} \varepsilon_{\tau,k}\phi_k(x_\tau) \right\|_{V_{t,k}^{-1}} .$$

Now, Theorem 1 of Abbasi-Yadkori et al. [2011] gives

$$\begin{aligned} \|\xi_t\|_{V_t^{-1}} &\le RD\sigma^2 + \sum_{k=1}^{\sigma} \sqrt{2\log\frac{\det(V_{t,k})^{1/2}}{\delta_{n(t-1)}}} \\ &\le RD\sigma^2 + \sigma\sqrt{2D(1+n(t-1)R^2/D) + 2\log(1/\delta_{n(t-1)})}\,, \end{aligned}$$

Thus the confidence width $w(t)$ is defined as

$$\begin{aligned} w(t) \overset{\triangle}{=} C\left(\sqrt{\left(2D(1+n(t-1)R^2/D) + 2\log(1/\delta_{n(t-1)})\right)} + RD\sigma^2\right) \cdot \\ \sqrt{\sum_{k=1}^{\sigma} \|\phi_k(x)\|^2_{V_t^{-1}}} . \end{aligned}$$

Note that this confidence bound must be computed for each action.

Now we state the regret bound result using Theorem 1.

Theorem 5. *With the estimate and confidence function described above,* CBP-SIDE *achieves expected regret*

$$\mathbb{E}\left[R_T\right] \le C_3 N + C_4 N^{3/2} D^2 \sqrt{T} \log T\,,$$

where $C_3, C_4 > 0$ are some **G***-dependent constants.*

Proof. The proof follows the same steps as that of Theorem 4 and thus it is omitted. □

7 Conclusions

In this paper we have considered partial-monitoring problems when the learner receives side information before he has to make a decision. Our solution shows that the strategy of Bartók et al. [2012] can be successfully generalized to this setting. The main idea is to use estimators that estimate the distributions of the observable symbols for each action given the side information. We have shown how the knowledge of these distributions (and confidence bounds for these distributions) can be used to make inferences about the losses of the individual actions, and thus eliminate suboptimal actions. As this approach does not attempt to directly estimate the outcome distribution, building suitable, computationally efficient estimators with good confidence bounds is expected to be less of a problem than if we attempted to estimate the distribution of the (unobserved) outcomes. However, estimating this distribution might allow the better use of information and thus may improve the dependence on the number of arms. It remains for future work to see if constructing such an estimator is feasible. In general, the dependence on the various problem dependent constants in our bounds is expected to be improvable, too. An interesting (and probably challenging) problem is to derive an estimator that matches existing lower bounds known for the bandit case such as given by Auer [2003]. Finally, we note that our results apply even when the side information is generated in a non-oblivious adversarial fashion. This is due to the strong pointwise bounds used in the construction of the confidence bounds.

References

Abbasi-Yadkori, Y., Szepesvári, Cs.: Regret bounds for the adaptive control of linear quadratic systems. Journal of Machine Learning Research - Proceedings Track (COLT 2011) 19, 1–26 (2011)

Abbasi-Yadkori, Y., Pál, D., Szepesvári, Cs.: Improved algorithms for linear stochastic bandits (extended version). In: NIPS, pp. 2312–2320 (2011), http://www.ualberta.ca/~szepesva/papers/linear-bandits-NIPS2011.pdf

Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. The Journal of Machine Learning Research 3, 422 (2003)

Bartók, G., Pál, D., Szepesvári, Cs.: Minimax regret of finite partial-monitoring games in stochastic environments. Journal of Machine Learning Research - Proceedings Track (COLT 2011) 19, 133–154 (2011)

Bartók, G., Zolghadr, N., Szepesvári, Cs.: An adaptive algorithm for finite stochastic partial monitoring. To appear in ICML (2012)

Cesa-Bianchi, N., Lugosi, G., Stoltz, G.: Regret minimization under partial monitoring. Math. Oper. Res. 31(3), 562–580 (2006)

Dudík, M., Hsu, D., Kale, S., Karampatziakis, N., Langford, J., Reyzin, L., Zhang, T.: Efficient optimal learning for contextual bandits. In: UAI, pp. 169–178 (2011)

Filippi, S., Cappé, O., Garivier, A., Szepesvári, Cs.: Parametric bandits: The generalized linear case. In: NIPS, pp. 586–594 (2010)

Foster, D.P., Rakhlin, A.: No internal regret via neighborhood watch. CoRR, abs/1108.6088 (2011)

Helmbold, D.P., Littlestone, N., Long, P.M.: Apple tasting. Information and Computation 161(2), 85–139 (2000)

Piccolboni, A., Schindelhauer, C.: Discrete Prediction Games with Arbitrary Feedback and Loss. In: Helmbold, D.P., Williamson, B. (eds.) COLT/EuroCOLT 2001. LNCS (LNAI), vol. 2111, pp. 208–223. Springer, Heidelberg (2001)

PAC Bounds for Discounted MDPs

Tor Lattimore and Marcus Hutter

Australian National University
{tor.lattimore,marcus.hutter}@anu.edu.au

Abstract. We study upper and lower bounds on the sample-complexity of learning near-optimal behaviour in finite-state discounted Markov Decision Processes (MDPs). We prove a new bound for a modified version of Upper Confidence Reinforcement Learning (UCRL) with only cubic dependence on the horizon. The bound is unimprovable in all parameters except the size of the state/action space, where it depends linearly on the number of non-zero transition probabilities. The lower bound strengthens previous work by being both more general (it applies to all policies) and tighter. The upper and lower bounds match up to logarithmic factors provided the transition matrix is not too dense.

Keywords: Reinforcement learning, sample-complexity, exploration exploitation, PAC-MDP, Markov decision processes.

1 Introduction

The goal of reinforcement learning is to construct algorithms that learn to act optimally, or nearly so, in unknown environments. In this paper we restrict our attention to finite state discounted MDPs with unknown transitions, but known rewards.[1] The performance of reinforcement learning algorithms in this setting can be measured in a number of ways, for instance by using regret or PAC bounds [Kak03]. We focus on the latter, which is a measure of the number of time-steps where an algorithm is not near-optimal with high probability. Many previous algorithms have been shown to be PAC with varying bounds [Kak03, SL05, SLW+06, SLL09, SS10, Aue11].

We construct a new algorithm, UCRLγ, based on Upper Confidence Reinforcement Learning (UCRL) [AJO10] and prove a PAC bound of

$$\tilde{O}\left(\frac{T}{\epsilon^2(1-\gamma)^3}\log\frac{1}{\delta}\right).$$

where T is the number of non-zero transitions in the unknown MDP. Previously, the best published bound [SS10] is

$$\tilde{O}\left(\frac{|S\times A|}{\epsilon^2(1-\gamma)^6}\log\frac{1}{\delta}\right)$$

[1] Learning reward distributions is substantially easier than transitions, so is omitted for clarity as in [SS10].

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 320–334, 2012.

Our bound is substantially better in terms of the horizon, $1/(1-\gamma)$, but can be worse if the state-space is very large compared to the horizon and the transition matrix is dense. A bound with quartic dependence on the horizon has been shown in [Aue11], but this work is still unpublished.

We also present a matching (up to logarithmic factors) lower bound that is both larger and more general than the previous best given by [SLL09].

2 Notation

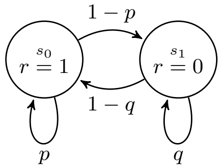
Proofs of the type found in this paper tend to use a number of complex magic constants. Readers will have an easier time if they consult the table of constants found in the appendix.

General. $\mathbb{N} = \{0, 1, 2, \cdots\}$ is the natural numbers. For the indicator function we write $[\![x = y]\!] = 1$ if $x = y$ and 0 if $x \neq y$. We use $\wedge$ and $\vee$ for logical and/or respectively. If A is a set then $|A|$ is its size and A^* is the set of all finite ordered subsets. Unless otherwise mentioned, log represents the natural logarithm. For random variable X we write $\mathbf{E}X$ and $\operatorname{Var} X$ for its expectation and variance respectively. We make frequent use of the progression defined recursively by $z_1 := 0$ and $z_{i+1} := \max\{1, 2z_i\}$. Define a set $\mathcal{Z}(a) := \{z_i : 1 \leq i \leq \arg\min_i \{z_i \geq a\}\}$. We write $\tilde{O}(\cdot)$ for big-O, but where logarithmic multiplicative factors are dropped.

Markov Decision Process. An MDP is a tuple $M = (S, A, p, r, \gamma)$ where S and A are finite sets of states and actions respectively. $r : S \to [0, 1]$ is the reward function. $p : S \times A \times S \to [0, 1]$ is the transition function and $\gamma \in (0, 1)$ the discount rate. A stationary policy π is a function $\pi : S \to A$ mapping a state to an action. We write $p_{s,a}^{s'}$ as the probability of moving from state s to s' when taking action a and $p_{s,\pi}^{s'} := p_{s,\pi(s)}^{s'}$. The value of policy π in M and state s is $V_M^\pi(s) := r(s) + \gamma \sum_{s' \in S} p_{s,\pi(s)}^{s'} V_M^\pi(s')$. We view V_M^π either as a function $V_M^\pi : S \to \mathbb{R}$ or a vector $V_M^\pi \in \mathbb{R}^{|S|}$ and similarly $p_{s,a} \in [0, 1]^{|S|}$ is a vector. $p_{s,a} \cdot V_M^\pi := \sum_{s'} p_{s,a}^{s'} V_M^\pi(s')$ is the scalar product. The optimal policy of M is defined $\pi_M^* := \arg\max_\pi V_M^\pi$. Common MDPs are M, $\widehat{M}$ and $\widetilde{M}$, which represent the true MDP, the estimated MDP using empirical transition probabilities and a model. We write $V := V_M$, $\widehat{V} := V_{\widehat{M}}$ and $\widetilde{V} := V_{\widetilde{M}}$ for their values respectively. Similarly, $\hat{\pi}^* := \pi_{\widehat{M}}^*$ and in general, variables with an MDP as a subscript will be written with a hat, tilde or nothing as appropriate and the subscript omitted.

3 Estimation

In the next section we will introduce the new algorithm, but first we give an intuitive introduction to the type of parameter estimation required to prove sample-complexity bounds for MDPs. The general idea is to use concentration inequalities to show the empiric estimate of a transition probability approaches the true probability exponentially fast in the number of samples gathered. There are many such inequalities, each catering to a slightly different purpose. We improve on previous work by using a version of Bernstein's inequality, which takes variance into account (unlike Hoeffding). The following example demonstrates the need for a variance dependent concentration inequality when estimating the value functions of MDPs. It also gives insight into the workings of the proof in the next two sections.



Consider the MDP on the right with two states and one action where rewards are shown inside the states and transition probabilities on the edges. We are only concerned with how well the value can be approximated. Assume $p > \gamma$, q arbitrarily large (but not 1) and let $\hat{p}$ be the empiric estimate of p. By writing out the definition of the value function one can show that

$$\left|V(s_0) - \widehat{V}(s_0)\right| \approx \frac{|\hat{p}-p|}{(1-\gamma)^2}. \tag{1}$$

Therefore if $V - \widehat{V}$ is to be estimated with ϵ accuracy, we need $|\hat{p}-p| < \epsilon(1-\gamma)^2$. Now suppose we bound $|\hat{p}-p|$ via a standard Hoeffding bound, then with high probability $|\hat{p}-p| \lesssim \sqrt{L/n}$ where n is the number of visits to state s_0 and $L = \log(1/\delta)$. Therefore to obtain an error less than $\epsilon(1-\gamma)^2$ we need $n > \frac{L}{\epsilon^2(1-\gamma)^4}$ visits to state s_0, which is already too many for a bound in terms of $1/(1-\gamma)^3$. If Bernstein's inequality is used instead, then $|\hat{p}-p| \lesssim \sqrt{Lp(1-p)/n}$ and so $n > \frac{Lp(1-p)}{\epsilon^2(1-\gamma)^4}$ is required, but Equation (1) depends on $p > \gamma$. Therefore $n > \frac{L}{\epsilon^2(1-\gamma)^3}$ visits are sufficient. If $p < \gamma$ then Equation (1) can be improved.

4 Upper Confidence Reinforcement Learning Algorithm

UCRL is based on the optimism principle for solving the exploration/exploitation dilemma. It is model-based in the sense that at each time-step the algorithm acts according to a model (in this case an MDP, $\widetilde{M}$) chosen from a model class. The idea is to choose the smallest model class guaranteed to contain the true model with high probability and act according to the most optimistic model within this class. With a good choice of model class this guarantees a policy that biases its exploration towards unknown states that may yield good rewards, while avoiding states that are known to be bad. The approach has been successful in obtaining uniform sample complexity (or regret) bounds in various domains where the exploration/exploitation problem is an issue [LR85, SL05, AO07, AJO10, Aue11]. We modify UCRL2 of Auer and Ortner (2010) to a new algorithm, UCRLγ, given below.

We start our analysis by considering a restricted setting where for each state/action pair in the true MDP there are at most two possible next-states, which are known. We will then apply the algorithm and bound in this setting to solve the general problem.

Assumption 1. *For each (s,a) pair the true unknown* MDP *satisfies $p_{s,a}^{s'} = 0$ for all but two $s' \in S$ denoted $sa^+, sa^- \in S$. Note that sa^+ and sa^- are dependent on (s,a) and are known to the algorithm.*

Algorithm 1. UCRLγ

```
$t = 1$, $k = 1$, $n(s,a) = n(s,a,s') = 0$ for all $s, a, s'$ and $s_1$ is the start state.
$v(s,a) = v(s,a,s') = 0$ for all $s, a, s'$
$H := \frac{1}{1-\gamma}\log\frac{8|S|}{\epsilon(1-\gamma)}$ and $w_{min} := \frac{\epsilon(1-\gamma)}{4|S|}$
$\delta_1 := \frac{\delta}{2|S\times A|}\left(\log_2|S|\log_2\frac{1}{w_{min}(1-\gamma)}\right)^{-1}$ and $L_1 := \log\frac{2}{\delta_1}$
$m := \frac{1280L_1}{\epsilon^2(1-\gamma)^2}\left(\log\log\frac{1}{1-\gamma}\right)^2\left(\log\frac{|S|}{\epsilon(1-\gamma)}\right)\log\frac{1}{\epsilon(1-\gamma)}$
loop
    $\hat{p}_{s,a}^{sa^+} := n(s,a,sa^+)/\max\{1, n(s,a)\}$
    $\mathcal{M}_k := \left\{\widetilde{M} : |\tilde{p}_{s,a}^{sa^+} - \hat{p}_{s,a}^{sa^+}| \leq$ ConfidenceInterval$(\tilde{p}_{s,a}^{sa^+}, n(s,a)),\ \forall(s,a)\right\}$
    $\widetilde{M} =$ ExtendedValueIteration$(\mathcal{M}_k)$
    $\pi_k = \tilde{\pi}^*$
    repeat
        Act
    until $v(s_{t-1}, a_{t-1}) \geq \max\{mw_{min}, n(s_{t-1}, a_{t-1})\}$ and $n(s_{t-1}, a_{t-1}) < \frac{|S|m}{1-\gamma}$
    Update$(s_{t-1}, a_{t-1})$ and Delay and $k = k + 1$
function Delay
    for $j = 1 \to H$ do
        Act
function Update$(s,a)$
    $n(s,a) = n(s,a) + v(s,a)$ and $n(s,a,s') = n(s,a,s') + v(s,a,s')\forall s'$
    $v(s,a) = v(s,a,\cdot) = 0$
function Act
    $a_t = \pi_k(s_t)$
    $s_{t+1} \sim p_{s_t,a_t}$                                        ▷ Sample from MDP
    $v(s_t,a_t) = v(s_t,a_t) + 1$ and $v(s_t,a_t,s_{t+1}) = v(s_t,a_t,s_{t+1}) + 1$ and $t = t + 1$
function ExtendedValueIteration$(\mathcal{M})$
    return optimistic $\widetilde{M} \in \mathcal{M}$ such that $V_{\widetilde{M}}^*(s) \geq V_{\widetilde{M}'}^*(s)$ for all $s \in S$ and $\widetilde{M}' \in \mathcal{M}$.
function ConfidenceInterval$(p, n)$
    return $\min\left\{\sqrt{\frac{2L_1 p(1-p)}{n}} + \frac{2L_1}{3n},\ \sqrt{\frac{L_1}{2n}}\right\}$
```

Extended Value Iteration. The function ExtendedValueIteration is as used in [SL08]. The only difference is the definition of the confidence intervals, which are now tighter for small/large values of $\hat{p}$.

Episodes and Phases. UCRLγ operates in *episodes*, which are contiguous blocks of time-steps ending when UPDATE is called. The length of each episode is not fixed, instead, an episode ends when either the number of visits to a state/action pair reaches mw_{min} for the first time or has doubled since the end of the last episode. We often refer to time-step t and episode k and unless there is ambiguity we will not define k and just assume it is the episode in which t resides. A *delay phase* is the period of $H := \frac{1}{1-\gamma}\log\frac{8|S|}{\epsilon(1-\gamma)}$ contiguous time-steps where UCRLγ is in the function DELAY, which happens immediately after an update. An *exploration phase* is a period of H time-steps starting at time t that is not in a delay phase and where $\widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t) \geq \epsilon/2$. Exploration phases do not overlap with each other, but may overlap with delay phases. More formally, the starts of exploration phases, $t_1, t_2, \cdots$, are defined inductively with $t_0 := -H$.

$$t_i := \min\left\{t : t \geq t_{i-1} + H \wedge \widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t) \geq \epsilon/2 \wedge t \text{ not in a delay phase}\right\}$$

Note there need not, and with high probability will not, be infinitely many such t_i. The exploration phases are only used in the analysis, they are not known to UCRLγ. We will later prove that the maximum number of updates is $U_{\max} := |S \times A| \log_2 \frac{|S|}{w_{min}(1-\gamma)}$ and that with high probability the number of exploration phases is bounded by $E_{\max} := 4m|S \times A| \log_2 |S| \log_2 \frac{1}{w_{min}(1-\gamma)}$. We write $n_t(s,a)$ to be the value of $n(s,a)$ at time-step t.

5 Upper PAC Bounds

We present two new PAC bounds. The first improves on all previous analyses, but relies on Assumption 1. The second is more general and optimal in all terms except the number of states, where it depends on the number of non-zero transition probabilities, T, rather than $|S \times A|$. This can be worse than the state-of-the-art if the transition matrix is dense, but by at most a factor of $|S|$.

Theorem 1. *Let M be the true* MDP *satisfying Assumption 1 and $0 < \epsilon \leq 1$ and $s_{1:t}$ the sequence of states seen up to time t. Then*

$$\mathrm{P}\left\{\sum_{t=1}^{\infty} [\![V^*(s_t) - V^{\text{UCRL}\gamma}(s_{1:t}) > \epsilon]\!] > HU_{\max} + HE_{\max}\right\} < \delta.$$

where $V^{\text{UCRL}\gamma}(s_{1:t})$ is the expected discounted value of UCRLγ *from $s_{1:t}$.*

If lower order terms are dropped then

$$HU_{\max} + HE_{\max} \in \tilde{O}\left(\frac{|S \times A|}{\epsilon^2(1-\gamma)^3}\log\frac{1}{\delta}\right).$$

Theorem 2. *Let T be the unknown number of non-zero transitions in the true* MDP *with $0 < \epsilon \leq 1$. Then there exists a modification of* UCRLγ *(see end of this section) such that*

$$\mathrm{P}\left\{\sum_{t=1}^{\infty} [\![V^*(s_t) - V^{\text{UCRL}\gamma}(s_{1:t}) > \epsilon]\!] > \frac{T}{|S \times A|} H\left(U_{\max} + E_{\max}\right)\right\} < \delta.$$

If the lower order terms are dropped then the modified PAC bound is of order

$$\tilde{O}\left(\frac{T}{\epsilon^2(1-\gamma)^3}\log\frac{1}{\delta}\right).$$

Before the proofs, we briefly compare Thereom 2 with the more recent work on the sample complexity of reinforcement learning when a generative model is available [AMK12]. In that paper they obtain a bound equal (up to logarithmic factors) to that of Theorem 2, but where the dependence on the number of states is linear. The online version of the problem studied in this paper is harder in two ways. Firstly, access to a generative model allows you to obtain independent samples from any state/action pair without needing to travel through the model. Secondly, and more subtly, the difference bounded in [AMK12] is $|V^*(s)-\hat{V}^*(s)|$ rather than the more usual $|V^*(s)-V^{\hat{\pi}^*}(s)|$, which is closer to what we require. Unfortunately, bounding the latter quantity appears to be somewhat more challenging due to subtle additional dependencies. Note that one can easily translate from the first type of bound to the second, but a naive method costs a factor of $1/(1-\gamma)$. In fact, it seems there is no clear way to modify the work in either this paper or theirs to achieve a bound on $|V^*(s)-V^{\hat{\pi}^*}(s)|$ that is both linear in the state space and cubic in the horizon, although either is possible at the expense of the other. It may eventually be a surprising fact that learning with the generative model is no easier than the online case considered in this paper.

Proof Overview. The proof of Theorem 1 borrows components from the work of [AJO10], [SL08] and [SS10]. It also shares similarities with the proofs in [AMK12], although these were independently and simultaneously discovered.

1. Bound the number of updates by $\tilde{O}\left(|S \times A| \log \frac{1}{\epsilon(1-\gamma)}\right)$, which follows from the algorithm. Since a delay phase only occurs after an update, the number of delaying phases is also bounded by this quantity.
2. Show that the true Markov Decision Process, M, remains in the model class $\mathcal{M}_k$ for all k with high probability.
3. Use the optimism principle to show that if $M \in \mathcal{M}_k$ and $V^* - V^{\text{UCRL}\gamma} > \epsilon$ then $\widetilde{V}^{\pi_k} - V^{\pi_k} > \epsilon/2$. This key fact shows that if UCRLγ is not nearly-optimal at some time-step t then the true value and model value of π_k differ and so some information is (probably) gained by following this policy.
4. The most complex part of the proof is then to show that the information gain is sufficiently quick to tightly bound the number of exploration phases by $E_{\max}$.
5. Note that $V^*(s_t) - V^{\text{UCRL}\gamma}(s_{1:t}) > \epsilon$ implies t is in a delay or exploration phase. Since with high probability there are at most $U_{\max} + E_{\max}$ of these phases, and both phases are exactly H time-steps long, the number of time-steps when UCRLγ is not ϵ-optimal is at most $HU_{\max} + HE_{\max}$.

Weights and Variances. We define the weight[2] of state/action pair (s,a) as follows.

$$\begin{aligned} w^{\pi}(s,a|s') &:= [\![(s',\pi(s')) = (s,a)]\!] + \gamma \sum_{s''} p_{s',\pi(s')}^{s''} w^{\pi}(s,a|s'') \\ w_t(s,a) &:= w^{\pi_k}(s,a|s_t). \end{aligned}$$

As usual, $\tilde{w}$ and $\hat{w}$ are defined as above but with p replaced by $\tilde{p}$ and $\hat{p}$ respectively. Think of $w_t(s,a)$ as the expected number of discounted visits to

[2] Also called the discounted future state-action distribution in [Kak03].

state/action pair (s,a) while following policy π_k starting in state s_t. The important point is that this value is approximately equal to the expected number of visits to state/action pair (s,a) within the next H time-steps. We also define the local variances of the value function. These measure the variability of values while following policy π.

$$\sigma^\pi(s,a)^2 := p_{s,a} \cdot V^{\pi^2} - [p_{s,a} \cdot V^\pi]^2 \quad \text{and} \quad \tilde{\sigma}^\pi(s,a)^2 := \tilde{p}_{s,a} \cdot \widetilde{V}^{\pi^2} - [\tilde{p}_{s,a} \cdot \widetilde{V}^\pi]^2.$$

Knownness. We define the knownness index of state s at time t as

$$\kappa_t(s,a) := \max\left\{z_i : z_i \leq \frac{n_t(s,a)}{m w_t(s,a)}\right\},$$

where m is as in the preamble of the algorithm above. The idea will be that if all states are sufficiently well known then UCRLγ will be ϵ-optimal. What we will soon show is that states with low weight need not have their transitions approximated as accurately as those with high weight. Therefore fewer visits to these states are required. Conversely, states with high weight need very accurate estimates of their transition probabilities. Fortunately, these states are precisely those we expect to visit often. By carefully balancing these factors we will show that all states become known after roughly the same number of exploration phases.

The Active Set. State/action pairs with very small $w_t(s,a)$ cannot influence the differences in value functions. Thus we define an *active* set of states where $w_t(s,a)$ is not tiny. At each time-step t define the *active* set X_t by

$$X_t \;:=\; \left\{(s,a) : w_t(s,a) > \frac{\epsilon(1-\gamma)}{4|S|} =: w_{min}\right\}.$$

We further partition the active set by knownness and weights.

$$\begin{aligned} \iota_t(s,a) \;&:=\; \min\left\{z_i : z_i \geq \frac{w_t(s,a)}{w_{min}}\right\} \\ X_{t,\kappa,\iota} \;&:=\; \{(s,a) : (s,a) \in X_t \wedge \kappa_t(s,a) = \kappa \wedge \iota_t(s,a) = \iota\} \end{aligned}$$

An easy computation shows that the indices κ and ι are contained in $\mathcal{Z}(|S|)$ and $\mathcal{Z}(\frac{1}{(1-\gamma)w_{min}})$ respectively. We write the joint index set,

$$\mathcal{K} \times \mathcal{I} := \mathcal{Z}(|S|) \times \mathcal{Z}(\frac{1}{(1-\gamma)w_{min}}).$$

Analysis. Space does not permit us to provide proofs for all results. Simple proofs are omitted while time-consuming ones are often only sketched. All details can be found in the technical report [LH12]. The proof of Theorem 1 follows easily from three key lemmas.

Lemma 3. *The following hold:*

1. *The total number of updates is bounded by* $U_{\max} := |S \times A| \log_2 \frac{|S|}{w_{min}(1-\gamma)}$.
2. *If* $M \in \mathcal{M}_k$ *and* t *is not in a delay phase and* $V^*(s_t) - V^{\text{UCRL}\gamma}(s_{1:t}) > \epsilon$ *then*
$$\widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t) > \epsilon/2.$$

Lemma 4. *$M \in \mathcal{M}_k$ for all k with probability at least $1 - \delta/2$.*

Lemma 5. *The number of exploration phases is bounded by $E_{\max}$ with probability at least $1 - \delta/2$.*

The proofs of the lemmas are delayed while we apply them to prove Theorem 1.

Proof of Theorem 1. By Lemma 4, $M \in \mathcal{M}_k$ for all k with probability $1-\delta/2$. By Lemma 5 we have that the number of exploration phases is bounded by $E_{\max}$ with probability $1 - \delta/2$. Now if t is not in a delaying or exploration phase and $M \in \mathcal{M}_k$ then by Lemma 3, UCRLγ is nearly-optimal. Finally note that the number of updates is bounded by $U_{\max}$ and so the number of time-steps in delaying phases is at most $HU_{\max}$. Therefore UCRLγ is nearly-optimal for all but $HU_{\max} + HE_{\max}$ time-steps with probability $1 - \delta$. ■

We now turn our attention to proving Lemmas 3, 4 and 5. Of these, only Lemma 5 presents a substantial challenge.

Proof of Lemma 3. For part 1 we note that no state/action pair is updated once it has been visited more than $|S|m/(1-\gamma)$ times. Since updates happen only when the visit counts would double, and only start when they are at least mw_{min}, the number of updates to pair (s, a) is bounded by $\log_2 \frac{|S|}{w_{min}(1-\gamma)}$. Therefore the total number of updates is bounded by $U_{\max} := |S \times A| \log_2 \frac{|S|}{w_{min}(1-\gamma)}$.

The proof of part 2 is closely related to the approach taken by [SL08]. Recall that $\widetilde{M}$ is chosen optimistically by extended value iteration. This generates an MDP, $\widetilde{M}$, such that $V^*_{\widetilde{M}}(s) \geq V^*_{\widetilde{M}'}(s)$ for all $\widetilde{M}' \in \mathcal{M}_k$. Since we have assumed $M \in \mathcal{M}_k$ we have that $\widetilde{V}^{\pi_k}(s) \equiv V^*_{\widetilde{M}}(s) \geq V^*_M(s)$. Therefore $\widetilde{V}^{\pi_k}(s_t) - V^{\text{UCRL}\gamma}(s_{1:t}) > \epsilon$. Finally note that t is a non-delaying time-step and so policy of UCRLγ will remain stationary and equal to π_k for at least H time-steps. Using the definition of the horizon, H, we have that $|V^{\text{UCRL}\gamma}(s_{1:t}) - V^{\pi_k}(s_t)| < \epsilon/2$. Therefore $\widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t) > \epsilon/2$ as required. ■

Proof of Lemma 4. In the previous lemma we showed that there are at most $U_{\max}$ updates where exactly one state/action pair is updated. Therefore we only need to check $M \in \mathcal{M}_k$ after each update. For each update let (s, a) be the updated state/action pair and apply the best of either Bernstein or Hoeffding inequalities[3] to show that $|\hat{p}^{sa^+}_{s,a} - p^{sa^+}_{s,a}| \leq \textsc{ConfidenceInterval}(p^{sa^+}_{s,a}, n(s, a)))$ with probability $1 - \delta_1$. Setting $\delta_1 := \frac{\delta}{2U_{\max}}$ and applying the union bound completes the proof. ■

We are now ready to work on the Lemma 5, which gives a high-probability bound on the number of exploration phases. First we will show that if t is the start of an exploration phase then there exists a (κ, ι) such that $|X_{t,\kappa,\iota}| > \kappa$. Since $X_{t,\kappa,\iota}$

[3] The application of these inequalities is somewhat delicate since although the samples from state action pair (s, a) are independent by the Markov property, they are not independent given the number of samples from (s, a). For a detailed discussion, and a proof that using these bounds is theoretically sound, see [SL08].

consists of active states with similar weights, we expect their visit counts to increase at approximately the same rate. More formally we show that:

1. If t is the start of an exploration phase then there exists (κ, ι) such that $|X_{t,\kappa,\iota}| > \kappa$.
2. If $|X_{t,\kappa,\iota}| > \kappa$ for sufficiently many t then sufficient information is gained for an update occur.
3. Combining the results above with the fact that there at most $U_{\max}$ updates completes the result.

Lemma 6. *Let t be a non-delaying time-step and assume $M \in \mathcal{M}_k$. If $|X_{t,\kappa,\iota}| \leq \kappa$ for all (κ, ι) then $|\widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t)| \leq \epsilon/2$.*

The full proof is long, technical and may be found in the associated technical report [LH12]. We provide a sketch, but first we need some useful results about MDPs and the differences in value functions. The first shows that less accurate transition probabilities are required for low-weight states than their high-weight counter parts. The second lemma formalises our intuitions in Section 3, motivates the use of Bernstein's inequalities and is the key observation to improve on the unpublished work in [Aue11], which has quartic dependence on the horizon.

Lemma 7. *Let M and $\widetilde{M}$ be two Markov decision processes differing only in transition probabilities and π be a stationary policy then*

$$V^{\pi}(s_t) - \widetilde{V}^{\pi}(s_t) \;=\; \gamma \sum_{s,a} w^{\pi}(s, a|s_t)(p_{s,a} - \tilde{p}_{s,a}) \cdot \widetilde{V}^{\pi}. \tag{2}$$

Proof sketch. Expand and rearrange the definition of the value functions. ■

Lemma 8 (Sobel 1982). *For any MDP $\widetilde{M}$, stationary policy π and state s',*

$$\sum_{s,a} \tilde{w}^{\pi}(s, a|s')\tilde{\sigma}^{\pi}(s, a)^2 \leq \frac{1}{\gamma^2(1-\gamma)^2}. \tag{3}$$

Proof sketch of Lemma 6. For ease of notation we drop references to π_k. We approximate $w_t(s,a) \approx \tilde{w}_t(s,a)$ and $|(p_{s,a} - \tilde{p}_{s,a}) \cdot \widetilde{V}| \lesssim \sqrt{\frac{L_1 \tilde{\sigma}(s,a)^2}{n_t(s,a)}}$. Using Lemma 7

$$|\widetilde{V}(s_t) - V(s_t)| \;\lesssim\; \left| \sum_{s,a \in X_t} w_t(s,a)(p_{s,a} - \tilde{p}_{s,a}) \cdot \widetilde{V} \right| \tag{4}$$

$$\lesssim \sum_{s,a \in X_t} w_t(s,a) \sqrt{\frac{L_1 \tilde{\sigma}(s,a)^2}{n_t(s,a)}} \lesssim \sum_{\kappa,\iota} \sum_{s,a \in X_{t,\kappa,\iota}} \sqrt{\frac{L_1 \tilde{w}_t(s,a)\tilde{\sigma}(s,a)^2}{\kappa m}} \tag{5}$$

$$\leq \sum_{\kappa,\iota} \sqrt{\frac{L_1 |X_{t,\kappa,\iota}|}{\kappa m} \sum_{s,a \in X_{t,\kappa,\iota}} \tilde{w}_t(s,a)\tilde{\sigma}_t(s,a)^2} \leq \sqrt{\frac{L_1 |\mathcal{K} \times \mathcal{I}|}{m\gamma^2(1-\gamma)^2}}, \tag{6}$$

where in Equation (4) we used Lemma 7 and the fact that states not in X_t are visited very infrequently. In Equation (5) we used the approximations for

$(p - \tilde{p}) \cdot \widetilde{V}$, the definition of $X_{t,\kappa,\iota}$ and the approximation $w \approx \tilde{w}$. In Equation (6) we used the Cauchy-Schwartz inequality,[4] the fact that $\kappa \geq |X_{t,\kappa,\iota}|$ and Lemma 8. Substituting

$$m := \frac{1280 L_1}{\epsilon^2(1-\gamma)^2}\left(\log\log\frac{1}{1-\gamma}\right)^2\left(\log\frac{|S|}{\epsilon(1-\gamma)}\right)\log\frac{1}{\epsilon(1-\gamma)}$$

completes the proof. The extra terms in m are needed to cover the errors in the approximations made here. ■

The full proof requires formalising the approximations made at the start of the sketch above. The second approximation is comparatively easy and follows from the definition of the confidence intervals. Showing that $w(s,a) \approx \tilde{w}(s,a)$ requires substantial work.

We have shown in Lemma 6 that if the value of UCRLγ is not ϵ-optimal then $|X_{t,\kappa,\iota}|$ must be greater than κ for some (κ, ι). Now we show that this cannot happen too often except with low probability. This will be sufficient to bound the number of exploration phases and therefore bound the number of times UCRLγ is not ϵ-optimal. Let t be the start of an exploration phase and define $\nu_t(s,a)$ to be the number of visits to state s within the next H time-steps. Formally,

$$\nu_t(s,a) := \sum_{i=t}^{t+H-1} [\![s_i = s \wedge \pi_k(s_i) = a]\!].$$

The following lemma captures our intuition that state/action pairs with high $w_t(s,a)$ will, in expectation, be visited more often.

Lemma 9. *Let t be the start of an exploration phase and $w_t(s,a) \geq w_{\min}$ then $\mathbf{E}\nu_t(s,a) \geq w_t(s,a)/2$.*

Proof of Lemma 5. Let $N := |S \times A| m$, where m is as in the proof of Lemma 6 or the appendix. We proceed in two stages. First we bound the total number of *useful* visits before $|X_{t,\kappa,\iota}| \leq \kappa$. Note that if the knownness, κ, is equal to $|S|$ then $|X_{t,\kappa,\iota}| \leq \kappa$ is vacuously true because the number of active state/action pairs is bounded by $|S|$. We then use this show that the number of exploration phases is at most $\tilde{O}(N)$ with high probability.

Bounding the Number of Useful Visits. A visit to state/action pair (s,a) in exploration phase starting at time-step t is (κ,ι)-*useful* if $(s,a) \in X_{t,\kappa,\iota}$ and $|X_{t,\kappa,\iota}| > \kappa$. Fixing a (κ,ι) we bound the number of (κ,ι)-useful visits to state/action pair (s,a). Suppose $t_1 < t_2$ with t_1 the start of an exploration phase and $(s,a) \in X_{t_1,\kappa,\iota}$. Therefore $n_{t_1}(s,a) < \kappa w_\iota m$. Now if $n_{t_2}(s,a) - n_{t_1}(s,a) \geq \kappa w_\iota m$ then an update ocurrs and for every $t_3 \geq t_2$ such that $\iota_t(s,a) = \iota$, $\kappa_t(s,a) > \kappa$. Therefore for each (κ,ι) pair there at most $|S \times A| m w_\iota \kappa \equiv N w_\iota \kappa$ visits that are (κ,ι)-useful.

Bounding the Number of Exploration Phases. Let t be the start of an exploration phase. Therefore $\widetilde{V}^{\pi_k}(s_t) - V^{\pi_k}(s_t) > \epsilon/2$ and so by Lemma 6 there exists a (κ,ι) such that $|S| \geq |X_{t,\kappa,\iota}| > \kappa$. For each (κ,ι), let $E_{\kappa,\iota}$ be the number

[4] $|\langle \mathbb{1}, v\rangle| \leq \|\mathbb{1}\|_2 \|v\|_2$.

of exploration phases where $|X_{t,\kappa,\iota}| > \kappa$. We shortly show that $\mathrm{P}\{E_{\kappa,\iota} > 4N\} < \delta_1$, which allows us to apply the union bound over all (κ, ι) pairs to show there are at most $E_{\max} := 4N|\mathcal{K} \times \mathcal{I}|$ exploration phases with probability at least $1 - \delta_1|\mathcal{K} \times \mathcal{I}| \equiv 1 - |\mathcal{K} \times \mathcal{I}|\frac{\delta}{2U_{\max}} > 1 - \delta/2$.

Bounding $\mathrm{P}\{E_{\kappa,\iota} > 4N\}$. Consider the sequence of exploration phases, $t_1, t_2, \cdots, t_{E_{\kappa,\iota}}$, such that $|X_{t_i,\kappa,\iota}| > \kappa$. We make the following observations:

1. $\{t_i\}$ is a (finite with probability 1) sequence of random variables depending on the MDP and policy.
2. The first part of this proof shows that the sequence necessarily ends after an exploration phase if the total number of (κ, ι)-useful visits is at least $Nw_\iota\kappa$. The sequence may end early for other reasons, such as states becoming unreachable or being visited while not exploring.
3. Define $\nu_i := \sum_{s,a \in X_{t_i,\kappa,\iota}} \nu_{t_i}(s, a)$, which is the number of (κ, ι)-useful visits in exploration phase t_i. Since $|X_{t_i,\kappa,\iota}| > \kappa$ and by Lemma 9, we have that $\mathbf{E}[\nu_i|\nu_1 \cdots \nu_{i-1}] \geq (\kappa + 1)w_\iota/2$ and $\mathrm{Var}[\nu_i|\nu_1 \cdots \nu_{i-1}] \leq \mathbf{E}[\nu_i|\nu_1 \cdots \nu_{i-1}]H$.[5]

We now wish to show the sequence has length at most $4N$ with probability at least $1 - \delta_1$. Define auxiliary sequences of length $4N$ by

$$\nu_i^+ := \begin{cases} \nu_i & \text{if } i \leq E_{\kappa,\iota} \\ w_\iota(\kappa+1)/2 & \text{otherwise} \end{cases} \qquad \bar{\nu}_i := \frac{\nu_i^+ w_\iota(\kappa+1)}{2\mathbf{E}[\nu_i^+|\nu_1^+ \cdots \nu_{i-1}^+]},$$

which are chosen such that $\mathbf{E}\bar{\nu}_i = \mathbf{E}[\bar{\nu}_i|\bar{\nu}_1 \cdots \bar{\nu}_{i-1}] = w_\iota(\kappa + 1)/2$. It is straightforward to verify that $\mathrm{P}\{E_{\kappa,\iota} > 4N\} \leq \mathrm{P}\left\{\sum_{i=1}^{4N} \bar{\nu}_i \leq Nw_\iota(\kappa + 1)\right\}$. We now use the method of bounded differences and the martingale version of Bernstein's inequality [CL06, §6] applied to $\sum \bar{\nu}_i$. Let $B_i := \mathbf{E}[\sum_{j=1}^{4N} \bar{\nu}_j|\bar{\nu}_1 \cdots \bar{\nu}_i]$, which forms a Doob martingale with $B_{4N} = \sum_{i=1}^{4N} \bar{\nu}_i$, $B_0 = 2Nw_\iota(\kappa+1)$ and $|B_{i+1} - B_i| \leq H$. Letting $\sigma^2 := \sum_{i=1}^{4N} \mathrm{Var}[B_i|B_1 \cdots B_{i-1}] \leq 2NHw_\iota(\kappa + 1)$, which follows by the definitions of B, $\bar{\nu}$ and by point 3 above. Then

$$\begin{aligned} \mathrm{P}\{E_{\kappa,\iota} > 4N\} &\leq \mathrm{P}\left\{\sum_{i=1}^{4N} \bar{\nu}_i \leq Nw_\iota(\kappa+1)\right\} = \mathrm{P}\{B_n - B_0 \leq -B_0/2\} \\ &\leq 2\exp\left(-\frac{\frac{1}{4}B_0^2}{2\sigma^2 + \frac{HB_0}{3}}\right) = 2\exp\left(-\frac{N^2w_\iota^2(\kappa+1)^2}{2\sigma^2 + \frac{2HNw_\iota(\kappa+1)}{3}}\right) \\ &\leq 2\exp\left(-\frac{Nw_\iota(\kappa+1)}{4H + \frac{2H}{3}}\right). \end{aligned}$$

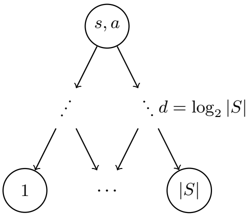
Setting this equal to δ_1, solving for N and noting that $w_\iota(\kappa + 1) \geq w_{\min}$ gives

$$N \geq \frac{5H}{w_{\min}} \log\frac{2}{\delta_1} \in \tilde{O}\left(\frac{|S|}{\epsilon(1-\gamma)^2} \log\frac{1}{\delta_1}\right)$$

Since N satisfies this, the result is complete. ∎

[5] If $X \in [0, H]$ then $\mathrm{Var}\, X < H\mathbf{E}X$. $\nu_i \in [0, H]$.

The result above completes the proof of Theorem 1. We now drop the assumption on the number of next-states by proving the more general Theorem 2. While it is possible to do this directly, we use the algorithm above.



Proof sketch of Theorem 2. The idea is to augment each state/action pair of the original MDP with $|S| - 2$ states in the form of a tree as pictured in the diagram below. The intention of the tree is to construct an MDP, $\overline{M}$, that with appropriate transition probabilities is functionally equivalent to the true MDP while satisfying Assumption 1. If we naively add the states as described above then we will add an unnecessary number of addition state/action pairs because the new states need only one action. This problem is fixed by modifying the definition of an MDP to allow a varying number of actions for each state. This adds no difficulty to the proof and means the augmented MDP now has $O(|S|^2|A|)$ state-action pairs. The rewards in the added states are set to zero.

To make the augmented MDP functionally equivalent to the true one we must also rescale γ. Let d be the depth of the tree then γ must be rescaled to $\bar{\gamma}$ such that $\bar{\gamma}^d = \gamma$. The augmented MDP is now functionally equivalent to the original in the obvious way. Policies and values can easily be translated between the two and importantly the augmented MDP now satisfies Assumption 1. Before we apply UCRLγ to $\overline{M}$ we note that the rescaling of γ has the potential to damange the bound. This is true, but fortunately the effect is not substantial since $\frac{1}{1-\bar{\gamma}} < \frac{\log |S|}{1-\gamma}$. Therefore the scaling loses at most $\log^3 |S|$ in the final PAC bound.

Now if we simply apply UCRLγ to $\overline{M}$ and use Theorem 1 to bound the number of mistakes then we obtain a PAC bound in the general case. Unfortunately, this leads to a bound depending on all the state/action pairs in $\overline{M}$, which total $|S|^2|A|$. To obtain dependence on the number of non-zero transitions, T, requires a little more justification. Let $T(s, a) := \sum_{s'} [\![p_{s,a}^{s'} > 0]\!]$ be the number of non-zero transitions from state/action pair (s, a). It is easy to show the number of reachable states in the tree associated with (s, a) is at most $T(s, a) \log |S|$. Therefore the total number of reachable state/action pairs is $|S \times A| + \log |S| \sum_{s,a} T(s, a) < 2T \log |S|$. Finally note that by Equation (2) from Lemma 7, state/action pairs that are not reachable do not contribute to the error and need no visits. This allows the analysis in Lemma 5 to be tightened, which completes the proof. ■

6 Lower PAC Bound

We now turn our attention to the lower bound. The approach is similar to that of [SLL09], but we make two refinements to improve the bound to depend on $1/(1-\gamma)^3$ and remove the policy restrictions. The first is to add a delaying state where no information can be gained, but where an algorithm may still fail to be PAC. The second is more subtle and will be described in the proof.

Theorem 10. *Let $\mathcal{A}$ be a (possibly non-stationary) policy depending on $S, A, r, \gamma, \epsilon$ and δ, then there exists a Markov decision process M_{hard} such that $V^*(s_t) - V^{\mathcal{A}}(s_{1:t}) > \epsilon$ for at least N time-steps with probability at least δ where*

$$N := \frac{c_1 |S \times A|}{\epsilon^2 (1-\gamma)^3} \log \frac{c_2}{\delta}$$

and $c_1, c_2 > 0$ are independent of the policy $\mathcal{A}$ as well as all inputs $S, A, \epsilon, \delta, \gamma$.

The proof is omitted, but we give the counter-example and intuition.

Counter Example. We prove Theorem 10 for a class of MDPs where $S = \{0, 1, \oplus, \ominus\}$ and $A = \{1, 2, \cdots, |A|\}$. The rewards and transitions for a single action are depicted in Figure 1 where $\epsilon(a^*) = 16\epsilon(1-\gamma)$ for some $a^* \in A$ and $\epsilon(a) = 0$ for all other actions. Some remarks:

1. States $\oplus$ and $\ominus$ are almost completely absorbing and confer maximum/minimum rewards respectively.
2. The transitions are independent of actions for all states except state 1. From this state, actions lead uniformly to $\oplus/\ominus$ except for one action, a^*, which has a slightly higher probability of transitioning to state $\oplus$ and so a^* is the optimal action in state 1.
3. State 0 has an absorption rate such that, on average, a policy will stay there for $1/(1-\gamma)$ time-steps.

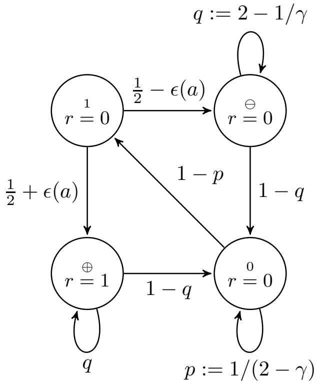


Fig. 1. Hard MDP

Intuition. The MDP in Figure 1 is very bandit-like in the sense that once a policy reaches state 1 it should choose the action most likely to lead to state $\oplus$ whereupon it will either be rewarded or punished (visit state $\oplus$ or $\ominus$). Eventually it will return to state 1 when the whole process repeats. This suggests a PAC-MDP algorithm can be used to learn the bandit with $p(a) := p^{\oplus}_{1,a}$. We then make use of a theorem of Mannor and Tsitsiklis on bandit sample-complexity [MT04] to show that with high probability the number of times a^* is not selected is at least

$$\tilde{O}\left(\frac{|A|}{\epsilon^2(1-\gamma)^2} \log \frac{1}{\delta}\right). \tag{7}$$

Improving the bound to depend on $1/(1-\gamma)^3$ is intuitively easy, but technically somewhat annoying. The idea is to consider the value differences in state 0 as well as state 1. State 0 has the following properties:

1. The absorption rate is sufficiently large that any policy remains in state 0 for around $1/(1-\gamma)$ time-steps.
2. The absorption rate is sufficiently small that the difference in values due to bad actions planned in state 1 still matter while in state 0.

While in state 0 an agent cannot make an error in the sense that $V^*(0) - Q^*(0, a) = 0$ for all a. But we are measuring $V^*(0) - V^{\mathcal{A}}(0)$ and so an agent can be penalised if its policy upon reaching state 1 is to make an error. Suppose the agent is in state 0 at some time-step before moving to state 1 and making a mistake. On average it will stay in state 0 for roughly $1/(1-\gamma)$ time-steps during which time it will *plan* a mistake upon reaching state 1. Thus the bound in Equation (7) can be multiplied by $1/(1-\gamma)$. The proof is harder because an agent need not plan to make a mistake in all future time-steps when reaching state 1 before eventually doing so in one time-step. Dependence on $|S|$ can be added easily by chaining together $|S|/4$ copies of the counter-example MDP with arbitrarily low transitions between them. Note that [SLL09] proved their theorem for a specific class of policies while Theorem 10 holds for all policies.

7 Conclusion

Summary. We presented matching upper and lower bounds on the number of time-steps when a reinforcement learning algorithm can be nearly-optimal with high probability. We now compare the bound proven in Theorem 1 with the current state-of-the-art, MORMAX [SS10].

$$\underbrace{\tilde{O}\left(\frac{T}{\epsilon^2(1-\gamma)^3}\log\frac{1}{\delta}\right)}_{\text{UCRL}\gamma} \qquad\qquad \underbrace{\tilde{O}\left(\frac{|S\times A|}{\epsilon^2(1-\gamma)^6}\log\frac{1}{\delta}\right)}_{\text{MORMAX}}$$

The dependence on ϵ and δ match the lower bound for both algorithms. UCRLγ is optimal in terms of the horizon where MORMAX loses by three factors. On the other hand, MORMAX has a bound that is linear in the state space where UCRLγ can depend quadratically. Nevertheless, UCRLγ will be prefered unless the state/action space is both dense and extremely large relative to the effective horizon. Importantly, the new upper and lower bounds now match up to logarithmic factors if the MDP has at most $|S \times A| \log |S \times A|$ non-zero transitions, so at least for this class UCRLγ is now unimprovable. Additionally, UCRLγ combined with Theorem 1 is the first demonstration of a PAC reinforcement learning algorithm with cubic dependence on the effective horizon.

Running Time. We did not analyze the running time of UCRLγ, but expect analysis similar to that of [SL08] can be used to show that UCRLγ can be approximated to run in polynomial time with no cost to sample-complexity.

References

[AJO10] Auer, P., Jaksch, T., Ortner, R.: Near-optimal regret bounds for reinforcement learning. J. Mach. Learn. Res. 99, 1563–1600 (2010)

[AMK12] Azar, M., Munos, R., Kappen, B.: On the sample complexity of reinforcement learning with a generative model. In: Proceedings of the 29th International Conference on Machine Learning. ACM, New York (2012)

[AO07] Auer, P., Ortner, R.: Logarithmic online regret bounds for undiscounted reinforcement learning. In: Advances in Neural Information Processing Systems 19, pp. 49–56. MIT Press (2007)

[Aue11] Auer, P.: Upper confidence reinforcement learning. Unpublished, keynote at European Workshop of Reinforcement Learning (2011)

[CL06] Chung, F., Lu, L.: Concentration inequalities and martingale inequalities a survey. Internet Mathematics 3, 1 (2006)

[Kak03] Kakade, S.: On The Sample Complexity of Reinforcement Learning. PhD thesis, University College London (2003)

[LH12] Lattimore, T., Hutter, M.: PAC bounds for discounted MDPs. Technical report (2012), http://arxiv.org/abs/1202.3890

[LR85] Lai, T., Robbins, H.: Asymptotically efficient adaptive allocation rules. Advances in Applied Mathematics 6(1), 4–22 (1985)

[MT04] Mannor, S., Tsitsiklis, J.: The sample complexity of exploration in the multi-armed bandit problem. J. Mach. Learn. Res. 5, 623–648 (2004)

[SL05] Strehl, A., Littman, M.: A theoretical analysis of model-based interval estimation. In: Proceedings of the 22nd International Conference on Machine Learning, ICML 2005, pp. 856–863 (2005)

[SL08] Strehl, A., Littman, M.: An analysis of model-based interval estimation for Markov decision processes. Journal of Computer and System Sciences 74(8), 1309–1331 (2008)

[SLL09] Strehl, A., Li, L., Littman, M.: Reinforcement learning in finite MDPs: PAC analysis. J. Mach. Learn. Res. 10, 2413–2444 (2009)

[SLW+06] Strehl, A., Li, L., Wiewiorac, E., Langford, J., Littman, M.: PAC model-free reinforcement learning. In: Proceedings of the 23rd International Conference on Machine Learning, ICML 2006, pp. 881–888. ACM, New York (2006)

[Sob82] Sobel, M.: The variance of discounted Markov decision processes. Journal of Applied Probability 19(4), 794–802 (1982)

[SS10] Szita, I., Szepesvári, C.: Model-based reinforcement learning with nearly tight exploration complexity bounds. In: Proceedings of the 27th International Conference on Machine Learning, pp. 1031–1038. ACM, New York (2010)

A Constants

$\lvert\mathcal{K}\times\mathcal{I}\rvert := \log_2\lvert S\rvert\log_2\frac{1}{w_{min}(1-\gamma)}$	$\tilde{O}\left(\log\lvert S\rvert\log\frac{1}{\epsilon(1-\gamma)}\right)$
$H := \frac{1}{1-\gamma}\log\frac{8\lvert S\rvert}{\epsilon(1-\gamma)}$	$\tilde{O}\left(\frac{1}{1-\gamma}\log\frac{\lvert S\rvert}{\epsilon}\right)$
$w_{\min} := \frac{\epsilon(1-\gamma)}{4\lvert S\rvert}$	$\tilde{\Omega}\left(\frac{\epsilon(1-\gamma)}{\lvert S\rvert}\right)$
$\delta_1 := \frac{\delta}{2U_{\max}}$	$\tilde{\Omega}\left(\frac{\delta}{\lvert S\times A\rvert\log\frac{1}{\epsilon(1-\gamma)}}\right)$
$L_1 := \log\frac{2}{\delta_1}$	$\tilde{O}\left(\log\frac{\lvert S\times A\rvert}{\delta\epsilon(1-\gamma)}\right)$
$m := \frac{1280L_1}{\epsilon^2(1-\gamma)^2}\left(\log\log\frac{1}{1-\gamma}\right)^2\left(\log\frac{\lvert S\rvert}{\epsilon(1-\gamma)}\right)\log\frac{1}{\epsilon(1-\gamma)}$	$\tilde{O}\left(\frac{1}{\epsilon^2(1-\gamma)^2}\log\frac{\lvert S\times A\rvert}{\delta}\right)$
$N := \lvert S\times A\rvert m$	$\tilde{O}\left(\frac{\lvert S\times A\rvert}{\epsilon^2(1-\gamma)^2}\log\frac{1}{\delta}\right)$
$E_{\max} := 4N\lvert\mathcal{K}\times\mathcal{I}\rvert$	$\tilde{O}\left(\frac{\lvert S\times A\rvert}{\epsilon^2(1-\gamma)^2}\log\frac{1}{\delta}\right)$
$U_{\max} := \lvert S\times A\rvert\log_2\frac{\lvert S\rvert}{w_{min}(1-\gamma)}$	$\tilde{O}\left(\lvert S\times A\rvert\log\frac{1}{\epsilon(1-\gamma)}\right)$

Buy Low, Sell High

Wouter M. Koolen and Vladimir Vovk

Computer Learning Research Centre, Department of Computer Science, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, United Kingdom

Abstract. We consider online trading in a single security with the objective of getting rich when its price ever exhibits a large upcrossing without risking bankruptcy. We investigate payoff guarantees that are expressed in terms of the extremity of the upcrossings. We obtain an exact and elegant characterisation of the guarantees that can be achieved. Moreover, we derive a simple canonical strategy for each attainable guarantee.

Keywords: Online investment, worst-case analysis, probability-free option pricing.

1 Introduction

We consider the simplest trading setup, where an investor trades in a single security as specified in Figure 1. An intuitive rule of thumb is to buy when the price is low, say a, and sell later when the price is high, say b. Trading successfully in such a manner exploits the so-called *upcrossing* $[a, b]$ and secures payoff b/a. In practice we do not know in advance when a stiff upcrossing will occur. Still, we can ask for a strategy whose payoff scales nicely with the extremity of the upcrossing present. A financial advisor, to express that her secret strategy approximates this ideal, may publish a function $G : \mathbb{R}^2 \to \mathbb{R}$ and promise that her strategy will

```
Initial price $\omega_0 > 0$
Starting capital $K_0 = 1$.
For $t = 1, 2, \dots$
  – Investor takes position $S_t \in \mathbb{R}$.
  – Market reveals price $\omega_t \geq 0$.
  – $K_t := K_{t-1} + S_t(\omega_t - \omega_{t-1})$
```

Fig. 1. Simple trading protocol

keep our capital above $G(a, b)$ for each upcrossing $[a, b]$

Before trusting her to manage our capital, we would like to answer the following questions:

1. Should we believe her? Is it actually *possible* to guarantee G?
2. Is she ambitious *enough*? Or can one guarantee strictly more than G?
3. Can we *reverse-engineer* a strategy to guarantee G ourselves?

The contribution of this paper is a complete resolution of these questions. We characterise the achievable guarantees, and the admissible (or Pareto optimal, i.e. not strictly dominated) guarantees. We construct, for each achievable G, a relatively simple strategy that achieves it.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 335–349, 2012.

1.1 Related Work

This work is a joint sequel to two lines of work. We think of the first line as a complete treatment of the goal of selling high (without buying low first), and of the second line as intuitive strategies for iterated trading. Let us summarise the material that we will use from each.

Sell High. Guarantees for trading once (selling at the maximum) were completely characterised in [1]. The results are as follows. We call an increasing right-continuous function $F : [1, \infty) \to [0, \infty)$ a *candidate guarantee*. A candidate guarantee F is an *adjuster* if there is a strategy that ensures $K_t \geq F(\max_s \omega_s)$ for every price evolution $\omega_0, \dots, \omega_t$. An adjuster that is not strictly dominated is called *admissible*. The goal is to find adjusters that are close to the unachievable $F_{\text{ideal}}(y) \coloneqq y$. What can be achieved is characterised as follows:

Theorem 1 (Characterisation). *A candidate guarantee F is an adjuster iff*

$$\int_1^\infty \frac{F(y)}{y^2} \,\mathrm{d}y \;\leq\; 1 \tag{1}$$

Moreover, it is admissible iff (1) holds with equality.

This elegant characterisation gives a simple test for adjusterhood. We can get reasonably close to F_{ideal}, for example using the adjusters

$$F(y) \;\coloneqq\; \alpha y^{1-\alpha} \quad \text{for some } 0 < \alpha < 1 \quad \text{or} \quad F(y) \;\coloneqq\; \frac{y^2 \ln(2)}{(1+y)\ln(1+y)^2}.$$

The following decomposition allows us to reverse engineer a canonical strategy for each adjuster F. For each price level $u \geq 1$, consider the *threshold guarantee* $F_u(y) \coloneqq u\mathbf{1}_{\{y \geq u\}}$, which is an adjuster witnessed by the strategy S_u that takes position 1 until the price first exceeds u and 0 afterwards. With this definition we have

Theorem 2 (Representation). *A candidate guarantee F is an adjuster iff there is a probability measure P on $[1, \infty)$ such that*

$$F(y) \;\leq\; \int F_u(y) \,\mathrm{d}P(u),$$

again with equality iff F is admissible.

In other words, we can witness any admissible adjuster F by the strategy $S_P \coloneqq \int S_u \,\mathrm{d}P(u)$, that is by splitting the initial capital according to the associated measure $P(u)$ over threshold strategies S_u and never rebalancing.

Iterated Trading. Intuitive trading strategies for iterated trading were proposed in [2, 3], and their worst-case performance guarantees were analysed. We briefly review the construction and guarantees specialised to the case of trading twice. The proposed strategies are of the form $S_Q := \int S_{\alpha,\beta}\,\mathrm{d}Q(\alpha,\beta)$, where Q is some bivariate probability measure and $S_{\alpha,\beta}$ is the threshold strategy that does not invest initially, subsequently invests all capital when the price first drops below α, and finally liquidates the position when the price first exceeds β. Clearly $S_{\alpha,\beta}$ witnesses the guarantee

$$G_{\alpha,\beta}(a,b) \;:=\; \frac{\beta}{\alpha}\mathbf{1}_{\{a\,\leq\,\alpha \text{ and } b\,\geq\,\beta\}},$$

and so the full strategy S_Q witnesses

$$G_Q(a,b) \;:=\; \int G_{\alpha,\beta}(a,b)\,\mathrm{d}Q(\alpha,\beta). \tag{2}$$

(We omit the iterated trading bounds and run-time analysis, they are outside the scope of this paper.)

1.2 Climax

Intuitively, the dual threshold strategies $S_{\alpha,\beta}$ are the natural generalisation of the single threshold strategies S_u. Since any univariate admissible adjuster is a convex combination of threshold guarantees, it is natural to conjecture that a bivariate candidate guarantee G is an admissible adjuster iff $G = G_Q$ for some Q.

Interestingly however, it turns out that mixture guarantees of the form (2) are typically *strictly dominated!* Let us illustrate what goes awry with a simple example. Consider the mixture-of-thresholds guarantee G defined by

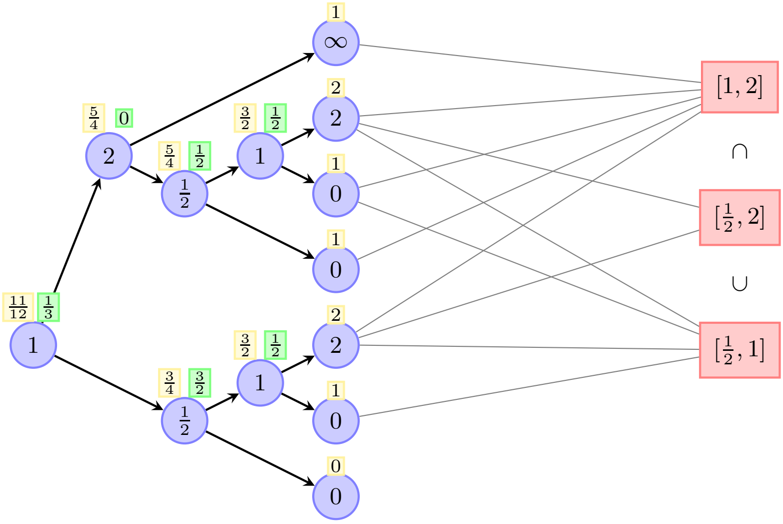
$$G(a,b) \;:=\; \frac{1}{2}G_{1,2}(a,b) + \frac{1}{2}G_{\frac{1}{2},1}(a,b) \;=\; \mathbf{1}_{\{a\,\leq\,1 \text{ and } b\,\geq\,2\}} + \mathbf{1}_{\left\{a\,\leq\,\frac{1}{2} \text{ and } b\,\geq\,1\right\}}.$$

(These weights and thresholds are chosen for simplicity and are by no means essential.) We now argue that G is strictly dominated, by showing that G can be guaranteed from initial capital $\frac{11}{12} < 1$, and hence that G is strictly dominated by the adjuster $\frac{12}{11}G$.

The smallest initial capital required to satisfy the guarantee G can be found from the tree of situations shown in Figure 2a. We restrict Market to the seven price paths that can be obtained by moving starting from the root (the left-most node labelled by 1) to the right along a branch of the tree to a leaf and reading off the price labels inside the circles. Formally, we do not allow price ∞, but it can be replaced by a sufficiently large number. The three intervals mentioned in the guarantee G and the inclusion relation between them are displayed in Figure 2c. Figure 2b indicates which price paths upcross which intervals. We can now compute the capital needed to guarantee G in each situation. First, as shown in Figure 2a, we assign to each leaf ω the capital necessary to guarantee G on it, which is by definition

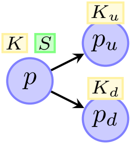
$$X_G(\omega) \;:=\; \max\bigl\{G(a,b) \bigm| a, b \text{ s.t. } \omega \text{ upcrosses } [a,b]\bigr\}$$

To label the intermediate situations we use backward induction. Let us first explain a single induction step, which is known as binomial pricing.


$\frac{11}{12}$ $\frac{1}{3}$
1
$\frac{5}{4}$ 0
2
$\frac{5}{4}$ $\frac{1}{2}$
$\frac{1}{2}$
$\frac{3}{2}$ $\frac{1}{2}$
1
$\frac{3}{4}$ $\frac{3}{2}$
$\frac{1}{2}$
$\frac{3}{2}$ $\frac{1}{2}$
1
1
∞
2
2
1
0
1
0
2
2
1
0
0
0
[1, 2]
∩
[$\frac{1}{2}$, 2]
∪
[$\frac{1}{2}$, 1]

(a) Prices, capitals, and positions (b) Upcrossing relation (c) Intervals

Fig. 2. Toy world



Binomial pricing tutorial. Consider a toy world in which the current price is p, and the future price is either $p_u > p$ in which case we want to guarantee payoff K_u, or $p_d < p$ in which case we want to guarantee K_d. The minimal initial capital K from which this is possible and the position S that achieves this are

$$K = \frac{p - p_d}{p_u - p_d} K_u + \frac{p_u - p}{p_u - p_d} K_d \qquad \text{and} \qquad S = \frac{K_u - K_d}{p_u - p_d}. \tag{3}$$

Using binomial pricing, we have labelled each internal situation in 2b by the price (left) and position (right) obtained in backward fashion. Formally, this argument only shows that an initial capital of $\frac{11}{12}$ is necessary (although intuitively it is clear that the tree exhausts all possibilities, and so $\frac{11}{12}$ is also sufficient). Indeed, it is now easy to check that an initial capital of $\frac{11}{12}$ is sufficient: the strategy that witnesses G from initial capital $\frac{11}{12}$ can be read off Figure 2a. Namely, we take position $\frac{1}{3}$ at time 0 leaving $\frac{7}{12}$ in cash. There are two cases:

- If and when the price reaches $\frac{1}{2}$ before reaching 2, we invest all our cash. This will make our position at least $\frac{7}{6} + \frac{1}{3} = \frac{3}{2}$. If and when the price reaches 1, we cash in 1 dollar leaving a position of at least $\frac{1}{2}$. If and when the price reaches 2, we cash in another dollar. In all cases, we are left with at least $X_G(\omega)$ at the end, where ω is the realized price path.
- Now suppose the price reaches 2 before reaching $\frac{1}{2}$. Cashing in our position, we get at least $\frac{7}{12} + \frac{2}{3} = \frac{5}{4}$ dollars. If and when the price reaches $\frac{1}{2}$, we take

a position of $\frac{1}{2}$, which leaves at least 1 dollar in cash. If and when the price reaches 2, we cash in another dollar. In all cases, we again are left with at least $X_G(\omega)$ at the end.

This argument shows that mixture guarantees can be strictly dominated. To get additional insight into why, let us consider the mixture strategy corresponding to G, which evenly divides its capital between $S_{1,2}$ and $S_{\frac{1}{2},1}$. The problem with this strategy is that it secures payoff 2 on price path $\omega = (1, 2, 1/2, 1, 0)$, but one only needs $X_G(\omega) = 1$ to guarantee G there. The reason is that both small intervals $[\frac{1}{2}, 1]$ and $[1, 2]$ are upcrossed, but their union $[\frac{1}{2}, 2]$ is not. In other words, the mixture strategy gives an additional payoff in certain circumstances that *does not contribute to the guarantee.* Since the binomial pricing formulas are linear in the payoffs, reducing the payoff at any leaf reduces the required initial capital.

1.3 Overview of Results

The previous section shows that the world is not simple, i.e. the intuitive characterization of guarantees is incorrect. We now present our more subtle results. We call a function $G : (0, 1] \times (0, \infty) \to [0, \infty)$ a *candidate guarantee* if it is upper semi-continuous, decreasing in its first argument and increasing in its second argument. We define the *second-argument upper inverse* of G by

$$G^{-1}(a, h) \;:=\; \inf\{b \geq a \mid G(a, b) \geq h\}. \tag{4}$$

Theorem 3 (Characterisation). *A candidate guarantee G is an adjuster iff*

$$\int_0^\infty 1 - \exp\left(\int_0^1 \frac{1}{a - G^{-1}(a, h)} \,\mathrm{d}a\right) \mathrm{d}h \;\leq\; 1. \tag{5}$$

Moreover, G is admissible iff (5) *holds with equality.*

We saw in the previous section that a subtle temporal analysis is needed when reasoning about guarantees. Although this is still true for the proof of this theorem, the result itself is elegantly timing-free.

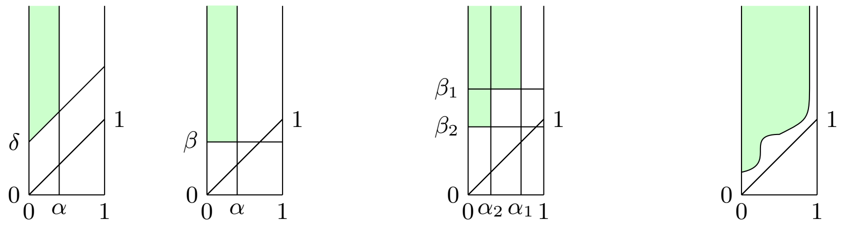
We also have a canonical representation in terms of convex combination of elementary guarantees. These elementary guarantees are analogous to the threshold strategies of the univariate case in the sense that they have just two payoff levels. However, they do have richer geometric structure. A closed set $I \subseteq (0, 1] \times (0, \infty)$ is called *north-west* if $(a, b) \in I$ implies $(0, a] \times [b, \infty) \subseteq I$. Some example north-west sets are displayed in Figure 3. We associate to each north-west set its *frontier*

$$f_I(a) \;:=\; \inf\big\{b \geq a \;\big|\; (a, b) \in I\big\}.$$

By the previous theorem, the following guarantee is an admissible adjuster:

$$G_I(a, b) \;:=\; \frac{\mathbf{1}_{\{f_I(a) \leq b\}}}{1 - \exp\left(\int_0^1 \frac{1}{a' - f_I(a')} \,\mathrm{d}a'\right)}.$$

A family $(I_h)_{h \geq 0}$ of north-west sets is called *nested* if $x \leq y$ implies $I_x \supseteq I_y$.



(a) Skewed (b) Square (c) Disjunction of squares (d) Curved set

Fig. 3. Example north-west sets

Theorem 4 (Representation). *A candidate guarantee G is an adjuster iff there are a probability measure Q on $[0,\infty)$ and a nested family $(I_h)_{h\geq 0}$ of north-west sets such that*

$$G(a,b) \;\leq\; \int G_{I_h}(a,b)\,\mathrm{d}Q(h),$$

with equality iff G is admissible.

This theorem gives us a means to construct a canonical strategy for each adjuster G. We first decompose G into a probability measure Q and a nested family of north-west sets $(I_h)_{h\geq 0}$. We then find a strategy S_{I_h} witnessing G_{I_h} for each h. Finally, we recompose these strategies to obtain the full strategy $S_G \coloneqq \int S_{I_h}\,\mathrm{d}Q(h)$.

These two theorems parallel those of [1] with a twist. Whereas [1] decomposes single-argument adjusters in terms of threshold guarantees (which have a single degree of freedom), our elementary guarantees are parametrised by the geometrically much richer north-west sets.

1.4 Outline

The paper is structured as follows. In Section 2 we reduce finding guarantees to a particular instance of probability-free option pricing. The actual option pricing is done in Section 3. Section 4 then discusses simple example guarantees, and in particular proposes an efficiently implementable strategy with an approximately ideal guarantee. The main proofs are delayed to Sections 5 and 6. We discuss the scope and applications of our results in Section 7, where we sketch the implications for online probability prediction and hypothesis testing.

2 Reduction to Option Pricing

We will make use of the definitions of probability-free *option pricing*, which we briefly review here. We assume that the initial asset price ω_0 is one, and that the investor starts with one unit of cash. Trading proceeds in rounds. In trading period t, the investor first chooses his *position* S_t, and then the new price ω_t is revealed. After T iterations, the investor has capital $K_T = 1+\sum_{t=1}^{T} S_t(\omega_t - \omega_{t-1})$. A *trading strategy* S assigns to each sequence of past prices $\omega_{<t} = (\omega_0, \ldots, \omega_{t-1})$ a *position* $S(\omega_{<t}) \in \mathbb{R}$. Let $S * \omega$ denote the payoff of strategy S on price function ω. That is

$$S * \omega \;:=\; 1 + \sum_{t=1}^{T} S(\omega_{<t})(\omega_t - \omega_{t-1}).$$

We denote by $S *_c \omega$ the payoff obtained by executing strategy S from initial capital c instead of one.

In general, an *option* X assigns to each price function ω a real value $X(\omega)$. (We have already seen one option, namely the payoff functional $\omega \mapsto S * \omega$.) The *upper price* of X, denoted $\overline{\mathbb{E}}\,[X]$, is the minimal initial capital necessary to super-replicate X, i.e.

$$\overline{\mathbb{E}}\,[X] \;:=\; \inf\bigl\{c \bigm| \exists \text{ strategy } S \;\forall \text{ price function } \omega : S *_c \omega \geq X(\omega) \bigr\}.$$

This definition allows us to price options at the start of the game. We may also wonder about the capital necessary to super-replicate X half-way through the game, say after some past $\omega' = (\omega'_0, \ldots \omega'_t)$. This so-called *conditional upper price* is given by

$$\overline{\mathbb{E}}\,[X|\omega'] \;:=\; \inf\bigl\{c \bigm| \exists \text{ strategy } S \;\forall \text{ price function } \omega : S *_c \omega \geq X(\omega'_{<t}\omega) \bigr\}.$$

where ω ranges over price functions starting from $\omega_0 = \omega'_t$ the current price. Note how the strategy only trades on the future ω, whereas the option value depends on the past ω'.

3 Characterisation of Candidate Guarantees

Suppose we conjure up some desirable candidate guarantee G, and wonder whether it is an adjuster, and if so, whether it is admissible. To decide this, we consider the option X_G that assigns to each price function ω the minimal payoff necessary to guarantee G on it:

$$X_G(\omega) \;:=\; \sup_{[a,b]\,:\,\omega \text{ upcrosses } [a,b]} G(a,b) \;=\; \max_{\substack{0 \leq i \leq j \\ 1 \geq \omega_i \leq \omega_j}} G(\omega_i, \omega_j) \tag{6}$$

We now connect adjusters and pricing

Proposition 5. *A candidate guarantee G is an adjuster iff $\overline{\mathbb{E}}[X_G] \leq 1$. Moreover, G is admissible iff $\overline{\mathbb{E}}[X_G] = 1$.*

Proof. The first equivalence holds by definition, and $\overline{\mathbb{E}}[X_G] < 1$ clearly implies inadmissibility. It follows from the pricing Theorem 6 below that a strictly dominated adjuster must have upper price < 1. □

This result reduces testing for adjusterhood to option pricing. Next we compute the upper price of X_G. Section 5 is dedicated to the proof.

Theorem 6. *The upper price of any candidate guarantee G is*

$$\overline{\mathbb{E}}[X_G] \;=\; \int_0^\infty 1 - \exp\left(\int_0^1 \frac{1}{a - G^{-1}(a,h)}\, \mathrm{d}a\right) \mathrm{d}h.$$

4 Example Adjusters

Before we go into proofs, we have a look at the consequences. We first recover the single-argument adjuster characterisation from the double-argument version. We then consider guarantees expressed in a single-parameter summary of $[a, b]$. Finally we really exploit both arguments, and design admissible adjusters that closely approach the ideal payoff b/a with computationally efficient strategies.

4.1 Selling High: Adjusters Expressed in the Maximum Price

Theorem 6 implies the results of [1] (in particular Theorem 1) as a special case.

Proof (Alternative proof of Theorem 1). Let $F : [1, \infty) \to [0, \infty)$ be an increasing right-continuous function. Construct the guarantee $G(a, b) \coloneqq F(b)\mathbf{1}_{\{b\geq 1\}}$ that ignores its first argument. By Theorem 6

$$\overline{\mathbb{E}}[X_G] = \int_0^\infty 1 - \exp\left(\int_0^1 \frac{\mathrm{d}a}{a - \inf\{b \mid F(b) \geq h\}}\right) \mathrm{d}h = \int_0^\infty \frac{\mathrm{d}h}{\inf\{b \mid F(b) \geq h\}}.$$

Using the variable substitution $h = F(y)$ (for $y \geq 1$ and $h \geq F(1)$) and integration by parts, we obtain

$$\begin{aligned} \overline{\mathbb{E}}[X_G] &= \int_0^{F(1)} \frac{1}{\inf\{b \mid F(b) \geq h\}} \mathrm{d}h + \int_{F(1)}^\infty \frac{1}{\inf\{b \mid F(b) \geq h\}} \mathrm{d}h \\ &= F(1) + \int_1^\infty \frac{1}{y} \mathrm{d}F(y) \qquad (7) \\ &= F(1) + \frac{F(y)}{y}\bigg|_1^\infty + \int_1^\infty \frac{F(y)}{y^2} \mathrm{d}y \\ &= \int_1^\infty \frac{F(y)}{y^2} \mathrm{d}y \qquad (8) \end{aligned}$$

This derivation assumes that $F(\infty)/\infty = 0$. If $F(\infty)/\infty$ exists and is strictly positive, both (7) and (8) are equal to ∞, and so $\overline{\mathbb{E}}[X_G]$ is still equal to (8). And if $F(\infty)/\infty$ does not exist, both (7) and (8) are again equal to ∞: if one or both of them were finite, $F(\infty)/\infty$ would exist as their difference. □

4.2 Adjusters Expressed in the Size of the Upcrossing

The two natural measures of the size of an upcrossing $[a, b]$ are the length $b - a$ and the ratio b/a. Let us consider guarantees expressed in each statistic.

Length. Using the tricks from the previous section we see that candidate guarantees of the form $G(a, b) = F(b - a)$ have upper price

$$\overline{\mathbb{E}}\left[X_G\right] = \int_0^\infty F(y) \frac{e^{-1/y}}{y^2} \mathrm{d}y.$$

This is analogous to (8), but with a twist. In financial terms, the distribution with density $\frac{e^{-1/y}}{y^2} \mathrm{d}y$ is the *risk-neutral measure* of the largest upcrossed length. Similarly, $y^{-2} \mathrm{d}y$ from (8) is the risk-neutral measure of the maximum price.

Ratio. We now show that guarantees of the form $G(a,b) = F(b/a)$ for some increasing and unbounded F have infinite upper price. Such guarantees are way too good to be true: they can not be made adjusters even by re-normalisation. For simplicity assume that F is invertible. Then

$$G^{-1}(a,h) = aF^{-1}(h),$$

so that $\overline{\mathbb{E}}[X_G] = \infty$, because

$$\int_0^1 \frac{1}{a - G^{-1}(a,h)} \,\mathrm{d}a = \int_0^1 \frac{1}{a(1 - F^{-1}(h))} \,\mathrm{d}a = -\infty.$$

Other impossibility results follow from the same argument. For example, the intuitively modest candidate $G(a,b) = b^p/a^q$ has infinite price for any $p, q > 0$.

4.3 Approximately Ideal Adjusters

Our goal is to secure payoff close to the ideal b/a. The previous section shows that we cannot simply dampen the ratio b/a itself, but must make essential use of both arguments. A simple admissible adjuster that approaches the ideal is

$$G(a,b) = \frac{(b-a)^p}{a^q} \frac{(\frac{p-q}{p})^p}{\Gamma(1-p)}$$

for any $0 \leq q < p < 1$. The results in Section 5.2 below imply that this guarantee is witnessed by the strategy that in situation ω with minimum price m takes position

$$S(\omega) = \frac{(p-q)}{m^{1-p+q}} \Phi\left(\frac{m^{\frac{p-q}{p}}}{\big(X_G(\omega)\Gamma(1-p)\big)^{1/p}}\right)$$

where $\Phi(x) = \frac{\int_0^x t^{-p}e^{-t}\,\mathrm{d}t}{\Gamma(1-p)}$ is the cumulative distribution function of the Gamma distribution (with shape $1-p$ and scale 1). This function can be evaluated to arbitrary precision by many computer mathematics support systems. Note that $X_G(\omega)$ and m can be maintained incrementally; when the next price r is revealed

$$\begin{aligned} X_G(\omega, r) &= \max\big\{X_G(\omega), G(m(\omega), r)\big\} \\ m(\omega, r) &= \min\big\{m(\omega), r\big\}. \end{aligned}$$

This admissible adjuster is hence extremely attractive. It approximates the ideal guarantee, and its strategy can implemented efficiently.

5 Proof of Theorem 6

In this section we prove the characterisation theorem. It will be convenient to prove the following more general statement.

Theorem 7. *Fix any candidate guarantee G and situation $\sigma = (\omega_0, \ldots, \omega_s)$. Let us abbreviate the current price to $r := \omega_s$, the lowest observed price to $m := \min_{i=0,\ldots,s} \omega_i$, and the minimal capital needed to satisfy G at time s to $C := X_G(\sigma)$ (see (6)). The conditional upper price of X_G in situation σ is*

$$\overline{\mathbb{E}}[X_G|\sigma] \;=\; C + \int_C^\infty 1 - \frac{G^{-1}(m,h) - r}{G^{-1}(m,h) - m} \exp\left(\int_0^m \frac{\mathrm{d}a}{a - G^{-1}(a,h)} \right) \mathrm{d}h. \quad (9)$$

The proof consists of two parts. For the lower bound we construct an adversarial Market based on random walks. For the upper bound we construct a strategy for Investor. It is quite surprising that these bounds meet, since these markets are generally highly incomplete. Our method is similar to that of [4], which derives option prices assuming continuous price paths. We are not aware of general probability-free option pricing results that allow discontinuous price processes.

5.1 Lower Bound from Market Strategy

We will find a lower bound on the conditional upper price $\overline{\mathbb{E}}[X_G|\sigma]$ of the option X_G using a finite up/down scheme. For a natural number n, we discretise the vertical price axis in bins of size 2^{-n}. Consider the following restricted Market starting from time $s+1$. At each discrete time step $t > s$ we have $\omega_t = \omega_{t-1} \pm 2^{-n}$, where ω_s is understood to be $R2^{-n}$, where $R := \lfloor \omega_s 2^n \rfloor$ (rather than the real ω_s). Define the stopping time τ to be least such that $\omega_\tau = 0$. On run ω, we desire to superreplicate X_G, which can be rewritten as

$$X_G(\omega) \;=\; \max_{\substack{0 \le i \le j \le \tau(\omega) \\ 1 \ge \omega_i \le \omega_j}} G(\omega_i, \omega_j)$$

We desire to lower bound the conditional upper price of X_G for the restricted Market. By binomial pricing, this price will be the expected value under a coin flip price process (formally, we explained binomial pricing only for finite games, but the extension to an infinite horizon is easy: consider a game lasting T rounds after which the price ω is frozen and then let $T \to \infty$). That is, the option's price will be at least

$$\mathbb{E}\, X_G\left(\omega_1, \ldots, \omega_s, 2^{-n}(R + \xi_1), 2^{-n}(R + \xi_1 + \xi_2), \ldots, 2^{-n}(R + \xi_1 + \cdots + \xi_\tau)\right),$$

where the regular expectation $\mathbb{E}$ refers to ξ_s being independent random variables taking values ± 1 with equal probabilities and the term ξ_τ should be ignored when $\tau = \infty$. (We say "at least" since ω_s can exceed $R2^{-n}$.) As a first step, observe that what is important are the incremental global minima of ω, and their subsequent maxima. Set $M := \lceil m2^n \rceil$. We have that incremental minima are reached at the levels $k2^{-n}$, $k = 1, \ldots, M-1$, in decreasing order.

Define $i_k = i_k(\omega)$, $k = 1, \ldots, M-1$, to be the largest i such that, after hitting level $k2^{-n}$ at time $t > s$, ω rises to level $(k+i)2^{-n}$ before hitting level $(k-1)2^{-n}$. Define $i_M = i_M(\omega)$ to be the largest i such that, after time s, ω rises to level $(M+i)2^{-n}$ before hitting level $(M-1)2^{-n}$. Now let

$$I_k \;:=\; G\big(k2^{-n}, (k+i_k)2^{-n}\big) \qquad \text{for } 1 \le k < M,$$

$$\begin{aligned} I_M &:= G\bigl(m,(M+i_M)2^{-n}\bigr) \qquad \text{and} \\ L &:= \max_{k=1,\dots,M-1} I_k \end{aligned}$$

so that

$$\begin{aligned} \tilde{\mathbb{E}}[X_G|\sigma] \ \geq \ \mathbb{E}\,(C \vee L \vee I_M) \ &= \ C + \mathbb{E}\left((L \vee I_M - C)^+\right) \\ &= \ C + \int_C^\infty \mathbb{P}(L \vee I_M \geq h)\,\mathrm{d}h \\ &= \ C + \int_C^\infty 1 - \mathbb{P}(L < h)\,\mathbb{P}(I_M < h)\,\mathrm{d}h, \end{aligned} \tag{10}$$

where $\tilde{\mathbb{E}}$ stands for upper probability under the assumed restrictions on Market. Upon hitting level $k2^{-n}$, where $k < M$, the probability that we rise to level $(k+i)2^{-n}$ (or higher) before we hit level $(k-1)2^{-n}$ equals $\frac{1}{i+1}$. We have $\mathbb{P}(i_k \leq j) = 1 - \frac{1}{2+j}$. Starting from the level $R2^{-n}$, the probability that we rise to level $(R+i)2^{-n}$ (or higher) before we hit level $(M-1)2^{-n}$ (where $M \leq R$) equals $\frac{R-M+1}{R-M+i+1}$. We have $\mathbb{P}(M + i_M \leq R + j) = \frac{j+1}{R-M+j+2}$; this formula is also true for $M = R+1$.

Since $G(a,b)$ is right-continuous in b for each a, the infimum in (4) is attained for each $h \geq 0$. We then have $G(a,b) < h$ for all $b < G^{-1}(a,h)$ and $G(a,b) \geq h$ for all $b \geq G^{-1}(a,h)$. And we have $G\left(a, G^{-1}(a,h)\right) \geq h$, with $>$ if the level h does not occur at all. Then, for $h \geq C$,

$$\begin{aligned} \mathbb{P}(I_M < h) \ &= \ \mathbb{P}\left(G\bigl(m,(M+i_M)2^{-n}\bigr) < h\right) \\ &= \ \mathbb{P}\left((M+i_M)2^{-n} < G^{-1}(m,h)\right) \\ &= \ \mathbb{P}\left(M + i_M < 2^n G^{-1}(m,h)\right) \\ &= \ \frac{1 - R + 2^n G^{-1}(m,h)}{2 - M + 2^n G^{-1}(m,h)} \end{aligned}$$

and, for $k = 1, \dots, M-1$,

$$\begin{aligned} \mathbb{P}(I_k < h) \ &= \ \mathbb{P}\left(G\bigl(k2^{-n},(k+i_k)2^{-n}\bigr) < h\right) \\ &= \ \mathbb{P}\left((k+i_k)2^{-n} < G^{-1}(k2^{-n},h)\right) \\ &= \ \mathbb{P}\left(i_k < -k + 2^n G^{-1}(k2^{-n},h)\right) \\ &= \ 1 - \frac{1}{2 - k + 2^n G^{-1}(k2^{-n},h)} \end{aligned}$$

Therefore,

$$\begin{aligned} \ln \mathbb{P}(L < h) \ &= \ \ln \prod_{k=1}^{M-1} \mathbb{P}(I_k < h) \ = \ \ln \prod_{k=1}^{M-1} \left(1 - \frac{1}{2 - k + 2^n G^{-1}(k2^{-n},h)}\right) \\ &= \ \sum_{k=1}^{M-1} \ln\left(1 - \frac{1}{2 - k + 2^n G^{-1}(k2^{-n},h)}\right) \end{aligned}$$

$$\begin{aligned}
&\le \; -\sum_{k=1}^{M-1} \frac{1}{2-k+2^n G^{-1}(k2^{-n},h)} \\
&= \; -2^{-n} \sum_{k=1}^{M-1} \frac{1}{G^{-1}(k2^{-n},h) - k2^{-n} + 2 \times 2^{-n}} \\
&\le \; -\sum_{k=1}^{M-1} \int_{k2^{-n}}^{(k+1)2^{-n}} \frac{\mathrm{d}a}{G^{-1}(a,h) - a + 3 \times 2^{-n}} \\
&\le \; -\int_{2^{-n}}^{M2^{-n}} \frac{\mathrm{d}a}{G^{-1}(a,h) - a + 3 \times 2^{-n}} \\
&\le \; -\int_{2^{-n}}^{m} \frac{\mathrm{d}a}{G^{-1}(a,h) - a + 3 \times 2^{-n}}.
\end{aligned}$$

Plugging these inequalities into (10) results in the lower bound

$$C + \int_C^\infty 1 - \frac{G^{-1}(m,h) - 2^{-n}(R-1)}{G^{-1}(m,h) - 2^{-n}(M-2)} \exp\left(\int_{2^{-n}}^{m} \frac{\mathrm{d}a}{a - G^{-1}(a,h) - 3 \times 2^{-n}}\right) \mathrm{d}h$$

for $\overline{\mathbb{E}}[X_G|\sigma]$. Letting $n \to \infty$, we obtain the inequality $\ge$ in (9). (Notice that we only need the convergence of the above outer integral to the outer integral in (9) when the limits of integration C and ∞ are replaced by $C \vee \epsilon$ and $D \in (C \vee \epsilon, \infty)$, respectively, where ϵ is a positive constant.)

5.2 Upper Bound from Investor Strategy

To prove the inequality $\le$ in (9), we consider the strategy that starts with initial capital equal to the expression in Theorem 6, and then in situation σ takes position (with m and C as defined in Theorem 7.)

$$S(\sigma) \;\coloneqq\; \int_C^\infty \frac{1}{G^{-1}(m,h) - m} \exp\left(\int_0^m \frac{\mathrm{d}a}{a - G^{-1}(a,h)}\right) \mathrm{d}h \qquad (11)$$

(this is the derivative of the right-hand side of (9) w.r.t. the current price r). We are required to show that this strategy's capital is always equal to or exceeds the right-hand side of (9). Suppose this condition is satisfied at time t. Since the right-hand side of (9) is linear in r, this condition will still be satisfied at time $t+1$ if neither C nor m change. More generally, if the price becomes p at time $t+1$, the strategy's capital at time $t+1$ is required to be at least

$$\begin{aligned}
f(p) \;\coloneqq\; & C \vee G(m,p) + \\
& \int\limits_{C \vee G(m,p)}^{\infty} 1 - \frac{G^{-1}(m \wedge p, h) - p}{G^{-1}(m \wedge p, h) - (m \wedge p)} \exp\left(\int\limits_{0}^{m \wedge p} \frac{\mathrm{d}a}{a - G^{-1}(a,h)}\right) \mathrm{d}h.
\end{aligned}$$

Since the current capital is at least $f(r)$, it suffices to prove that $f(p)$ lies below our tangent $f(r) + S \cdot (p - r)$ to $f(p)$ at the point $p = r$. Therefore, it suffices to prove that f is concave. There are three regimes:

$$\frac{\partial^2 f(p)}{\partial^2 p} = \begin{cases} -\displaystyle\int_C^\infty \frac{\exp\left(\int_0^p \frac{\mathrm{d}a}{a-G^{-1}(a,h)}\right) \frac{\partial G^{-1}(p,h)}{\partial p}}{(p - G^{-1}(p,h))^2}\,\mathrm{d}h & \text{if } p < m \\ 0 & \text{if } m < p < G^{-1}(m,C) \\ -\dfrac{\exp\left(\int_0^m \frac{\mathrm{d}a}{a-G^{-1}(a,G(m,p))}\right) \frac{\partial G(m,p)}{\partial p}}{p - m} & \text{if } G^{-1}(m,C) < p \end{cases}$$

The first case is negative as $G^{-1}(p,h)$ increases in p. The last case is negative too, as $p-m$ is positive, and $G(m,p)$ increases in p. In the borderline cases $p = m$ and $p = G^{-1}(m,C)$, the required conditions for concavity on the one-sided first derivatives of f are easy to check.

6 Proof of Theorem 4

In this section we prove the representation theorem.

6.1 From North-West-Sets to Adjusters

Say $(I_h)_{h\geq 0}$ is a nested family of north-west sets, and Q is a probability measure on $[0,\infty)$. We now argue that

$$G(a,b) \;\coloneqq\; \int_0^\infty G_{I_h}(a,b)\,\mathrm{d}Q(h)$$

is an adjuster. It is a candidate guarantee; it is upper semi-continuous since all its super-level sets are closed and it is decreasing-increasing since each super-level set is north-west. It is an adjuster, witnessed by the strategy that splits the capital according to Q over strategies S_{I_h}.

6.2 From Adjusters to North-West-Sets

Say we have an arbitrary adjuster G. We now write it as a convex combination of nested north-west adjusters. Consider the family of super-level sets

$$I_h \;\coloneqq\; \left\{(a,b) \;\middle|\; G(a,b) \geq h\right\}$$

Since G is a candidate guarantee, each I_h is closed and north-west. By Theorem 6

$$G_{I_h}(a,b) \;=\; \frac{\mathbf{1}_{\{(a,b)\in I_h\}}}{1 - \exp\left(\int_0^1 \frac{1}{a-G^{-1}(a,h)}\,\mathrm{d}a\right)}$$

is an admissible adjuster. Now construct the measure Q on $[0, \infty)$ with

$$Q(\,\mathrm{d}h) \;:=\; \left(1 - \exp\left(\int_0^1 \frac{1}{a - G^{-1}(a,h)}\,\mathrm{d}a\right)\right)\mathrm{d}h.$$

Obviously Q is non-negative. In addition, since G is an admissible adjuster, Q integrates to 1 and hence is a probability measure. Finally, for each (a, b)

$$\int_0^\infty G_{I_h}(a,b)\,\mathrm{d}Q(u) \;=\; \int_0^\infty \mathbf{1}_{\{(a,b)\in I_h\}}\,\mathrm{d}h \;=\; G(a,b).$$

7 Discussion/Conclusion

We presented strategies for online trading that guarantee a large payoff when the price ever exhibits a large upcrossing, without taking any risk. We obtained an exact and elegant characterisation of the guarantees that can be achieved. We designed a guarantee that is close to ideal, and obtained an efficient strategy.

7.1 Applications

Our results are phrased in terms of finance. However, as we show in Theorem 4, a guarantee can always be achieved by a strategy that neither *sells short*, i.e. takes a negative position $S_t < 0$, or *uses leverage*, i.e. takes a position $S_t \geq K_{t-1}/\omega_{t-1}$ that is more expensive than the capital. So the fraction of capital invested $S_t\omega_{t-1}/K_{t-1} \in [0, 1]$ is a proper probability. We can therefore think of our strategies as maintaining weights on two experts. If we substitute, in place of the price, the likelihood ratio between these two experts we obtain online methods for probability prediction with the log loss function.

One application lies is hierarchical modelling, where we want to aggregate at each level of detail the predictions of a model of that complexity, and the recursive combination of more refined models. This construction drives for example the successful data compression method Context Tree Weighting [5].

Another application is hypothesis testing, where a so-called null hypothesis is compared with an alternative hypothesis. Again, substituting the likelihood ratio for the price, securing a high payoff translates to amassing evidence against the null. The presence of a large upcrossing translates back to the existence of a sub-interval of data on which the null looks particularly fishy. Our strategies would report a fair and sharp measure of evidence in the presence of any such anomalous blocks. The advantage of this method is that the loss of evidence (the adjustment) is expressed in terms of the evidential power of the anomaly and not in its timing.

7.2 Downcrossings

A natural question is whether we can exploit the fact that a downcrossing $[a, b]$ occurs, i.e. that the price exceeds b before it drops below a. However, worst-case price paths for the univariate adjuster case always eventually collapse to 0, thus downcrossing any $[a, b]$ for $0 \leq a \leq b \leq \max_t \omega_t$. Hence, the presence of a downcrossing $[a, b]$ only conveys to us the information that the maximum is at least b, and we find ourselves back in the univariate adjuster case.

7.3 Future Work

In this paper we focus on two-argument guarantees for buying once, then selling once. We are currently working on a full analysis of multi-argument guarantees for iterated trading: both for a fixed number of times and for arbitrary references.

Acknowledgments. The first author is supported by NWO Rubicon grant 680-50-1010.

References

[1] Dawid, A.P., de Rooij, S., Grünwald, P., Koolen, W.M., Shafer, G., Shen, A., Vereshchagin, N., Vovk, V.: Probability-free pricing of adjusted American lookbacks. ArXiv e-prints (August 2011)

[2] Koolen, W.M., de Rooij, S.: Switching Investments. In: Hutter, M., Stephan, F., Vovk, V., Zeugmann, T. (eds.) Algorithmic Learning Theory. LNCS (LNAI), vol. 6331, pp. 239–254. Springer, Heidelberg (2010), http://www.cwi.nl/~wmkoolen/switching_investments.pdf

[3] Koolen, W.M., de Rooij, S.: Switching investments. Theoretical Computer Science (2012); The Special Issue on Hutter, M., Stephen, F., Vork,V., Zeugmann, T. (eds.) ALT 2010. LNCS (LNAI), vol. 6331, pp. 239–257. Springer, Heidelberg (2010)

[4] Vovk, V.: Continuous-time trading and the emergence of probability. Tech. rep., Royal Holloway, University of London (2010), http://arxiv.org/abs/0904.4364

[5] Willems, F., Shtarkov, Y., Tjalkens, T.: The context tree weighting method: basic properties. IEEE Transactions on Information Theory 41(3), 653–664 (1995)

Kernelization of Matrix Updates, When and How?

Manfred K. Warmuth[1,⋆], Wojciech Kotłowski[2,⋆⋆], and Shuisheng Zhou[3,⋆⋆⋆]

[1] Department of Computer Science, University of California, Santa Cruz, CA 95064
manfred@cse.ucsc.edu
[2] Institute of Computing Science, Poznań University of Technology, Poland
wkotlowski@cs.put.poznan.pl
[3] School of Science, Xidian University, Xian, China, 710071
sszhou@mail.xidian.edu.cn

Abstract. We define what it means for a learning algorithm to be kernelizable in the case when the instances are vectors, asymmetric matrices and symmetric matrices, respectively. We can characterize kernelizability in terms of an invariance of the algorithm to certain orthogonal transformations. If we assume that the algorithm's action relies on a linear prediction, then we can show that in each case the linear parameter vector must be a certain linear combination of the instances. We give a number of examples of how to apply our methods. In particular we show how to kernelize multiplicative updates for symmetric instance matrices.

Keywords: Kernelization, multiplicative updates, rotational invariance.

1 Introduction

The following kernelization trick was popularized by a paper on support vector machines [4] and has become one of the most successful methods in machine learning: Any algorithm that reduces to computing dot products between instance vectors $\boldsymbol{x} \in \mathbb{R}^n$ can be enhanced by a feature map that maps the instances $\boldsymbol{x}$ to $\phi(\boldsymbol{x}) \in \mathbb{R}^N$ as long as there is a kernel function available which efficiently computes the dot products $\phi(\boldsymbol{x})'\phi(\widetilde{\boldsymbol{x}})$ between expanded instances. The dimension N of the expanded instance is typically much larger than the dimension n of the original instances and even may be infinite. Complicated neural nets are often beaten by simple linear models which are enhanced with a carefully chosen problem specific feature map or kernel function. The resulting algorithms only access the expanded instances $\phi(\boldsymbol{x})$ via the kernel function $k(\boldsymbol{x}, \widetilde{\boldsymbol{x}}) = \phi(\boldsymbol{x})'\phi(\widetilde{\boldsymbol{x}})$, i.e. the components of the feature vectors are never accessed.

[⋆] The author was supported by NSF grants IIS-0325363 and IIS-0917397.

[⋆⋆] This research was done while visiting UCSC supported by the NSF grant IIS-0917397.

[⋆⋆⋆] This research was done while visiting UCSC supported by the NNSFC grant 61179040.

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 350–364, 2012.

In this paper we discuss kernel methods in the matrix domain. We begin by considering instances that are outer products $\boldsymbol{x}\boldsymbol{y}'$, where $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$ (it is easy to generalize from outer products to asymmetric instance matrices $\boldsymbol{M} \in \mathbb{R}^{n\times m}$). As long as algorithms only rely on dot products between pairs $\boldsymbol{x}, \widetilde{\boldsymbol{x}}$ of *left* instances and dot products between pairs $\boldsymbol{y}, \widetilde{\boldsymbol{y}}$ of *right* instances, then we can expand the left instances $\boldsymbol{x}$ via a feature map $\phi(\boldsymbol{x})$ and the right $\boldsymbol{y}$ instances via a second feature map $\psi(\boldsymbol{y})$. Note that matrix parameters can model all interactions between components, and therefore can take second order information into account. We also consider a case when the instances are symmetric products of the form $\boldsymbol{x}\boldsymbol{x}'$, with a single feature map $\boldsymbol{x}\boldsymbol{x}' \mapsto \phi(\boldsymbol{x})\phi(\boldsymbol{x})'$.

The goal of this paper is to give "if and only if" conditions for kernelizable algorithms. We do this for three cases: vector instances, asymmetric matrix instances and symmetric matrix instances under the assumption that the algorithm is linear and produces a unique solution. The vector case has been largely worked out in [18], but we rephrase it here mainly as a reference for comparison. The matrix cases are the main contribution of the paper. We define an algorithm to be *kernelizable* if its output depends on the data only via the kernel matrix (matrices) which contains the dot products between the instance vectors. We next give a simple equivalent characterization in each case in terms of certain geometric invariance properties of the algorithm[1]. In the vector case, multiplying the instance by an orthogonal matrix must essentially keep the algorithm unchanged. In the asymmetric matrix case, the algorithm must produce the same output if the instance matrices are left and right multiplied by two orthogonal matrices. The symmetric matrix case gives the invariance under left and right multiplication by the same orthogonal matrix.

The main point of the paper is to show that in each case, if the output of the algorithm is a linear function of the input, then the algorithm is kernelizable iff the linear parameter vector/matrix is a linear combination of the instances and remains invariant under an appropriate orthogonal transformation. In particular, in the vector case the parameter vector $\boldsymbol{w}$ must be a linear combination of the instance vectors, $\boldsymbol{w} = \sum_i c_i \boldsymbol{x}_i$. When the instances are asymmetric outer products $\boldsymbol{x}_i\boldsymbol{y}_i'$, then the parameter matrix must have the form $\boldsymbol{W} = \sum_{i,j} c_{i,j}\ \boldsymbol{x}_i\boldsymbol{y}_j'$. For the symmetric outer products $\boldsymbol{x}_i\boldsymbol{x}_i'$, the symmetric parameter matrix must have the form $\boldsymbol{W} = c\boldsymbol{I} + \sum_{i,j} c_{i,j}\ \boldsymbol{x}_i\boldsymbol{x}_j'$, where $\boldsymbol{I}$ is the identity matrix in $\mathbb{R}^n$ and $c_{i,j} = c_{j,i}$. The presence of an additional identity term $\boldsymbol{I}$ in the expansion for symmetric matrices stems from the existence of a unique element that is invariant under all orthogonal transformations. Such an element does not exist for asymmetric matrices.

We then prove versions of the Representer Theorem for both asymmetric and symmetric outer products. This helps us to develop a number of methods for building kernelizable algorithms from optimization problems. In particular, we give methods for kernelizing the matrix versions of various "multiplicative"

[1] Although invariance is with respect to orthogonal transformations, we use the term *rotational invariance* rather than *orthogonal invariance*, as the former term is commonly used in the literature.

update algorithms [14,16,9]. This family of algorithms is motivated by using the quantum relative entropy as a regularization, and methods from online learning can be used to prove regret bounds that grow logarithmically in the dimensions of the vectors. The logarithmic dependence lets us use high dimensional feature spaces. Moreover, we show that if the loss function is negative (i.e., we are maximizing gains rather than minimizing losses), then the logarithmic dependence on the dimension can be reduced to the logarithmic dependence on the rank of the kernel matrix. For outer product instances, this rank is at most the number of instances T. Multiplicative algorithms learn well when there is a low-rank matrix that can accurately explain the labels [16]. The kernel method greatly enhances the applicability of multiplicative algorithms because now we can expand the instances to outer products of high-dimensional feature vectors and still obtain efficient algorithms as long as the instance matrices have low total rank.

Relationship to Previous Work: One way to ensure kernelizability in the vector case is to apply the Representer Theorem [8,11]. It states that whenever the solution minimizes the trade-off between the square Euclidean distance and a loss function that only depends on the dot products between the weight vector and feature vector, then the solution is always a linear combination of the feature vectors. Representer type theorems have recently been generalized to the case of outer product instances [1,2]. For instance, it is shown in [1] that as long as the regularization term is increasing in the spectrum of the parameter matrix and the loss function only depends on the traces of the product of the parameter matrix and the outer product instances, then algorithms that minimize a trade-off between the regularization and the loss can be kernelized. However this is only a necessary condition.

In contrast we give necessary and sufficient conditions for kernelization. Using our results we are able to prove a simple Representer Theorem that holds under conditions incomparable with those from [1]: we only assume that the problem is rotationally invariant and the solution is unique. Our proofs are elementary and intuitive. We can also handle the case of symmetric outer product instances, which is the mainstay of multiplicative updates, but was not considered in [1,2]. In [5] it was also shown that the matrix version of the p-norm perceptron can be kernelized. Again kernelizability is easily implied by our methods.

We show in this paper for an algorithm to be kernelizable, it must not even be defined as minimizing the trade-off between a regularization and a loss. Instead we show that kernelizability is characterized by a geometric invariance property. We also went through the painstaking exercise of translating our proofs to the case when instance domains are arbitrary Hilbert spaces instead of real vector spaces. No new insights were gained from this translation and we therefore present our results in the notationally simpler case of real vector spaces.

The question of whether multiplicative update algorithms are kernelizable has been a longstanding open problem in machine learning and we resolve this problem. In previous work [9], regret bounds were proven for matrix versions of multiplicative algorithms that grow logarithmically with the feature dimension N. Our work shows that the total rank of the instance matrices (or, equivalently, the rank of the kernel matrix) is the crucial parameter instead of the feature dimension N. Now the regret bounds are logarithmic in the total rank instead of the feature dimension N which can be unbounded.

2 Kernalization via Rotational Invariance

Vector Instances: We begin with the case of vector instances $\boldsymbol{x} \in \mathbb{R}^n$. Examples $(\boldsymbol{x}, \ell)$ are labeled instances where ℓ is in some fixed label domain. A *learning algorithm* $\mathcal{A}$ is any mapping from example sequences $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t, \ell_t)\}_{t=1}^T$ followed by a next instance $\boldsymbol{x}$ to some fixed output range. Informally, the output of the algorithm is the "action" that $\mathcal{A}$ takes after receiving the $\boldsymbol{\mathcal{S}}$ and an unlabeled instance $\boldsymbol{x}$. We denote with $\widehat{\boldsymbol{X}}$ the matrix with the $T+1$ instances as columns and call $\widehat{\boldsymbol{X}}'\widehat{\boldsymbol{X}}$, the *augmented kernel matrix*, where "augmented" hints at the fact that we included the unlabeled instance $\boldsymbol{x}$ as the $(T+1)$st instance. Note that $[\widehat{\boldsymbol{X}}'\widehat{\boldsymbol{X}}]_{pq}$ is the dot product $\boldsymbol{x}_p'\boldsymbol{x}_q$ for $1 \le p, q \le T+1$.

We define algorithm $\mathcal{A}$ for vector instances to be *kernelizable* if for any two input sequences $\boldsymbol{\mathcal{S}}, \boldsymbol{x}$ and $\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}$ with the same labels and the same augmented kernel matrix, algorithm $\mathcal{A}$ maps to the same output, i.e. $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}) = \mathcal{A}(\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}})$. We next rewrite this characterization using the following elementary lemma:

Lemma 1. *Two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times t}$ are orthogonal transformations of each other (i.e. there is an orthogonal matrix $\boldsymbol{U}$, such that $\boldsymbol{B} = \boldsymbol{U}\boldsymbol{A}$) iff the kernel matrices $\boldsymbol{A}'\boldsymbol{A}$ and $\boldsymbol{B}'\boldsymbol{B}$ are the same.*

For any orthogonal matrix $\boldsymbol{U} \in \mathbb{R}^{n \times n}$, let $\boldsymbol{U}\boldsymbol{\mathcal{S}}$ denote the transformed sequence $\{(\boldsymbol{U}\boldsymbol{x}_t, \ell_t)\}_{t=1}^T$. Note that the labels remain unchanged. The above lemma implies the following:

Theorem 1. *An algorithm $\mathcal{A}$ is kernelizable iff for all sequences $\boldsymbol{\mathcal{S}}$, next instance $\boldsymbol{x}$ and orthogonal matrix $\boldsymbol{U}$, $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}) = \mathcal{A}(\boldsymbol{U}\boldsymbol{\mathcal{S}}, \boldsymbol{U}\boldsymbol{x})$.*

Proof. The sequences $\boldsymbol{\mathcal{S}}, \boldsymbol{x}$ and $\boldsymbol{U}\boldsymbol{\mathcal{S}}, \boldsymbol{U}\boldsymbol{x}$ have the same labels and augmented kernel matrix. Therefore, $\mathcal{A}$ kernelizable implies that $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}) = \mathcal{A}(\boldsymbol{U}\boldsymbol{\mathcal{S}}, \boldsymbol{U}\boldsymbol{x})$ for all suitable $\boldsymbol{\mathcal{S}}$, $\boldsymbol{x}$ and $\boldsymbol{U}$. To prove the contrapositive of the opposite implication we assume there are two sequences $\boldsymbol{\mathcal{S}}, \boldsymbol{x}$ and $\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}$ with the same augmented kernel matrix for which $\mathcal{A}$ produces a different output (witnessing that $\mathcal{A}$ is not kernelizable). Then by the above lemma there is an orthogonal matrix $\boldsymbol{U}$ for which $\widetilde{\boldsymbol{\mathcal{S}}} = \boldsymbol{U}\boldsymbol{\mathcal{S}}$, $\widetilde{\boldsymbol{x}} = \boldsymbol{U}\boldsymbol{x}$, and therefore $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}) \neq \mathcal{A}(\boldsymbol{U}\boldsymbol{\mathcal{S}}, \boldsymbol{U}\boldsymbol{x})$. □

We now make an additional assumption which assures that the algorithm predicts with a linear combination of the instances: An *algorithm $\mathcal{A}$ is linear*, if upon receiving input sequence $\boldsymbol{\mathcal{S}}$ and an unlabeled instance $\boldsymbol{x}$, $\mathcal{A}$ first computes a weight vector $\boldsymbol{w} \in \mathbb{R}^n$ from the input sequence $\boldsymbol{\mathcal{S}}$ and then outputs the dot product $\boldsymbol{w}'\boldsymbol{x}$. In short, the algorithm learns a linear function. Clearly the produced $\boldsymbol{w}$ may be nonlinear in $\boldsymbol{\mathcal{S}}$.

Theorem 2. *A linear algorithm $\mathcal{A}$ is kernelizable iff for every input sequence $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t, \ell_t)\}_{t=1}^T$ the weight vector $\boldsymbol{w}$ is a linear combination of the instances of*

$\boldsymbol{\mathcal{S}}$, and the coefficients of the linear combination depend on $\boldsymbol{\mathcal{S}}$ only via the kernel matrix $\boldsymbol{X}'\boldsymbol{X}$, where $\boldsymbol{X}$ contains the instances $\{\boldsymbol{x}_t\}_{t=1}^T$ as columns.

This can be proven by essentially repackaging a theorem given in [18]. The key contribution of this paper is that we will develop analogous theorems for the case when the instances are matrices.

Asymmetric Matrix Instances: We first consider the case of asymmetric matrices. In this case the instances are outer products $\boldsymbol{x}\boldsymbol{y}'$, where $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$. Examples have the form $(\boldsymbol{x}\boldsymbol{y}', \ell)$, where ℓ is from some labeling domain. A learning algorithm $\mathcal{A}$ is again, any mapping from example sequences $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t\boldsymbol{y}_t', \ell_t)\}_{t=1}^T$, followed by a next instance $\boldsymbol{x}\boldsymbol{y}'$ to some fixed output range.[2] Now we have two augmented kernel matrices, $\widehat{\boldsymbol{X}}'\widehat{\boldsymbol{X}}$ and $\widehat{\boldsymbol{Y}}'\widehat{\boldsymbol{Y}}$, where $\widehat{\boldsymbol{X}}$ contains the T instances $\{\boldsymbol{x}_t\}_{t=1}^T$ plus $\boldsymbol{x}$ as columns and $\widehat{\boldsymbol{Y}}$ is defined similarly.

Analogous to the vector case, an algorithm $\mathcal{A}$ for asymmetric outer product instances is *kernelizable* if for any two input sequences $\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{y}'$ and $\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{y}}'$ with the same labels and the same augmented kernel matrices, algorithm $\mathcal{A}$ maps to the same output, i.e. $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{y}') = \mathcal{A}(\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{y}}')$. In the asymmetric case, we need two orthogonal matrices. For any orthogonal matrices $\boldsymbol{U} \in \mathbb{R}^{n\times n}$, and $\boldsymbol{V} \in \mathbb{R}^{m\times m}$, we let $\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}'$ denote the transformed sequence $\{(\boldsymbol{U}\boldsymbol{x}_t\boldsymbol{y}_t'\boldsymbol{V}', \ell_t)\}_{t=1}^T$. By applying Lemma 1 twice (to the left vectors $\boldsymbol{x}_t$ and the right vectors $\boldsymbol{y}_t$), it follows that algorithm $\mathcal{A}$ is kernelizable iff for all sequences $\boldsymbol{\mathcal{S}}$, next instances $\boldsymbol{x}\boldsymbol{y}'$ and orthogonal matrices $\boldsymbol{U}, \boldsymbol{V}$,

$$\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{y}') = \mathcal{A}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}', \boldsymbol{U}\boldsymbol{x}\boldsymbol{y}'\boldsymbol{V}'). \tag{1}$$

The generalization of the linearity of algorithms to the matrix domain is straightforward: An algorithm $\mathcal{A}$ is *linear* if $\mathcal{A}$, upon input $\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{y}'$, first computes a weight matrix $\boldsymbol{W} \in \mathbb{R}^{n\times m}$ from the input sequence $\boldsymbol{\mathcal{S}}$ and then outputs the trace $\mathrm{tr}(\boldsymbol{W}'\boldsymbol{x}\boldsymbol{y}')$. As we shall prove now, the linearity of the algorithm has the consequence that the algorithm maintains a weight vector that is a linear combination of the instances.

Theorem 3. *A linear algorithm $\mathcal{A}$ is kernelizable iff for every input sequence $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t\boldsymbol{y}_t', \ell_t)\}_{t=1}^T$ the weight matrix of $\mathcal{A}$ can be written as $\boldsymbol{W} = \boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'$, where $\boldsymbol{X}$ contains the instances $\{\boldsymbol{x}_t\}_{t=1}^T$ as columns, $\boldsymbol{Y}$ contains the $\{\boldsymbol{y}_t\}_{t=1}^T$ as columns, and the coefficient matrix $\boldsymbol{C} \in \mathbb{R}^{T\times T}$ depends on $\boldsymbol{\mathcal{S}}$ only via the kernel matrices $\boldsymbol{X}'\boldsymbol{X}$ and $\boldsymbol{Y}'\boldsymbol{Y}$.*

Note that the expression $\boldsymbol{W} = \boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'$ is just a concise way of expressing the linear combination of instances $\sum_{i=1}^T \sum_{j=1}^T \boldsymbol{C}_{ij}\, \boldsymbol{x}_i\boldsymbol{y}_j'$.

Proof. Let $\boldsymbol{W}(\boldsymbol{\mathcal{S}})$ denote the weight matrix produced by algorithm $\mathcal{A}$ from the sequence $\boldsymbol{\mathcal{S}}$. Since $\mathcal{A}$ is kernelizable and outputs the trace $\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{x}\boldsymbol{y}')$ we have

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\, \boldsymbol{x}\boldsymbol{y}') = \mathrm{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}')'\, \boldsymbol{U}\boldsymbol{x}\boldsymbol{y}'\boldsymbol{V}'), \tag{2}$$

[2] For conciseness we use outer products $\boldsymbol{x}\boldsymbol{y}'$ as instances instead of the longer notation $(\boldsymbol{x}, \boldsymbol{y})$. Technically this means that the kernel matrices are only determined up to sign patterns but this is immaterial.

for all sequences $\boldsymbol{\mathcal{S}}$, orthogonal matrices $\boldsymbol{U}, \boldsymbol{V}$ of dimensions $n \times n$ and $m \times m$, and instances $\boldsymbol{x}\boldsymbol{y}'$, for $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$. In Part 1, we first show that (2) implies that for any $\boldsymbol{\mathcal{S}}$, $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'$ for some $\boldsymbol{C} \in \mathbb{R}^{T\times T}$. In Part 2, we show that (2) implies that for any $\boldsymbol{\mathcal{S}}$ and orthogonal matrices $\boldsymbol{U}, \boldsymbol{V}$ of dimensions $n \times n$ and $m \times m$, respectively, $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}') = \boldsymbol{U}\boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'\boldsymbol{V}'$. This means that $\boldsymbol{C}$ is invariant under left and right orthogonal transformations $\boldsymbol{U}$ and $\boldsymbol{V}$ of the example sequence $\boldsymbol{\mathcal{S}}$, and thus by Lemma 1 this is equivalent to stating that $\boldsymbol{C}$ depends on $\boldsymbol{\mathcal{S}}$ only via the kernel matrices $\boldsymbol{X}'\boldsymbol{X}$ and $\boldsymbol{Y}'\boldsymbol{Y}$.

The opposite direction is easy: Since $\boldsymbol{C}$ depends on $\boldsymbol{\mathcal{S}}$ only via the kernel matrices, $\boldsymbol{C}$ is invariant under orthogonal transformation $\boldsymbol{\mathcal{S}} \mapsto \boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}'$ (which leaves the kernel matrices unchanged), and thus $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}') = \boldsymbol{U}\boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'\boldsymbol{V}' = \boldsymbol{U}\boldsymbol{W}(\boldsymbol{\mathcal{S}})\boldsymbol{V}'$. This implies (2) and kernelizability:

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}')'\boldsymbol{U}\boldsymbol{x}\boldsymbol{y}'\boldsymbol{V}') = \mathrm{tr}((\boldsymbol{U}\boldsymbol{W}(\boldsymbol{\mathcal{S}})\boldsymbol{V}')'\boldsymbol{U}\boldsymbol{x}\boldsymbol{y}'\boldsymbol{V}') = \mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{x}\boldsymbol{y}').$$

Proof of Part 1: Let $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^{r_1}$ be an orthonormal basis for $\mathit{Span}\left(\{\boldsymbol{x}_t\}_{t=1}^T\right)$ and $\{\widehat{\boldsymbol{y}}_j\}_{j=1}^{r_2}$ be an orthonormal basis for $\mathit{Span}\left(\{\boldsymbol{y}_t\}_{t=1}^T\right)$, where r_1 and r_2 are the ranks of the corresponding spaces. Complete these two bases to orthonormal bases for $\mathbb{R}^m$ and $\mathbb{R}^n$, respectively, and denote these bases as $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^n$ and $\{\widehat{\boldsymbol{y}}_j\}_{j=1}^m$. Since $\{\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}_j' \mid i = 1\dots n, j = 1\dots m\}$ is an orthonormal basis for $\mathbb{R}^{n\times m}$ we can rewrite the matrix $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) \in \mathbb{R}^{n\times m}$ as

$$\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \sum_{i=1}^n \sum_{j=1}^m \hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}_j'.$$

Choose any index $r_1 < p \le n$ and any index $1 \le q \le m$, and we now show that $\hat{c}_{p,q} = 0$ (the case $r_2 < q \le m$ and $1 \le p \le n$ is proven similarly). We use the notion of transformation invariance (2). We choose $\boldsymbol{x} = \widehat{\boldsymbol{x}}_p$ and $\boldsymbol{y} = \widehat{\boldsymbol{y}}_q$. Furthermore, choose $\boldsymbol{U}$ as the *Hauseholder reflection matrix* $\boldsymbol{I} - 2\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_p'$ and $\boldsymbol{V} = \boldsymbol{I}$. Since $\widehat{\boldsymbol{x}}_p \perp \boldsymbol{x}_t$ for any $t = 1,\dots,T$ (because $p > r_1$), $\boldsymbol{U}\boldsymbol{x}_t = \boldsymbol{x}_t - 2\widehat{\boldsymbol{x}}_p(\widehat{\boldsymbol{x}}_p'\boldsymbol{x}_t) = \boldsymbol{x}_t$. Also $\boldsymbol{V}\boldsymbol{y}_t = \boldsymbol{I}\boldsymbol{y}_t = \boldsymbol{y}_t$. It thus follows that the transformed sample $\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}'$ is same as the original sample $\boldsymbol{\mathcal{S}}$, and therefore $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}') = \boldsymbol{W}(\boldsymbol{\mathcal{S}})$. Thus the l.h.s. of Equation (2) becomes:

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{x}\boldsymbol{y}') = \mathrm{tr}\Big((\sum_{i,j}\hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}_j')'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}_q'\Big) = \sum_{i,j}\hat{c}_{i,j}\widehat{\boldsymbol{x}}_i'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}_q'\widehat{\boldsymbol{y}}_j = \hat{c}_{p,q}\widehat{\boldsymbol{x}}_p'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}_q'\widehat{\boldsymbol{y}}_q = \hat{c}_{p,q}.$$

However since $\widehat{\boldsymbol{x}}_p'\widehat{\boldsymbol{x}}_p = 1$, $\boldsymbol{U}\widehat{\boldsymbol{x}}_p = \widehat{\boldsymbol{x}}_p - 2\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_p'\widehat{\boldsymbol{x}}_p = -\widehat{\boldsymbol{x}}_p$ and therefore the r.h.s. of Equation (2) has the opposite sign:

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V})'\boldsymbol{U}\boldsymbol{x}\boldsymbol{y}'\boldsymbol{V}') = -\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}_q') = -\hat{c}_{p,q}.$$

We conclude that the transformation invariance (2) implies $\hat{c}_{p,q} = 0$ if $p > r_1$ (and similarly $\hat{c}_{p,q} = 0$ if $q > r_2$). Since for any $p \le r_1$, $\widehat{\boldsymbol{x}}_p$ is a linear combination of $\boldsymbol{x}_1,\dots,\boldsymbol{x}_T$, and for any $q \le r_2$, $\widehat{\boldsymbol{y}}_q$ is a linear combination of $\boldsymbol{y}_1,\dots,\boldsymbol{y}_T$, it follows that

$$\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2} \hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j = \sum_{i=1}^{T}\sum_{j=1}^{T} \boldsymbol{C}_{i,j}\boldsymbol{x}_i\boldsymbol{y}'_j,$$

for some coefficient matrix $\boldsymbol{C} \in \mathbb{R}^{T\times T}$.

Proof of Part 2: By Part 1, $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2} \hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j$. By applying Part 1 to the sequence $\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}'$ we get $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}') = \sum_{i=1}^{r_1}\sum_{j=1}^{r_2} \hat{d}_{i,j}\boldsymbol{U}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j\boldsymbol{V}'$ for some coefficients $\hat{d}_{i,j}$, because if $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^{r_1}$ is an orthonormal basis for $\boldsymbol{X}$, $\{\boldsymbol{U}\widehat{\boldsymbol{x}}_i\}_{i=1}^{r_1}$ is an orthonormal basis for $\boldsymbol{U}\boldsymbol{X}$ (and similarly for $\boldsymbol{Y}$ and $\boldsymbol{V}\boldsymbol{Y}$). To prove the Part 2, it suffices to show that $\hat{c}_{p,q} = \hat{d}_{p,q}$ for any $1 \le p \le n$ and $1 \le q \le m$. By (2),

$$\hat{c}_{p,q} = \operatorname{tr}\Big(\Big(\sum_{i,j}\hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j\Big)'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}'_q\Big) = \operatorname{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}'_q) = \operatorname{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{V}')'\boldsymbol{U}\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}_q\boldsymbol{V}')$$

$$= \operatorname{tr}\Big(\Big(\sum_{i,j}\hat{d}_{i,j}\boldsymbol{U}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j\boldsymbol{V}'\Big)'\boldsymbol{U}\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}'_q\boldsymbol{V}'\Big) = \operatorname{tr}\Big(\Big(\sum_{i,j}\hat{d}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{y}}'_j\Big)'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{y}}'_q\Big) = \hat{d}_{p,q}.\square$$

Note that the size of the coefficient matrix $\boldsymbol{C}$ is quadratic in the number of instances T. Actually $r_1 \times r_2$ non-zero coefficients suffice, where r_1, r_2 is the rank of the kernel matrices $\boldsymbol{X}, \boldsymbol{Y}$, respectively. The reason for the quadratic size is that transformation invariance for asymmetric matrices involves two orthogonal matrices $\boldsymbol{U}$ and $\boldsymbol{V}$. If we viewed the outer products $\boldsymbol{x}_t\boldsymbol{y}'_t$ in $\mathbb{R}^{n\times m}$ as vectors in $\mathbb{R}^{nm}$ and assumed rotational invariance with respect to a single orthogonal matrix of dimension $k = nm$, then $\boldsymbol{\mathcal{S}}$ would have the form $\sum_t c_t\, \boldsymbol{x}_t\boldsymbol{y}'_t$, i.e. only one coefficient per outer product instance.

There are straightforward generalizations of the above theorem to the case when the instances are general matrices of a given rank s. Using the SVD decomposition, the instances then can be written as sums of a fixed number of outer products. That is, now the instances have the form

$$\underset{n\times s}{\boldsymbol{X}_t}\ \underset{s\times m}{\boldsymbol{Y}'_t} = \sum_{q=1}^{s}\boldsymbol{x}_t^q\boldsymbol{y}_t^{q\prime}.$$

In other words the vectors $\{\boldsymbol{x}_t^q\}_{q=1}^s$ and $\{\boldsymbol{y}_t^q\}_{q=1}^s$ are the columns of $\boldsymbol{X}_t$ and $\boldsymbol{Y}_t$, respectively. The above theorem remains essentially unchanged, but for a sequence $\{\boldsymbol{X}_t\boldsymbol{Y}'_t\}_{t=1}^T$ of T instances, the kernel matrix $\boldsymbol{X}\boldsymbol{X}'$ is formed by letting $\boldsymbol{X}$ contain the columns of all $\boldsymbol{X}_t$, which adds up to sT columns in total. Similarly, $\boldsymbol{Y}$ contains the sT columns of all $\boldsymbol{Y}_t$ and both indices in the sums in the proof of Theorem 3 range from one to sT.

Symmetric Matrix Instances: Let us now consider the case of symmetric outer product instances. A broad set of applications falls into this framework, including Principal Component Analysis, Fisher Discriminant Function, or Quantum Information Theory. In this case, the instances are $\boldsymbol{x}\boldsymbol{x}'$ for $\boldsymbol{x} \in \mathbb{R}^n$, and the learning algorithm $\mathcal{A}$ is any mapping from example sequences $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t\boldsymbol{x}'_t, \ell_t)\}_{t=1}^T$ followed by a next instance $\boldsymbol{x}\boldsymbol{x}'$ to some fixed output range. Contrary to asymmetric instances, we now have a single augmented kernel matrix $\widehat{\boldsymbol{X}}'\widehat{\boldsymbol{X}}$, which contains the T instances $\{\boldsymbol{x}_t\}_{t=1}^T$ plus $\boldsymbol{x}$ as columns.

An algorithm $\mathcal{A}$ for symmetric outer product instances is *kernelizable* if for any two input sequences $\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{x}'$ and $\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{x}}'$ with the same labels and the same augmented kernel matrix, algorithm $\mathcal{A}$ maps to the same output, i.e. $\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{x}') = \mathcal{A}(\widetilde{\boldsymbol{\mathcal{S}}}, \widetilde{\boldsymbol{x}}\widetilde{\boldsymbol{x}}')$. By applying Lemma 1 it follows that $\mathcal{A}$ is kernelizable iff for all $\boldsymbol{\mathcal{S}}$, $\boldsymbol{x}\boldsymbol{x}'$ and orthogonal matrices $\boldsymbol{U}$,

$$\mathcal{A}(\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{x}') = \mathcal{A}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}', \boldsymbol{U}\boldsymbol{x}\boldsymbol{x}'\boldsymbol{U}').$$

Note that contrary to the asymmetric case, the same matrix $\boldsymbol{U}$ is applied on both sides. An *a*lgorithm $\mathcal{A}$ is linear, if upon input $\boldsymbol{\mathcal{S}}, \boldsymbol{x}\boldsymbol{x}'$, $\mathcal{A}$ first computes a *symmetric* weight matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ from the input sequence $\boldsymbol{\mathcal{S}}$ and then outputs the trace $\mathrm{tr}(\boldsymbol{W}'\boldsymbol{x}\boldsymbol{x}')$.[3]

Theorem 4. *A linear algorithm $\mathcal{A}$ is kernelizable iff for every input sequence $\boldsymbol{\mathcal{S}} = \{(\boldsymbol{x}_t\boldsymbol{x}_t', \ell_t)\}_{t=1}^T$ the weight matrix of $\mathcal{A}$ can be written as $\boldsymbol{W} = \boldsymbol{X}\boldsymbol{C}\boldsymbol{X}' + c\boldsymbol{I}$, where $\boldsymbol{X}$ contains the instances $\{\boldsymbol{x}_t\}_{t=1}^T$ as columns, $\boldsymbol{C} \in \mathbb{R}^{T\times T}$ is a symmetric coefficient matrix, c is a real number, $\boldsymbol{I}$ is the identity matrix in $\mathbb{R}^n$, and $\boldsymbol{C}$ and c depend on $\boldsymbol{\mathcal{S}}$ only via the kernel matrix $\boldsymbol{X}'\boldsymbol{X}$.*

Proof. We only show the part of the proof which corresponds to "Part 1" of the proof of Theorem 3 (the rest of the proof follows closely the proof of Theorem 3). Since $\mathcal{A}$ is kernelizable, we have

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\, \boldsymbol{x}\boldsymbol{x}') = \mathrm{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}')'\, \boldsymbol{U}\boldsymbol{x}\boldsymbol{x}'\boldsymbol{U}'), \tag{3}$$

for all $\boldsymbol{\mathcal{S}}, \boldsymbol{U}, \boldsymbol{x}\boldsymbol{x}'$. We want to show that (3) implies that for any $\boldsymbol{\mathcal{S}}$, $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \boldsymbol{X}\boldsymbol{C}\boldsymbol{X}' + c\boldsymbol{I}$ for some symmetric $\boldsymbol{C} \in \mathbb{R}^{T\times T}$ and $c \in \mathbb{R}$. Let $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^r$ be an orthonormal basis for *Span* $\left(\{\boldsymbol{x}_t\}_{t=1}^T\right)$. Complete this basis to an orthonormal basis $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^n$ for $\mathbb{R}^n$, We decompose $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \sum_{i,j} \hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{x}}_j'$, and due to symmetry of $\boldsymbol{W}(\boldsymbol{\mathcal{S}})$, $\hat{c}_{i,j} = \hat{c}_{j,i}$ for all i, j.

We need to show that $\hat{c}_{p,q} = 0$ if $p \neq q$ and either $p > r$ or $q > r$, and that $\hat{c}_{p,p} = \hat{c}$ for some constant $\hat{c}$, for $p > r$. We show the former first. Due to the symmetry of $\boldsymbol{W}(\boldsymbol{\mathcal{S}})$, it suffices to show that that $\hat{c}_{p,q} = 0$ for any $q > r$ and any p. Choose $\boldsymbol{x} = \widehat{\boldsymbol{x}}_p + \widehat{\boldsymbol{x}}_q$ and $\boldsymbol{U}$ as the Hauseholder reflection $\boldsymbol{I} - 2\widehat{\boldsymbol{x}}_q\widehat{\boldsymbol{x}}_q'$, so that $\boldsymbol{U}\boldsymbol{x}_t = \boldsymbol{x}_t$ for $1 \le t \le T$, $\boldsymbol{U}\widehat{\boldsymbol{x}}_p = \widehat{\boldsymbol{x}}_p$, and $\boldsymbol{U}\widehat{\boldsymbol{x}}_q = -\widehat{\boldsymbol{x}}_q$. Then, the transformed sample $\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}'$ is the same as the original sample $\boldsymbol{\mathcal{S}}$, and $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}') = \boldsymbol{W}(\boldsymbol{\mathcal{S}})$. Therefore, the l.h.s. and r.h.s. of (3) become

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{x}\boldsymbol{x}') = \sum_{i,j} \hat{c}_{i,j}\ \widehat{\boldsymbol{x}}_i'(\widehat{\boldsymbol{x}}_p + \widehat{\boldsymbol{x}}_q)(\widehat{\boldsymbol{x}}_p' + \widehat{\boldsymbol{x}}_q')\widehat{\boldsymbol{x}}_j = \hat{c}_{p,p} + \hat{c}_{q,q} + \hat{c}_{p,q} + \hat{c}_{q,p}$$

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U})'\boldsymbol{U}\boldsymbol{x}\boldsymbol{x}'\boldsymbol{U}') = \sum_{i,j} \hat{c}_{i,j}\ \widehat{\boldsymbol{x}}_i'(\widehat{\boldsymbol{x}}_p - \widehat{\boldsymbol{x}}_q)(\widehat{\boldsymbol{x}}_p' - \widehat{\boldsymbol{x}}_q')\widehat{\boldsymbol{x}}_j = \hat{c}_{p,p} + \hat{c}_{q,q} - \hat{c}_{p,q} - \hat{c}_{q,p},$$

which along with $\hat{c}_{p,q} = \hat{c}_{q,p}$ implies $\hat{c}_{p,q} = 0$.

To show that $\hat{c}_{p,p} = \hat{c}$ for some constant $\hat{c}$, for all $p > r$, we choose $\boldsymbol{x} = \widehat{\boldsymbol{x}}_p$, and $\boldsymbol{U}$ to be a permutation matrix that swaps the basis vectors $\widehat{\boldsymbol{x}}_p$ and $\widehat{\boldsymbol{x}}_q$ for some $q > r$, while leaving all other basis vectors unchanged, i.e.:

[3] The assumption on the symmetry of $\boldsymbol{W}$ comes without loss of generality: Given any matrix $\boldsymbol{W}$, we can always take a symmetrized version $\boldsymbol{W}_{\mathrm{sym}} = \frac{\boldsymbol{W}+\boldsymbol{W}'}{2}$, and for any $\boldsymbol{x}\boldsymbol{x}'$, it holds $\mathrm{tr}(\boldsymbol{W}_{\mathrm{sym}}'\boldsymbol{x}\boldsymbol{x}') = \mathrm{tr}(\boldsymbol{W}'\boldsymbol{x}\boldsymbol{x}')$.

$$\boldsymbol{U} = \boldsymbol{I} - \widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_p' - \widehat{\boldsymbol{x}}_q\widehat{\boldsymbol{x}}_q' + \widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_q' + \widehat{\boldsymbol{x}}_q\widehat{\boldsymbol{x}}_p'.$$

For this choice of $\boldsymbol{U}$, $\boldsymbol{U}\boldsymbol{U}' = \boldsymbol{I}$, $\boldsymbol{U}\widehat{\boldsymbol{x}}_p = \widehat{\boldsymbol{x}}_q$, $\boldsymbol{U}\widehat{\boldsymbol{x}}_q = \widehat{\boldsymbol{x}}_p$, and $\boldsymbol{U}\boldsymbol{x}_t = \boldsymbol{x}_t$ for all $1 \leq t \leq T$ (because $p, q > r$). Thus, the transformed sample $\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}'$ is the same as the original sample $\boldsymbol{\mathcal{S}}$, so that $\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U}') = \boldsymbol{W}(\boldsymbol{\mathcal{S}})$. On the other hand, $\boldsymbol{U}\boldsymbol{x}\boldsymbol{x}'\boldsymbol{U}' = \boldsymbol{U}\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_p'\boldsymbol{U}' = \widehat{\boldsymbol{x}}_q\widehat{\boldsymbol{x}}_q'$. The l.h.s. and r.h.s. (3) become

$$\operatorname{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{x}\boldsymbol{x}') = \sum_{i,j} \hat{c}_{i,j}\ \widehat{\boldsymbol{x}}_i'\widehat{\boldsymbol{x}}_p\widehat{\boldsymbol{x}}_p'\widehat{\boldsymbol{x}}_j = \hat{c}_{p,p}$$
$$\operatorname{tr}(\boldsymbol{W}(\boldsymbol{U}\boldsymbol{\mathcal{S}}\boldsymbol{U})'\boldsymbol{U}\boldsymbol{x}\boldsymbol{x}'\boldsymbol{U}') = \sum_{i,j} \hat{c}_{i,j}\ \widehat{\boldsymbol{x}}_i'\widehat{\boldsymbol{x}}_q\widehat{\boldsymbol{x}}_q'\widehat{\boldsymbol{x}}_j = \hat{c}_{q,q},$$

which implies $\hat{c}_{p,p} = \hat{c}_{q,q}$. Since q was an arbitrary index such that $q > r$, we conclude that $\hat{c}_{p,p} = \hat{c}$ for some constant $\hat{c}$, for all $p > r$.

We conclude that the transformation invariance (3) implies that

$$\boldsymbol{W}(\boldsymbol{\mathcal{S}}) = \sum_{i=1}^{r}\sum_{j=1}^{r} \hat{c}_{i,j}\widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{x}}_j' + \hat{c}\sum_{i=r+1}^{n} \widehat{\boldsymbol{x}}_i\widehat{\boldsymbol{x}}_i' = \sum_{i,j} \boldsymbol{C}_{i,j}\boldsymbol{x}_i\boldsymbol{x}_j' + c\boldsymbol{I} = \boldsymbol{X}\boldsymbol{C}\boldsymbol{X}' + c\boldsymbol{I}$$

for some coefficient matrix $\boldsymbol{C}$ and real number c, where the second equality follows from the fact that $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^r$ is an orthonormal basis for *Span* $\left(\{\boldsymbol{x}_t\}_{t=1}^T\right)$, and $\{\widehat{\boldsymbol{x}}_i\}_{i=1}^n$ an orthonormal basis for $\mathbb{R}^n$. W.l.o.g. $\boldsymbol{C}$ is symmetric, because if $\boldsymbol{C}_{i,j} \neq \boldsymbol{C}_{j,i}$, then changing both to $\frac{\boldsymbol{C}_{i,j}+\boldsymbol{C}_{j,i}}{2}$ does not change $\boldsymbol{W}(\boldsymbol{\mathcal{S}})$. □

Comparing Theorem 4 with Theorem 3, an additional term $c\boldsymbol{I}$ entered the expansion. The term was absent for asymmetric matrices as there is no identity matrix in this case. The term $c\boldsymbol{I}$ can easily be dealt with when the instances are expanded via a feature map $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$, as it leads to the expression of the form $\operatorname{tr}(c\boldsymbol{I}\phi(\boldsymbol{x})\phi(\boldsymbol{x})') = c\,k(\boldsymbol{x},\boldsymbol{x})$. We note that Theorem 4 also generalizes easily from symmetric outer product instances to symmetric matrix instances with fixed rank s.

3 Kernelization via a Representer Theorem

The following Representer Theorem for asymmetric outer product instances was proven in [1]: Given a penalty function $\Omega(\boldsymbol{W}) = \sum_{i=1}^d s_i(\sigma_i(\boldsymbol{W}))$, where $\{\sigma_1, \ldots, \sigma_d\}$ is the set of singular values of $\boldsymbol{W}$ in decreasing order, and s_i are non-decreasing functions satisfying $s(0) = 0$, then there exists a solution to the minimization problem

$$\min_{\boldsymbol{W}} \quad \Omega(\boldsymbol{W}) \ + \ \eta\sum_t \operatorname{loss}_t(\operatorname{tr}(\boldsymbol{W}'\boldsymbol{x}_t\boldsymbol{y}_t')), \tag{4}$$

which can be written as $\boldsymbol{W} = \sum_{i,j}\boldsymbol{C}_{i,j}\boldsymbol{x}_i\boldsymbol{y}_j' = \boldsymbol{X}\boldsymbol{C}\boldsymbol{Y}'$. Using our results, we are able to prove a version of the Representer Theorem with different, not directly comparable assumptions:

Theorem 5. *Consider the minimization problem* $\min_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W},\boldsymbol{\mathcal{S}})$*, which for all* $\boldsymbol{\mathcal{S}}$ *has a unique solution and is* rotationally invariant*, i.e. for any* $\boldsymbol{\mathcal{S}}$ *and any orthogonal matrices* $\boldsymbol{U}$ *and* $\boldsymbol{V}$*,* $\mathcal{L}(\boldsymbol{W},\boldsymbol{\mathcal{S}}) = \mathcal{L}(\boldsymbol{UWV}',\boldsymbol{U\mathcal{S}V}')$*. In this case the solution* $\boldsymbol{W}^*(\boldsymbol{\mathcal{S}})$ *can be written as* $\boldsymbol{W}^*(\boldsymbol{\mathcal{S}}) = \boldsymbol{XCY}'$ *where* $\boldsymbol{C}$ *depends on* $\boldsymbol{\mathcal{S}}$ *only via the kernel matrices* $\boldsymbol{X}'\boldsymbol{X}$*,* $\boldsymbol{Y}'\boldsymbol{Y}$*.*

Proof. Let algorithm $\mathcal{A}$ produce the matrix $\boldsymbol{W}(\boldsymbol{\mathcal{S}}) := \boldsymbol{W}^*(\boldsymbol{\mathcal{S}})$. The algorithm satisfies our definition of linearity. fIt is also kernelizable, because due to rotational invariance of $\mathcal{L}$ and uniqueness of the solution, $\boldsymbol{W}^*(\boldsymbol{U\mathcal{S}V}') = \boldsymbol{UW}^*(\boldsymbol{\mathcal{S}})\boldsymbol{V}'$ and thus $\boldsymbol{W}(\boldsymbol{U\mathcal{S}V}') = \boldsymbol{UW}(\boldsymbol{\mathcal{S}})\boldsymbol{V}'$, so that for any $\boldsymbol{xy}'$,

$$\mathrm{tr}(\boldsymbol{W}(\boldsymbol{\mathcal{S}})'\boldsymbol{xy}') = \mathrm{tr}(\boldsymbol{V}'\boldsymbol{W}(\boldsymbol{U\mathcal{S}V}')'\boldsymbol{Uxy}') = \mathrm{tr}(\boldsymbol{W}(\boldsymbol{U\mathcal{S}V}')'\boldsymbol{UxyV}').$$

The theorem now follows from the forward direction of Theorem 3. □

We also note that with some more effort, it is possible to prove Theorem 5 under the weaker assumption that $\min_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W},\boldsymbol{\mathcal{S}})$ has a unique solution only for a particular sequence $\boldsymbol{\mathcal{S}}$, rather than for all sequences $\boldsymbol{\mathcal{S}}$.

The problem (4) is rotationally invariant (because Ω is a function of the singular values only), so the Theorem 5 applies as long as the solution is unique. Note that [1] specify different conditions: no uniqueness assumption is needed, rather some structure of the penalty function is imposed. Therefore our conditions are not directly comparable with theirs. Our proof of the Representer Theorem is however much simpler than the proof in [1]. Also, our conditions apply to a much wider class of algorithms, which does not need be defined as solution to the optimization problem above. Moreover, using our approach it is straightforward to generalize the Representer Theorem to the optimization problem with constraints, as long as the constraints are rotationally invariant and the solution is unique. Finally, we easily get the version of the Representer Theorem for the case of symmetric outer products, which has not be considered elsewhere:

Theorem 6. *Consider the problem* $\min_{\mathrm{sym.}\boldsymbol{W}} \mathcal{L}(\boldsymbol{W},\boldsymbol{\mathcal{S}})$*, which for all* $\boldsymbol{\mathcal{S}}$ *has a unique solution and is* rotationally invariant*, i.e. for any* $\boldsymbol{\mathcal{S}}$ *and any orthogonal matrix* $\boldsymbol{U}$*,* $\mathcal{L}(\boldsymbol{W},\boldsymbol{\mathcal{S}}) = \mathcal{L}(\boldsymbol{UWU}',\boldsymbol{U\mathcal{S}U}')$*. In this case the solution* $\boldsymbol{W}^*(\boldsymbol{\mathcal{S}})$ *can be written as* $\boldsymbol{W}^*(\boldsymbol{\mathcal{S}}) = \boldsymbol{XCX}' + c\boldsymbol{I}$*, where* $\boldsymbol{C}$ *and* c *depend on* $\boldsymbol{\mathcal{S}}$ *only via the kernel matrix* $\boldsymbol{X}'\boldsymbol{X}$*.*

4 Example Applications

We provide a few examples of how the arguments given in this paper can shed light on the kernelization of algorithms for particular learning problems. We focus on the online setting, i.e. when the instances are revealed sequentially to the learner. We also give algorithms only for the matrix case (both asymmetric and symmetric), as the vector case has been much exploited in the last decades, mostly in connection to support vector machines.

The algorithms of this section require the use of the singular value decomposition of the matrix $\boldsymbol{XCV}'$, or the eigenvalue decomposition (in the symmetric case) of the symmetric matrix $\boldsymbol{XCX}'$. As discussed in the introduction, the dimensions n and m of the left instances $\boldsymbol{x}_i$ and the right instances $\boldsymbol{y}_i$, respectively, are typically much larger than the number of instances T. Thus the dimension of the matrix $\boldsymbol{XCY}' \in \mathbb{R}^{n\times m}$ (or $\boldsymbol{XCX}' \in \mathbb{R}^{n\times n}$) is too large. The key is to obtain its decomposition in terms of the smaller kernel matrices $\boldsymbol{X}'\boldsymbol{X}, \boldsymbol{Y}'\boldsymbol{Y} \in \mathbb{R}^{T\times T}$:

Lemma 2. *For any left instance set $\boldsymbol{X} \in \mathbb{R}^{n\times T}$, right instance set $\boldsymbol{Y} \in \mathbb{R}^{m\times T}$ and square matrix $\boldsymbol{C} \in \mathbb{R}^{T\times T}$, if $\boldsymbol{U\Sigma V}'$ is a compact SVD of $\sqrt{\boldsymbol{X}'\boldsymbol{X}}\boldsymbol{C}\sqrt{\boldsymbol{Y}'\boldsymbol{Y}}$, where $\boldsymbol{\Sigma} = diag(\sigma_1, \cdots, \sigma_r)$, then the compact SVD of $\boldsymbol{XCY}'$ is $\tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\tilde{\boldsymbol{V}}$ with $\tilde{\boldsymbol{U}} = \boldsymbol{XC}\sqrt{\boldsymbol{Y}'\boldsymbol{Y}}\boldsymbol{V\Sigma}^{-1}$ and $\tilde{\boldsymbol{V}} = \boldsymbol{YC}'\sqrt{\boldsymbol{X}'\boldsymbol{X}}\boldsymbol{U\Sigma}^{-1}$. Similarly, for any $\boldsymbol{X} \in \mathbb{R}^{n\times T}$ and symmetric matrix $\boldsymbol{C} \in \mathbb{R}^{T\times T}$, if $\boldsymbol{U\Sigma U}'$ is a compact eigendecomposition of $\sqrt{\boldsymbol{X}'\boldsymbol{X}}\boldsymbol{C}\sqrt{\boldsymbol{X}'\boldsymbol{X}}$, where $\boldsymbol{\Sigma} = diag(\sigma_1, \cdots, \sigma_r)$, then the compact eigendecomposition of $\boldsymbol{XCX}'$ is $\tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\tilde{\boldsymbol{U}}'$ with $\tilde{\boldsymbol{U}} = \boldsymbol{XC}\sqrt{\boldsymbol{X}'\boldsymbol{X}}\boldsymbol{U\Sigma}^{-1}$.*

The proof (omitted) consists of checking the orthogonality of $\tilde{\boldsymbol{U}}$ and $\tilde{\boldsymbol{V}}$ and showing that $\boldsymbol{XCY}' = \tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\tilde{\boldsymbol{V}}$. For symmetric instances, a particularly simple case is obtained when $\boldsymbol{C} = \boldsymbol{I}$:

Corollary 1. *For any $\boldsymbol{X} \in \mathbb{R}^{n\times T}$, if $\boldsymbol{U\Sigma U}'$ is a compact eigendecomposition of $\boldsymbol{X}'\boldsymbol{X}$, then $\boldsymbol{XX}'$ has the compact eigendecomposition $\widetilde{\boldsymbol{U}}\boldsymbol{\Sigma}\widetilde{\boldsymbol{U}}'$, where $\widetilde{\boldsymbol{U}} = \boldsymbol{XU\Sigma}^{-1/2}$.*

This known fact was key to the kernelization of PCA and Fisher Linear Discriminant Functions [12,10].

Asymmetric Case and Additive Updates: Consider the following online learning problem: The data $\{(\boldsymbol{x}_t\boldsymbol{y}_t', \ell_t)\}_{t=1}^T$ is revealed to the learner sequentially. The learner predicts at trial t with a matrix $\boldsymbol{W}_t \in \mathcal{W}$ from some convex set $\mathcal{W}$, and suffers a convex loss denoted as $\text{loss}(\text{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{y}_t'), \ell_t)$. The goal of the learner is to have total loss in trials $t = 1, \dots, T$ not much higher then the total loss of the best matrix $\boldsymbol{W}^* \in \mathcal{W}$ chosen in hindsight, i.e. to have small regret

$$\text{Reg}(\boldsymbol{\mathcal{S}}) = \sum_t \text{loss}(\text{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{y}_t'), \ell_t) - \min_{\boldsymbol{W}\in\mathcal{W}} \sum_t \text{loss}(\text{tr}(\boldsymbol{W}'\boldsymbol{x}_t\boldsymbol{y}_t'), \ell_t).$$

Assume that $\mathcal{W} = \{\boldsymbol{W} \colon \|\boldsymbol{W}\| \le B\}$, where $\|\boldsymbol{W}\|$ is a rotationally invariant norm, i.e. depends on $\boldsymbol{W}$ only via its singular values. A typical choice, used e.g. in collaborative filtering, would be the trace norm $\|\boldsymbol{W}\|_1$. Let us also assume for simplicity that $\|\boldsymbol{x}_t\|_2 \le 1$ and $\|\boldsymbol{y}_t\|_2 \le 1$ for all t, where $\|\cdot\|_2$ is the Euclidean norm. A popular approach to solve the minimization problem is the online gradient descent (GD) [7]: Let $\partial_t(\boldsymbol{W})$ denote the subgradient $\partial_{\hat{\ell}_t}\text{loss}(\hat{\ell}_t, \ell_t)$ at $\hat{\ell}_t = \text{tr}(\boldsymbol{W}'\boldsymbol{x}_t\boldsymbol{y}_t')$. The GD step can be derived as the solution to the following optimization problem:

$$\boldsymbol{W}_{t+1} = \underset{\boldsymbol{W}\in\mathcal{W}}{\text{argmin}} \;\; \|\boldsymbol{W} - \boldsymbol{W}_t\|_F^2 + \eta\partial_t(\boldsymbol{W}_t)\,\text{tr}(\boldsymbol{W}'\boldsymbol{x}_t\boldsymbol{y}_t'), \tag{5}$$

where $\|\cdot\|_F$ is the Frobenious norm. Solving (5) leads to the *additive update*:

$$\boldsymbol{W}_{t+1} = \operatorname{proj}\left(\boldsymbol{W}_t - \eta\partial_t(\boldsymbol{W}_t)\boldsymbol{x}_t\boldsymbol{y}_t'\right),$$

where the projection operation is defined as $\operatorname{proj}(\boldsymbol{W}) = \operatorname{argmin}_{\|\widetilde{\boldsymbol{W}}\|\leq B} \|\boldsymbol{W} - \widetilde{\boldsymbol{W}}\|_F^2$. Since the norm $\|\cdot\|$ is rotationally invariant, the projection becomes a projection on the singular values $\{\sigma_1, \ldots, \sigma_{\min\{n,m\}}\}$ of $\boldsymbol{W}$. In particular, for $\|\cdot\|$ being the trace norm, the projection leads to $\sigma_i \mapsto (\sigma_i - \tau)_+$, where τ is the smallest value for which $\sum_i(\sigma_i - \tau)_+ \leq B$. When $\boldsymbol{W}_1 = \boldsymbol{0}$, one can show by a simple induction that the problem (5) is rotationally invariant for all t. Due to the strictly convex objective function, (5) has a unique solution, and we conclude from Theorem 5 that $\boldsymbol{W}_t$ is in the span of the data, i.e. has the form $\boldsymbol{XCY}'$. Thus the algorithm can be kernelized by calculating the SVD of the matrices $\boldsymbol{XCY}'$ i.t.o. of the kernel matrices using Lemma 2. Also the output $\operatorname{tr}(\boldsymbol{XCY}'\boldsymbol{xy}') = x'\boldsymbol{XCY}'\boldsymbol{y}$ only relies on the kernel matrices. For the trace norm, it can be shown using a standard analysis of GD, that given $|\partial_t(\boldsymbol{W})| \leq G$, $\operatorname{Reg}(\boldsymbol{\mathcal{S}}) \leq BG\sqrt{T}$, and is independent of the dimension of the feature space[4] [13].

Symmetric Case and Multiplicative Updates: In the symmetric case, the data sequence becomes $\{(\boldsymbol{x}_t\boldsymbol{x}_t', \ell_t)\}_{t=1}^T$. Let us assume for simplicity that $\|\boldsymbol{x}_t\|_2 = 1$ for all t. The learner predicts at trial t with the symmetric matrix $\boldsymbol{W}_t \in \mathcal{W}$, and suffers loss $\operatorname{loss}(\operatorname{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{x}_t'), \ell_t)$. We focus on the interesting case when $\mathcal{W}$ is a set of positive-semidefinite matrices with unit trace (*density* matrices), a generalization of the probability simplex to symmetric matrices. A choice of the algorithm is the Matrix Exponentiated Gradient [14], defined as a trade-off between minimization of the quantum relative entropy and the negative gradient of the loss:

$$\boldsymbol{W}_{t+1} = \operatorname*{argmin}_{\boldsymbol{W}\in\mathcal{W}} \operatorname{tr}\left(\boldsymbol{W}\left(\mathbf{log}\,\boldsymbol{W} - \mathbf{log}\,\boldsymbol{W}_t\right)\right) + \eta\partial_t(\boldsymbol{W}_t)\operatorname{tr}(\boldsymbol{W}'\boldsymbol{x}_t\boldsymbol{x}_t'), \qquad (6)$$

which leads to to the following *multiplicative update* [14]:

$$\boldsymbol{W}_{t+1} = \frac{\mathbf{exp}\left(\mathbf{log}\,\boldsymbol{W}_t - \eta\partial_t(\boldsymbol{W}_t)\boldsymbol{x}_t\boldsymbol{x}_t'\right)}{Z_t}, \qquad (7)$$

where $Z_t = \operatorname{tr}\left(\mathbf{exp}\left(\mathbf{log}\,\boldsymbol{W}_t - \eta\partial_t(\boldsymbol{W}_t)\boldsymbol{x}_t\boldsymbol{x}_t'\right)\right)$ is the normalization factor. When $\boldsymbol{W}_1 = \boldsymbol{I}/n$, a simple inductive argument proves rotational invariance of (6) for all t. Due to the strictly convex objective function, (6) has a unique solution, and we conclude from Theorem 6 that the algorithm can be kernelized (we note that the standard representer theorems do not cover this case). The main challenge in the update (7) is to do the **exp** operation, but it can be done by eigendecomposition of $\mathbf{log}\,\boldsymbol{W}_t - \eta\partial_t(\boldsymbol{W}_t)\boldsymbol{x}_t\boldsymbol{x}_t'$, which by Lemma 2 only requires to calculate the kernel matrix.

A particularly interesting case is when $\operatorname{loss}(\operatorname{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{x}_t'), \ell_t) = -\operatorname{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{x}_t')$. In other words, the game is the *gain game* with a linear gain function $\operatorname{tr}(\boldsymbol{W}_t\boldsymbol{x}_t\boldsymbol{x}_t')$. Then, the offline solution to the problem $\boldsymbol{W}^*$ is a one-dimensional projector to the subspace that captures the most of the variance of the data, i.e. the subspace

[4] In practical applications the choice of B may still depend on the dimension.

associated with the largest eigenvalue of $\sum_t \boldsymbol{x}_t \boldsymbol{x}_t'$. This is exactly the problem of single-component PCA[5] [17]. In this case, the EG update (7) simplifies to $\boldsymbol{W}_{t+1} = Z_t^{-1} \mathbf{exp}\left(\eta \sum_{i=1}^{t} \boldsymbol{x}_i \boldsymbol{x}_i'\right)$, and the eigendecomposition can be handled using Corollary 1.

By modifying the EG analysis of [17,9], we can show shown that $\mathrm{Reg}(\boldsymbol{\mathcal{S}}) \le \sqrt{2L^* \ln n} + \ln n$, where L^* is the *approximation error*, i.e. part of the variance in the data not captured by $\boldsymbol{W}^*$, $L^* = \min_{\boldsymbol{W} \in \mathcal{W}} \left\{\sum_{t=1}^{T} (1 - \mathrm{tr}(\boldsymbol{W}' \boldsymbol{x}_t \boldsymbol{x}_t'))\right\}$. Unfortunately, this bound (which essentially appears in [9]) is not satisfactory, as it depends on the feature space dimension n. When the instances $\boldsymbol{x}\boldsymbol{x}'$ are replaced by $\phi(\boldsymbol{x})\phi(\boldsymbol{x})'$ then the $\ln n$ term can become unbounded. Below we sketch a new method for replacing $\ln n$ by $\ln r$, where are is the total rank of the instances. So for the first time, we obtain a bound for a Matrix EG algorithm that does not depend on the feature dimension.

We observe that the best density matrix in hindsight $\boldsymbol{W}^*$ projects into the span of the data. If we knew the span in hindsight, we could disregard the other dimensions and play EG within this subspace, achieving the bound $\sqrt{2L^* \ln r} + \ln r$, where r is the dimension of the subspace, i.e. the rank of the kernel matrix $\boldsymbol{X}'\boldsymbol{X}$. This bound is independent on n, as $r \le T$. Of course, the data span is unknown to the learner, but we can slightly modify the EG algorithm (let us call the modification EG$^+$) to obtain the bound $\sqrt{2L^* \ln r} + \ln r + 1 \le \sqrt{2L^* \ln T} + \ln T + 1$ without any prior knowledge of the span. The EG$^+$ algorithm is defined by modifying the update (7) to $\boldsymbol{W}_t^+ = \left(Z_t^+\right)^{-1} \mathbf{exp}^+\left(\eta \sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i'\right)$, where $Z_t^+ = \mathrm{tr}\left(\mathbf{exp}^+\left(\eta \sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i'\right)\right)$, and $\mathbf{exp}^+(\boldsymbol{A})$ is a function that exponentiate the positive eigenvalues of $\boldsymbol{A}$ only, and leaves the zero eigenvalues unchanged.[6] In other words if $\boldsymbol{A}$ has a compact eigenvalue decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}'$, then, $\mathbf{exp}^+(\boldsymbol{A}) = \boldsymbol{U}\,\mathbf{exp}(\boldsymbol{\Sigma})\boldsymbol{U}'$. To prove the regret bound $\sqrt{2L^* \ln r} + \ln r + 1$ for EG$^+$, it suffices to show that given a feature space with dimension n, the total loss of EG$^+$ (which does not know n) is by at most one larger than the total loss of EG (which knows n):

Lemma 3. *Let* $\boldsymbol{W}_t$ *and* $\boldsymbol{W}_t^+$ *be the matrices produced by the EG and EG*$^+$ *algorithms, respectively. Then* $\sum_{t=1}^{T} -\mathrm{tr}(\boldsymbol{W}_t^{+\prime} \boldsymbol{x}_t \boldsymbol{x}_t') - \sum_{t=1}^{T} -\mathrm{tr}(\boldsymbol{W}_t' \boldsymbol{x}_t \boldsymbol{x}_t') \le 1$.

Proof. Fix iteration t and let $\boldsymbol{S}_{t-1} := \sum_{i=1}^{t-1} \boldsymbol{x}_i \boldsymbol{x}_i'$. If $\boldsymbol{x}_t$ is a linear combination of past instances $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{t-1}$, then the loss incurred by EG$^+$ is smaller than the loss incurred by EG. Indeed, $\mathrm{tr}\left(\mathbf{exp}^+\left(\eta \boldsymbol{S}_{t-1}\right) \boldsymbol{x}_t \boldsymbol{x}_t'\right) = \mathrm{tr}\left(\mathbf{exp}\left(\eta \boldsymbol{S}_{t-1}\right) \boldsymbol{x}_t \boldsymbol{x}_t'\right)$ (because $\boldsymbol{x}_t$ belongs to the subspace associated with non-zero eigenvalues of $\boldsymbol{S}_{t-1}$), but $Z_t^+ \le Z_t$ (because $\mathbf{exp}^+(\boldsymbol{A}) \preceq \mathbf{exp}(\boldsymbol{A})$ for any positive matrix $\boldsymbol{A}$). If $\boldsymbol{x}_t$ is linearly independent of $\boldsymbol{x}_1, \dots, \boldsymbol{x}_{t-1}$, then the loss incurred by EG$^+$ in any trial $t > 1$ is larger by at most $\frac{1}{n}$ (and $t = 1$ can be handled seperately):

[5] By capping the eigenvalues to $\frac{1}{k}$ (as done in [17]) we can generalize this algorithm to k-component PCA where one seeks a k-dimensional subspace with maximal variance.

[6] The initial weight matrix $\boldsymbol{W}_1$ is set arbitrarily.

$$
\begin{aligned}
-\mathrm{tr}(\boldsymbol{W}_t^{+'}\boldsymbol{x}_t\boldsymbol{x}_t') &= -(Z_t^{+})^{-1}\mathrm{tr}\left(\mathbf{exp}^{+}\left(\eta\boldsymbol{S}_{t-1}\right)\boldsymbol{x}_t\boldsymbol{x}_t'\right)\\
&\leq -Z_t^{-1}\mathrm{tr}\left(\mathbf{exp}^{+}\left(\eta\boldsymbol{S}_{t-1}\right)\boldsymbol{x}_t\boldsymbol{x}_t'\right)\\
&\leq -Z_t^{-1}\mathrm{tr}\left(\left(\mathbf{exp}\left(\eta\boldsymbol{S}_{t-1}\right)-\boldsymbol{I}\right)\boldsymbol{x}_t\boldsymbol{x}_t'\right)\\
&= -\mathrm{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{x}_t') + Z_t^{-1}\\
&\leq -\mathrm{tr}(\boldsymbol{W}_t'\boldsymbol{x}_t\boldsymbol{x}_t') + 1/n,
\end{aligned}
$$

where we used the fact that $\mathbf{exp}^{+}(\boldsymbol{A}) \succeq \mathbf{exp}(\boldsymbol{A}) - \boldsymbol{I}$ for any positive matrix $\boldsymbol{A}$, and that $Z_t = \mathrm{tr}\left(\mathbf{exp}\left(\eta\sum_{i=1}^{t-1}\boldsymbol{x}_i\boldsymbol{x}_i'\right)\right) \geq \mathrm{tr}(\boldsymbol{I}) = n$. □

Note that the EG$^+$ is as easy to kernelize as the EG, because they differ only in the update of the eigenvalues. We can also easily handle the case when the instances are positive symmetric matrices of rank at most s. Since the EG bound does not depend on the sparsity of the instances, we immediately get the same regret bound $\sqrt{2L^* \log r} + \ln r$, where $r \leq Ts$.

We finally note that one can also use an additive update (GD) algorithm in the symmetric case, and obtain the bound $\sqrt{T}$ for outer product instances, and $\sqrt{Ts}$ for matrix instances. The bounds for the GD and the EG$^+$ algorithms are not directly comparable: EG$^+$ has an additional $\log r$ factor, but GD scales worse with the rank s of matrix instances. Moreover, the EG$^+$ bound is especially useful for low-noise conditions, when the approximation error L^* is small. There is no corresponding bound known for the GD in this case.

5 Conclusion

We gave necessary and sufficient conditions for kernelizability for the case of vector, asymmetric matrix, and symmetric matrix instances, under the assumption that the algorithm is linear, produces a unique solution and satisfies a certain rotational invariance. We also proved simple representer theorems for both asymmetric and symmetric matrix instances, and gave a number of examples of our methods, including the kernelization of multiplicative updates. In some sense our approach resembles how the models in Physics are built, where the equations of motion follow from certain invariance properties of physical laws.

We conclude with a subtle open problem. A new family of so called “Forward” algorithms was developed [3] whose predictions may depend on the current unlabeled instance for which the algorithm is to produce a label. In particular in the case of linear regression [15,6], better regret bounds were proven for the Forward algorithm than for the standard Ridge Regression algorithm. Therefore a natural open problem is whether our characterization of kernelizability can be generalized to algorithms that may predict with linear combinations of the labeled as well as the last unlabeled instance.

References

1. Abernethy, J., Bach, F., Evgeniou, T., Vert, J.P.: A new approach to collaborative filtering: Operator estimation with spectral regularization. Journal of Machine Learning 10, 803–826 (2009)

2. Argyriou, A., Micchelli, C.A., Pontil, M.: When is there a representer theorem? vector versus matrix regularizers. Journal of Machine Learning Research 10, 2507–2529 (2009)
3. Azoury, K., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Journal of Machine Learning 43(3), 211–246 (2001)
4. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proc. 5th Annual ACM Workshop on Comput. Learning Theory, pp. 144–152. ACM Press, New York (1992)
5. Cavallanti, G., Cesa-Bianchi, N., Gentile, C.: Linear algorithms for online multitask classification. In: Proceedings of the 21st Annual Conference on Learning Theory (COLT 2008), pp. 251–262 (July 2008)
6. Forster, J.: On Relative Loss Bounds in Generalized Linear Regression. In: Ciobanu, G., Păun, G. (eds.) FCT 1999. LNCS, vol. 1684, pp. 269–280. Springer, Heidelberg (1999)
7. Herbster, M., Warmuth, M.K.: Tracking the best linear predictor. Journal of Machine Learning Research 1, 281–309 (2001)
8. Kimeldorf, G.S., Wahba, G.: Some results on Tchebycheffian spline functions. J. Math. Anal. Applic. 33, 82–95 (1971)
9. Kuzmin, D., Warmuth, M.K.: Online Kernel PCA with entropic matrix updates. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007). ACM International Conference Proceedings Series, pp. 465–471 (June 2007)
10. Mika, S., Ratsch, G., Weston, J., Schölkopf, B., Mullers, K.R.: Fisher discriminant analysis with kernels. In: Proc. NNSP 1999. IEEE Signal Processing Society Workshop, pp. 41–48 (1999)
11. Schölkopf, B., Herbrich, R., Smola, A.J.: A Generalized Representer Theorem. In: Helmbold, D.P., Williamson, B. (eds.) COLT/EuroCOLT 2001. LNCS (LNAI), vol. 2111, pp. 416–426. Springer, Heidelberg (2001)
12. Schölkopf, B., Smola, A.J., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10(5), 1299–1319 (1998)
13. Srebro, N., Sridharan, K., Tewari, A.: On the universality of online mirror descent. In: Advances in Neural Information Processing Systems 23 (NIPS 2011), pp. 2645–2653 (2011)
14. Tsuda, K., Rätsch, G., Warmuth, M.K.: Matrix exponentiated gradient updates for on-line learning and Bregman projections. Journal of Machine Learning Research 6, 995–1018 (2005)
15. Vovk, V.: Competitive on-line statistics. International Statistical Review 69, 213–248 (2001)
16. Warmuth, M.K.: Winnowing subspaces. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), ACM Press (June 2007)
17. Warmuth, M.K., Kuzmin, D.: Randomized PCA algorithms with regret bounds that are logarithmic in the dimension. Journal of Machine Learning Research 9, 2217–2250 (2008)
18. Warmuth, M.K., Vishwanathan, S.V.N.: Leaving the Span. In: Auer, P., Meir, R. (eds.) COLT 2005. LNCS (LNAI), vol. 3559, pp. 366–381. Springer, Heidelberg (2005); Journal version in progress

Predictive Complexity and Generalized Entropy Rate of Stationary Ergodic Processes

Mrinalkanti Ghosh and Satyadev Nandakumar

Department of Computer Science and Engineering,
Indian Institute of Technology Kanpur,
Kanpur, U.P., India.

Abstract. In the online prediction framework, we use generalized entropy to study the loss rate of predictors when outcomes are drawn according to stationary ergodic distributions over the binary alphabet. We show that the notion of generalized entropy of a regular game [11] is well-defined for stationary ergodic distributions. In proving this, we obtain new game-theoretic proofs of some classical information theoretic inequalities. Using Birkhoff's ergodic theorem and convergence properties of conditional distributions, we prove that a generalization of the classical Shannon-McMillan-Breiman theorem holds for a restricted class of regular games, when no computational constraints are imposed on the prediction strategies.

If a game is mixable, then there is an optimal aggregating strategy which loses at most an additive constant when compared to any other lower semicomputable strategy. The loss incurred by this algorithm on an infinite sequence of outcomes is called its *predictive complexity*. We prove that when a restricted regular game has a predictive complexity, the average predictive complexity converges to the generalized entropy of the game almost everywhere with respect to the stationary ergodic distribution.

1 Introduction

We consider the online prediction question studied by Vovk and Watkins [17], Vyugin and V'yugin [18], Kalnishkan et. al. [11] [10], Fortnow and Lutz [8], in the setting of a stationary stochastic process. In this setting, we have a sequence of outcomes $x_0, x_1, \dots$ from a finite alphabet. A predictor, given the history up to a certain index, predicts what the next outcome will be. We allow the predictor to present its prediction as an element of a compact space. The game proceeds by revealing the next outcome, and then asking for the prediction of the future outcome. For an overview of this area, see Cesa-Bianchi and Lugosi [2]. Independently, a related question has been widely studied in information theory (see for example, Merhav and Feder [14], Feder [5] and Feder et. al. [6]) – this is the question of universal predictors with respect to Shannon entropy, over a family of stationary ergodic distributions. It is known that the log-loss game characterizes Shannon entropy. The present line of work contrasts with this in

N.H. Bshouty et al. (Eds.): ALT 2012, LNAI 7568, pp. 365–379, 2012.

two ways – first, in considering loss functions besides log-loss, and second, in considering optimal predictors over given stationary ergodic distributions.

A natural question in this context is how well the predictor is doing as the game progresses. We measure the discrepancy between the actual outcome and the predicted one, with a *loss function*. This helps us to ask whether *optimal* predictors exist – those which incur at most the same loss as as any other predictor on any outcome, ignoring additive constants. Indeed if such an optimal predictor exists, we can use its loss rate on a particular sequence of outcomes to define its inherent *predictability* (see for example, Vovk and Watkins [17], Vyugin and V'yugin [18]).

Besides competitive advantage above other predictors, we can also characterize the performance of an optimal predictor by examining its expected loss assuming the outcomes are drawn from a particular distribution. Prior work by Kalnishkan et al. [11] establishes that if the outcomes are drawn independently according to a Bernoulli distribution on the alphabet, then the loss rate of an optimal predictor on almost-every infinite sequence of outcomes is the *generalized entropy* (Grünwald and Dawid [9]) of the loss function. In this paper, we extend this result to the important setting of stationary ergodic distributions.

The contributions of our paper are threefold.

1. First, we show that the generalized entropy rate of a stationary ergodic process is well-defined, if the game is *regular*. We provide "game-theoretic" proofs of classical information-theoretic inequalities, giving new intuitive proofs even in the special case of the Shannon entropy. This constitutes sections 3 and 4 of the paper.
2. Second, under a continuity and an integrability constraint, we show that optimal strategies exist for regular games.[1] We show that the loss rate incurred by such a strategy is the generalized entropy rate of the stationary ergodic process almost-everywhere. This is a Shannon-McMillan-Breiman theorem for generalized entropy. This result is new, and we provide a proof using Vitali Convergence. This constitutes section 5 of the paper. The Shannon-McMillan-Breiman Theorem deals directly with optimal processes on infinite sequences.
3. Using the above results, we show that when a game has *predictive complexity*, an optimal aggregator algorithm attains the entropy rate of the game. This result deals with limiting loss rate made by an optimal strategy on finite strings.

   The proof that the aggregator incurs at most the entropy rate of loss uses the Ergodic Theorem.

   The proof that the aggregator incurs at least the entropy rate of loss uses some properties of stationary ergodic processes that we prove in Sections 3 and 4. This constitutes the final section of the paper.

[1] There is an independent characterization of games with optimal strategies in terms of convexity of loss-regions (see Kalnishkan et. al. [10]). We deal with this approach in the final section of our paper.

2 Preliminaries

As defined in Kalnishkan et. al.[11], a game $\mathcal{G}$ is a triple $(\Sigma, \Gamma, \lambda)$ where Σ is a finite alphabet space, Γ is the space of predictions and $\lambda : \Sigma \times \Gamma \to [0, \infty]$ is the loss function, to be defined below. We will only consider the binary alphabet in this paper, and the sample space is Σ^∞, the space of infinite binary sequences.

Intuitively, we model a predictor function which, given the string of outcomes so far, will predict the next outcome. We consider a slightly general framework where the predictor is allowed to output a point γ in a compact set Γ. The game proceeds by revealing the next outcome. Let this outcome be b. The prediction strategy is said to incur the loss $\lambda(b, \gamma)$.

As is customary, we adopt the notation $\mathbb{N}$ for the set of natural numbers, starting from 0. The set of strings of length n is denoted Σ^n. The set of finite binary strings is denoted Σ^* and the set of infinite binary sequences is denoted Σ^∞. For a finite or an infinite sequence x, the notation x_i^j denotes $x_i \dots x_j$. If x is shorter than n bits, x_0^{n-1} denotes x itself. If x is a finite string, and ω is a finite string or an infinite sequence, then $x \cdot \omega$ denotes the result of concatenating ω to x. For each natural number i, let Π^i be the class of all functions mapping i-long strings to Γ.

We call a family of functions $\wp$ a *strategy* if $\forall i \in \mathbb{N}, |\wp \cap \Pi^i| = 1$, i.e, there is a unique function which takes an i-length string as input and produces a prediction based on the input. We call that function $\wp^i$. Thus the prediction strategy is a non-uniform family. We impose no computational constraints until the final part of the paper.

3 Loss Functions

The generalized entropy of a game is defined in terms of loss functions described above. We define the losses incurred by a strategy on a finite string w of outcomes, as the cumulative loss that it incurs on each bit of w. This follows the definition given in Kalnishkan et. al. [11] and [10]. We generalize the notion slightly to deal with the expected loss that a strategy incurs with respect to a stationary distribution.

Definition 1. *The* loss *that a prediction strategy $\wp$ incurs on a finite string w of outcomes is defined to be $Loss(w, \wp) = \sum_{i=0}^{|w|-1} \lambda(w_i, \wp^i(w_0^{i-1}))$.*

In order to study when a strategy is better than another, we study the average loss it incurs, when outcomes are drawn from a stationary distribution. We consider the strategy which incurs the minimal expected loss on a particular set, if such a strategy exists. Let $(\Sigma^\infty, \mathcal{F}, P)$ be the probability space where $\mathcal{F}$ is the Borel σ-algebra generated by cylinders $C_x = \{\omega \in \Sigma^\infty \mid x \text{ is a prefix of } \omega\}$ for all finite strings x, and $P : \mathcal{F} \to [0, 1]$ is the probability measure.

Let $\overline{X} = (X_0, X_1, \dots)$ be a sequence of random variables on the probability space – for each $i \in \mathbb{N}$, X_i maps Σ^∞ to $\mathbb{R}$. For $k \geq 0$, let $S_k(\overline{X})$ denote the sequence $(X_k, X_{k+1}, \dots)$ – that is, X "shifted left" k times.

Definition 2. *[15] A sequence of random variables $\overline{X}$ is* stationary *if the distributions of $S_k X$ and X coincide for every $k \geq 0$. That is, for every Borel set B in the σ-algebra over $\mathbb{R}^\infty$, $P(X \in B) = P(S_k(\overline{X}) \in B)$.*

We could also use the terminology of measure-preserving transformations to capture stationarity. A transformation $T : \Omega \to \Omega$ is said to be *measure-preserving* if for every measurable set A, $P(T^{-1}A) = P(A)$. A measure-preserving transformation is said to be *ergodic* if $T^{-1}(A) = A$ if and only if $P(A)$ is either 0 or 1. (see, for example, Billingsley [1])

The class of stationary processes corresponds almost exactly to the class of probability spaces $(\Omega, \mathcal{F}, P, T)$, where $T : \Omega \to \Omega$ is a P-measure-preserving transformation. For $k \in \mathbb{N}$, let T^k denote the iterated application of T, k times. It is easy to see that if T is measure preserving and X is a random variable, then $(X, X \circ T, X \circ T^2, \dots)$ is a stationary sequence. We also have the converse. [15] On an alphabet space, we are interested in the coordinate random variables $X_i(\omega) = \omega_i$ $(i \in \mathbb{N})$, and any probability distribution such that $\overline{X} = (X_0, X_1, \dots)$ is stationary with respect to it, will be called a *stationary distribution.* A probability space with respect to which the left-shift transformation is ergodic will be called an *ergodic distribution.*

Definition 3. *We define the n-step* generalized entropy *of the game to be $H_n = \inf_{\wp} \sum_{w \in \Sigma^n} P(w) Loss(w, \wp)$, where $(\Sigma^\infty, \mathcal{F}, P)$ is a probability space.*

In order to avoid degenerate games (for example, games where the least expected loss is infinity, precluding any incentive to play the game), following Kalnishkan et al. [11], we restrict the game in the following manner.

- We restrict Γ to be a compact space. For the binary alphabet space, we let the prediction space be $[0, 1]$.
- The loss function λ is an extended real-valued function on $\Sigma \times \Gamma$. For each bit b, $\lambda(b, .)$ is a convex function on Γ. We take the discrete topology on the alphabet and the standard topology on $[0, 1]$. Then λ is continuous with respect to their product topology.
- There is a prediction $\gamma \in \Gamma$ such that for every $b \in \Sigma$, the inequality $\lambda(b, \gamma) < \infty$ holds. This property ensures that the n-ary entropy is a finite quantity.
- If there are $\gamma \in \Gamma$ such that for some $b \in \Sigma$, the loss $\lambda(b, \gamma) = \infty$, then there is a sequence $\gamma_1, \gamma_2, \dots \to \gamma$ such that for each γ_i, we have $\lambda(b, \gamma_i) < \infty$.

A game which obeys these conditions is said to be *regular*. The last condition is necessary (but not sufficient) to ensure that predictive complexity exists for the game. We need this property crucially in Theorems 18 and 24.

The n step generalized entropy is the least expected loss incurred by any strategy, on Σ^n. Since Σ^n (from the compactness of Σ) and Γ are compact spaces and λ is continuous in both its arguments, the infimum in the above expression is attained by some strategy. [2]

[2] The authors remark in [11] that such a strategy need not exist for Σ^*.

Example 4. The Log-Loss game: Consider the binary alphabet and let predictions be values in [0,1]. Let p_0 and p_1 be the probability of the bit 0 and bit 1, respectively.

Suppose we define the loss function by $\lambda(b,\gamma) = -\log(|\,(1-b)-\gamma\,|)$, where b is a bit, and $\gamma \in [0,1]$. Then the minimal expected loss over one bit is obtained at $\gamma = p_1$, ensuring that H_1 is the Shannon entropy of the distribution. □

Definition 5. *The* generalized conditional entropy *of Σ^n given Σ^m is defined as*

$$H_{n|m} = \inf_{\wp} \sum_{w\cdot x\in\Sigma^{n+m}} P(w\cdot x) \sum_{i=0}^{m-1} \lambda\left(x_i, \wp^{i+m}(w\cdot x_0^{i-1})\right).$$

This is an analogue of the definition of conditional Shannon entropy.

When we generalize the theory to handle arbitrary loss functions, we do lose some ideal properties that Shannon entropy has. The following theorem states that Shannon entropy is the unique function having certain ideal properties that we desire in a measure of information (see Khinchin [12]).

Theorem 6. *For each $n \in \mathbb{N}$, suppose F_n is a continuous function mapping a probability vector $(p_1,\ldots,p_n)$ to $\mathbb{R}$ having the following properties.*

1. *For any finite set of disjoint events A and B, $F(A,B) = F(A) + F(B|A)$.* [3]
2. *For any n and probabilities $(p_1,\ldots,p_n)$, $F(p_1,\ldots,p_n)$ is maximal when $p_i = \frac{1}{n}$, $i = 1,\ldots,n$.*
3. *For any n and probabilities $(p_1,\ldots,p_n)$, $F(p_1,\ldots,p_n,0) = F(p_1,\ldots,p_n)$.*

Then there is a positive constant c such that for every n-dimensional probability vector $(p_1,\ldots,p_n)$, $H(p_1,p_2,\ldots,p_n) = cF(p_1,p_2,\ldots,p_n)$.

With our definition of the cumulative loss, we can establish the chain rule for generalized entropy.

Lemma 7. *For all positive natural numbers m and n, we have $H_{m+n} = H_m + H_{n|m}$.*

Proof. In Definition 5, $\wp^i$ for $0 \le i \le m$ does not play any role in the infimum and likewise in Definition 3, $\wp^i$ for $i \ge n$ does not play any role in the infimum inf. This observation allows us to deduce that

$$\begin{aligned} H_m + H_{n|m} &= \inf_{\wp}\left(\sum_{w\in\Sigma^m} P(w) \sum_{x\in\Sigma^n} P\{x|w\} \sum_{i=0}^{m-1} \lambda\left(x_i, \wp^{i+m}(w\cdot x_0^{i-1})\right)\right) + \\ &\qquad \inf_{\wp} \sum_{w\in\Sigma^m} P(w)\mathrm{Loss}(w,\wp) \\ &= \inf_{\wp} \sum_{w\in\Sigma^m} P(w)\left(\sum_{x\in\Sigma^n}\sum_{i=0}^{m-1} \lambda\left(x_i, \wp^{i+m}(w\cdot x_0^{i-1})\right) + \sum_{w\in\Sigma^m} \mathrm{Loss}(w,\wp)\right). \quad (1) \end{aligned}$$

[3] Khinchin [12]) describes them as "finite schemes".

Now,

$$
\begin{aligned}
&\inf_{\wp} \sum_{w\in\Sigma^m} P(w)\left(\mathrm{Loss}(w,\wp) + \sum_{x\in\Sigma^n} P\{x|w\} \sum_{i=0}^{m-1} \lambda(x_i, \wp^{i+m}(w\cdot x_0^{i-1}))\right)\\
=&\inf_{\wp} \sum_{w\in\Sigma^m} P(w) \sum_{x\in\Sigma^n} P\{x|w\} \left(\mathrm{Loss}(w,\wp) + \sum_{i=0}^{m-1} \lambda(x_i, \wp^{i+m}(w\cdot x_0^{i-1}))\right)\\
=&\inf_{\wp} \sum_{w\in\Sigma^{m+n}} P(w)\mathrm{Loss}(w,\wp) = H_{m+n}.
\end{aligned}
$$

□

Since λ is non-negative, it is clear that all entropies defined so far are non-negative. An immediate consequence of this is $H_{m+n} \geq H_m$ for all $m, n \geq 0$. We see that this style of proof referring to strategies in games yields new intuitive proofs of such inequalities.

Any generalization of Shannon entropy will result in violating one of the conditions of Khinchin's uniqueness theorem. Comparing with the Khinchin Uniqueness theorem, we see that in our approach, Lemma 7 does not ensure that for any finite events A and B, $H(A,B) = H(A) + H(B|A)$ for loss functions besides the log-loss function.

4 Entropy of a Regular Game

The goal of this section is to define the notion of the entropy of a regular game. Our idea is to define it to be the limiting rate of the n-step generalized entropies of the game. We now show that if the game is regular and the probability distribution is stationary, such a limit exists. Thus the notion of the entropy of a regular game is well-defined. We now prove that the *entropy rate* of a regular game is well-defined. First, we need a few technical lemmas that are used in Theorem 10 proving the existence of the entropy rate.

Lemma 8. *[Generalized Shannon Inequality] For any regular game, any stationary distribution P defined on it, and non-negative integers m and n, we have $H_{m|n} \leq H_m$.*

Proof. The following proof is for $m = 1$. In this special case $H_1 = \inf_{\gamma\in\Gamma} \sum_{a\in\Sigma} P(a)\lambda(a,\gamma)$ and

$$
H_{1|n} = \inf_{f\in\Pi^n} \sum_{w\in\Sigma^n} P(w) \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w)) = \inf_{f\in\Pi^n} \sum_{a\in\Sigma} P(a) \sum_{w\in\Sigma^n} P\{w|a\}\lambda(a, f(w))
$$

Now pick the $\gamma \in \Gamma$ which matches H_1. We can do this because the regularity condition of the game requires Γ to be compact. The loss function is continuous in both its arguments ensuring that the expected loss in (3) is a continuous function on a compact space. Now define $f' : \Sigma^n \to \{\gamma\}$. Clearly, $f' \in \Pi^n$. So,

$$\begin{aligned} H_{1|n} &\leq \sum_{a\in\Sigma} P(a) \sum_{w\in\Sigma^n} P\{w|a\}\lambda(a, f'(w)) = \quad \sum_{a\in\Sigma} P(a) \sum_{w\in\Sigma^n} P\{w|a\}\lambda(a,\gamma) \\ &= \sum_{a\in\Sigma} P(a)\lambda(a,\gamma) = H_1 \end{aligned}$$

The general case proceeds by induction by defining $f'^{i+n}(ww_0'^{i-1}) =$ $f^i(w_0'^{i-1})$, where w is an n-long string and $1 \leq i \leq m$. □

In the special case of the log-loss game with a Bernoulli distribution on the finite alphabet, the argument above yields a new argument for the Shannon inequality. The next lemma demonstrates that the conditional entropy is non-increasing with the length of the history we consider − this will be relevant in Theorem 10 to show that the limiting entropy rate exists.

Lemma 9. *For any regular game, any stationary distribution P defined on it, and any positive pair of natural numbers m and n, $H_{m|n} \geq H_{m|n+1}$.*

Proof. We prove the inequality for $m = 1$. The general case follows from application of Lemma 7. We have,

$$H_{1|n} = \inf_{f\in\Pi^n} \sum_{a\in\Sigma} \sum_{w\in\Sigma^n} P\{wa\}\lambda(a, f(w))$$

and similarly $H_{1|n+1} = \inf_{f'\in F^{n+1}} \sum_{a\in\Sigma} \sum_{w\in\Sigma^{n+1}} P\{wa\}\lambda(a, f'(w))$.

We show for each $f \in \Pi^n$ we have a $f' \in F^{n+1}$ which matches the inner quantity on which infimum is taken. Then, by taking infimum over F^{n+1},we will have $H_{1|k} \geq H_{1|k+1}$. Fix a $f \in \Pi^n$ and consider $f' \in F^{n+1}$ defined as $f'(bw) = f(w)$ for all $w \in \Sigma^n, b \in \Sigma$. Now,

$$\begin{aligned} \sum_{a\in\Sigma} \sum_{w\in\Sigma^{n+1}} P\{wa\}\lambda(a, f'(w)) &= \sum_{a\in\Sigma} \sum_{b\in\Sigma} \sum_{w'\in\Sigma^n} P\{bw'a\}\lambda(a, f'(bw')) \\ = \sum_{a\in\Sigma} \sum_{w'\in\Sigma^n} \sum_{b\in\Sigma} P\{bw'a\}\lambda(a, f(w')) &= \sum_{a\in\Sigma} \sum_{w'\in\Sigma^n} P\{w'a\}\lambda(a, f(w')) \end{aligned}$$

where the last step follows from stationarity of P (i.e, $\sum_{b\in\Sigma} P\{bw\} = P\{w\}$ for all $w \in \Sigma^n$). □

Theorem 10. *For any regular game $\mathcal{G}$ and stationary $(\Sigma^\infty, \mathcal{F}, P)$, $\lim_{n\to\infty} \dfrac{H_n}{n}$ exists and is finite.*

Proof. From the regularity condition, we get H_1 is finite. From Lemma 7, it follows that $H_n = \sum_{i=0}^{n-1} H_{1|i}$.

By Lemma 9, $H_{1|k} \geq H_{1|(k+1)}$. Since entropies are non-negative, the sequence $\{H_{1|n}\}$ is a bounded, decreasing sequence of reals. Hence, it has a limit which we denote by $H_{1|\infty}$. It also follows that $H_{1|\infty}$ is at most H_1.

So, $\lim\limits_{n\to\infty} \dfrac{H_n}{n} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} H_{1|i} = \lim_{n\to\infty} H_{1|n} = H_{1|\infty}$. □

Definition 11. *Let $\mathcal{G} = (\Sigma^\infty, \Gamma, \lambda)$ be a regular game and $(\Sigma^\infty, \mathcal{F}, P)$ be a stationary distribution. Then the* generalized entropy *of the game is defined as $H = \lim_{n\to\infty} \frac{H_n}{n}$.*

Thus the notion of entropy rate is well-defined, enabling us to investigate the existence of a Shannon-McMillan-Breiman Theorem.

5 A Shannon-McMillan-Breiman Theorem

We now show that for regular games with a suitable restriction on the loss functions, optimal processes exist and they attain the generalized entropy rate of the stationary ergodic process. Our approach to this result is through uniform integrability and the Vitali Convergence theorem, which contrasts with the usual approach using the Dominated Convergence Theorem. First, we define the notion of a *strongly regular game*, for which the result holds. [4] We will derive two consequences of strong regularity, *viz.*

1. The existence of a limiting function for the loss function, P-almost everywhere.
2. The integrability of this limiting function.

We utilize these in the proof of the Shannon-McMillan-Breiman Theorem. We conclude with two examples, illustrating that Theorem 18 properly generalizes the classical Shannon-McMillan-Breiman theorem.

Definition 12. *Let $(\Omega, \mathcal{F}, P)$ be a probability space. A sequence of functions $\{f_n\}_{n=1}^{\infty}$ is called* uniformly integrable *if*

$$\lim_{\alpha\to\infty} \sup_n \int |f_n| I_{[|f_n|>\alpha]} dP = 0, \tag{2}$$

where $I_{[|f_n|>\alpha]}$ is the indicator function which is 1 at points ω with $|f_n(\omega)| > \alpha$ and is 0 otherwise.

In addition to uniform integrability, we also need a continuity requirement over the space of strategies. We now introduce this. The next lemma characterizes $H_{1|n}$ in terms of the loss incurred by an optimal strategy on Σ^n. Lemma 13 lets us analyse loss incurred by some "optimal" strategy. From Lemma 13, we can see given $w \in \Sigma^n$, optimal loss depends on the conditional probability distribution $(P\{0|w\}, P\{1|w\})$.

[4] Kalnishkan et al. [10] consider the notion of mixable games, which characterize regular games with optimality. In comparison, our conditions are based on integrability of the loss function.

Lemma 13.

$$H_{1|n} = \inf_{f\in\Pi^n} \sum_{w\in\Sigma^n} P(w) \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w))$$
$$= \sum_{w\in\Sigma^n} P(w) \inf_{f\in\Pi^n} \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w))$$

Proof. Let n be an arbitrary number. For any string w of length n, $P(w) \geq 0$, thus it follows that

$$\inf_{f\in\Pi^n} \sum_{w\in\Sigma^n} \sum_{a\in\Sigma} P\{wa\}\lambda(a, f(w)) \geq \sum_{w\in\Sigma^n} P(w) \inf_{f\in\Pi^n} \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w)),$$

hence it suffices to prove that that the opposite inequality holds.

For each n-long string w, let f_w be the function which attains the infimum $\inf_{f\in\Pi^n} \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w))$.

Thus, the required expectation of infima can be written in terms of these functions as

$$\sum_{w\in\Sigma^n} P(w) \inf_{f\in\Pi^n} \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w)) \; = \; \sum_{w\in\Sigma^n} P(w) \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f_w(w)).$$

We can now define a function $f : \Sigma^n \to \Sigma$ as $f(w) = f_w(w), \qquad w \in \Sigma^n$. It is clear from the definition of the function that

$$\sum_{w\in\Sigma^n} P(w) \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f(w)) = \sum_{w\in\Sigma^n} P(w) \sum_{a\in\Sigma} P\{a|w\}\lambda(a, f_w(w)),$$

which implies the desired inequality. □

Let $s(P\{0|w\})$ be the strategy that gives optimal loss in $H_{1|n}$.

In the following proof, we will consider two-way infinite sequences. However, the same theorem holds for one-way sequences as well (see Chapter 13 of Billingsley [1]). We briefly mention the formal correspondence.

Let $(X, \mathcal{B}, \mu)$ be a measure space with T being a measure preserving transform, not necessarily invertible. It is possible to construct a measure preserving system $(\hat{X}, \hat{\mathcal{B}}, \hat{\mu}, \hat{T})$ such that $\hat{T}$ is an invertible transform given by $\hat{T}^{-1}(x_1, \cdots)= (x_2, \cdots)$. Since T is measure preserving, $\hat{T}$ is also measure preserving. $(\hat{X}, \hat{\mathcal{B}}, \hat{\mu}, \hat{T})$ is called *natural extension of* $(X, \mathcal{B}, \mu, T)$. It is ergodic iff the original system is ergodic. For unilateral alphabet system, its natural extension has same entropy. For details, see Fact 4.3.2 of Downarowicz [4].

Let us define the following functions on the space of two-way infinite sequences.

$$g_k(\omega) = \lambda(\omega_0, s(P\{0|\omega_{-k}^{-1}\})) \;\text{ and }\; g(\omega) = \lambda(\omega_0, s(P\{0|\omega_{-\infty}^{-1}\})).$$

We now define the notion of strongly regular games, imposing two technical restrictions. We justify these restrictions by examining their consequences, and use these to prove the Shannon-McMillan-Breiman theorem.

Definition 14. *A regular game is* strongly regular *if*

1. *s is a continuous function of the conditional probability.*
2. *For each natural number N, define $G_N : \Sigma^\infty \to [0,\infty]$ by $G_N(\omega) = \sup_{k \geq N} |g_k(\omega) - g(\omega)|$. We require that $\{G_N\}_{N=1}^\infty$ be a uniformly integrable sequence.*

Chernov et al. [3] show that for a loss function to obey condition (1), it is sufficient for it to be a *proper loss function.* [5] First, we explain a consequence of condition (1). For a stationary ergodic distribution P, $P\{0 \mid \omega_{-k}^{-1}\} \to P\{0 \mid \omega_{-\infty}^{-1}\}$ as $k \to \infty$, and since g_k is a continuous function of the conditional distribution by condition (1), we have that $g_k \to g$ as $k \to \infty$, P-almost everywhere. (see Theorem 11.2 of Billingsley [1])

The requirement (2) is technical, but is necessary to handle the integrability of a large class of loss functions. We note that if the loss function is bounded as in square-loss and absolute-loss games, then it satisfies condition 2 trivially (we show this in Example 16), however it can also handle certain unbounded loss functions including log-loss.

Example 15. Log-loss Game. The loss function $\lambda : \{0,1\} \times [0,1] \to [0,\infty]$ is defined by $\lambda(b,\gamma) = -\log(|(1-b)-\gamma|)$. The optimal strategy is given by $P\{0 \mid \omega_{-k}^{-1}\}$, which is a continuous function of the conditional probability.

We have that for any N,

$$\int \sup_{k \geq N} |g_k(\omega) - g(\omega)| dP \leq \int \sup_{n \geq 1} |g_n(\omega) - g(\omega)| dP \quad \leq \int \sup_{n \geq 1} |g_n(\omega)| + \int g dP.$$

Hence to show that the sequence $\sup_{k \geq N} |g_k(\omega) - g(\omega)|$ is uniformly integrable, it suffices to show that $\int \sup_{n \geq 1} |g_n(\omega)| dP$ is integrable. It is easy to show that for a stationary distribution P and any $r \in \mathbb{R}$, $P\{\omega \mid \sup_k |g_k(\omega)| \geq r\} \leq 2e^{-r}$, from which the integrability of $\sup_k g_k$ follows. Thus $\sup_{k \geq N} |g_k - g|$, for $N = 1, 2, \ldots$ forms a uniformly integrable sequence of functions, hence the log-loss game is strongly regular. □

Example 16. Square-loss game. The loss function in the square loss game $\lambda : \{0,1\} \times [0,1] \to [0,1]$ defined by $\lambda(b,\gamma) = (b-\gamma)^2$. The optimal strategy in the square-loss game is to pick $\gamma = P\{1|\omega_{-k}^{-1}\}$, which is continuous in the conditional probability.

This loss function is bounded, hence $\int \sup_{k \geq 1} |g_n(\omega) - g(\omega)| \, dP \leq \int 1 dP = 1$, ensuring that $G_N = \sup_{k \geq N} |g_k(\omega) - g(\omega)|$ is uniformly integrable. Hence the square-loss game is strongly regular. □

We now elicit some consequences of our assumption of uniform integrability. For uniformly integrable sequences of functions, their limit function is integrable even in the absence of any dominating function. This is known as the *Vitali Convergence Theorem* (see, for example, Folland [7]).

[5] A loss function $\lambda : P(\Sigma) \times \Sigma \to [0,\infty]$, where $P(\Sigma)$ is the space of probabilities on Σ, is called *proper* if for any two $\pi, \pi' \in P(\Sigma)$, $E_\pi \lambda(\pi, \cdot) \leq E_\pi \lambda(\pi', \cdot)$.

Vitali Convergence Theorem. Let $(\Omega, \mathcal{F}, P)$ be a probability space. If $\{f_n\}_{n=1}^{\infty}$ is a sequence of uniformly integrable functions such that $f_n \to f$ P-almost everywhere, then f is integrable and $\lim_{n\to\infty} \int |f_n - f| dP = 0$.

Vitali Convergence of $\{G_N\}_{N=1}^{\infty}$ will be required in the final part of the proof of Theorem 18. We first show that uniform integrability of $\{G_N\}_{N=1}^{\infty}$ yields the integrability of the optimal loss. This is crucial in the Theorem that follows, and yields us a dominating function for the integrability in Theorem 18.

Lemma 17. *For a strongly regular game and a stationary distribution P,*

$$\lim_{n\to\infty} \int g_n \, dP = \int \lim_{n\to\infty} g_n \, dP = \int g \, dP. \tag{3}$$

Proof. We know that for each $n \in \mathbb{N}$, $\int |g_n| \, dP = \int g_n dP = H_{1|n}$, which exists for regular games and stationary distributions. Now, for every n, $\int |g_n| \, dP = \int |g - g_n - g| \, dP \geq \int |g| dP - \int |g - g_n| dP$. Hence we have

$$H = \lim_{n\to\infty} \int |g_n| \, dP \geq \int |g| dP - \liminf_{n\to\infty} \int |g - g_n| dP. \tag{4}$$

By the uniform integrability of $\{G_N\}_{N=1}^{\infty}$, we have that $\lim_{n\to\infty} \int |g - g_n| dP = 0$. Thus, by (4), we have $H \geq \int |g| dP$, ensuring that $\int g$ exists. Thus the interchange of the limit and the integral in (3) is justified by the Lebesgue dominated convergence theorem [7]. □

Using uniform integrability and the notion of continuity, we can introduce the setting for our Shannon-McMillan-Breiman Theorem.

Theorem 18. *For a strongly regular game $(\Sigma, \Gamma, \lambda)$, and stationary ergodic distribution $(\Sigma^{\infty}, \mathcal{F}, P)$, let H be the generalized entropy of the game. Moreover, let $\wp$ be a strategy such that for every n, $\wp^n$ achieves H_n. Then*

$$\lim_{n\to\infty} \frac{Loss(\omega_0^{n-1}, \wp^n)}{n} = H \tag{5}$$

for P-almost every $\omega \in \Sigma^{\infty}$.

We cannot use Birkhoff's ergodic theorem (see for example, Billingsley [1]) directly to prove the above theorem, since the summands in the Birkhoff average on the left of (5) depend in general on n, and are not the same integrable function. We however can use the convergence in conditional distributions ensured by a stationary distribution, in conjunction with Birkhoff's ergodic theorem to establish our result.

Proof. Recall that $g_k \to g$ almost everywhere, and $\int g$ exists by Lemma 17. We know $\text{Loss}(\omega_0^{n-1}, \wp^n) = \sum_{k=0}^{n} g_k(\omega)$.

Since T is measure preserving transformation, by change of variable, $\int g_k(\omega) dP = \int g_k(T^k \omega) dP = H_{1|k}$. Thus $\int g dP = \lim_{n\to\infty} \int g_n dP = \lim_{n\to\infty} H_{1|n} = H$.

By the Ergodic theorem, we get $\lim_{n\to\infty} \frac{1}{n}\sum_{k=0}^{n-1} g(T^k w) = \int g(w)dP = H$, for P-almost every $\omega \in \Omega$.

Now, $\frac{1}{n}\sum_{k=0}^{n-1} g_k(T^k w) = \frac{1}{n}\sum_{k=0}^{n-1} g(T^k w) + \frac{1}{n}\sum_{k=0}^{n-1}(g_k(T^k w) - g(T^k w))$, where the first term tends to H as $n \to \infty$. If we show second term in the previous equation tends to 0 a.e. as $n \to \infty$, we are done.

Define $G_N(w) = \sup_{k\geq N} |g_k(w) - g(w)|$. By the assumption of strong regularity, the sequence of functions $\{G_N\}_{N=1}^{\infty}$ is uniformly integrable. Also, since $g_n \to g$ P-a.e., we know that $G_N \to 0$ P-almost everywhere as $N \to \infty$. By the Vitali Convergence Theorem, $\lim_{N\to\infty} \int G_N\, dP = \int \lim_{N\to\infty} G_N\, dP = 0$.

Now for each N,

$$\begin{aligned}\limsup_{n\to\infty}\left|\frac{1}{n}\sum_{k=0}^{n-1}(g_k(T^k\omega) - g(T^k\omega)\right| &\leq \limsup_{n\to\infty}\frac{1}{n}\sum_{k=0}^{n-1}|g_k(T^k\omega) - g(T^k\omega)| \\ &\leq \limsup_{n\to\infty}\frac{1}{n}\sum_{k=0}^{n-1}G_N(T^k\omega) = \int G_N(\omega)dP\end{aligned}$$

where the last equality follows from Birkhoff Ergodic Theorem. Note that this holds for all values of N and right side converges to 0 a.e. as $N \to \infty$. Since the left side is non-negative, it is 0 a.e. So, $\frac{1}{n}\sum_{k=0}^{n-1}(g_k(T^k\omega) - g(T^k\omega)) \to 0$ as $n \to \infty$. This concludes the proof. □

Recall that the generalized entropy of the log-loss game is the Shannon entropy. We have shown previously that the square loss and the log-loss games are strongly regular, thus establishing that we have a proper generalization of the classical Shannon-McMillan-Breiman theorem.

6 Predictive Complexity - Main Theorem

We now consider computable prediction strategies. We would like to define the inherent unpredictability of a string x as the performance of an optimal computable predictor on x. It is not clear that one such predictor exists for any game. The work of Vovk and Watkins [17] establishes a sufficient condition for predictive complexity to exist.

Definition 19. *A pair of points* $(s_0, s_1) \in (-\infty, \infty]^2$ *is called a* superscore[6] *if there is a prediction* $\gamma \in \Gamma$ *such that* $\lambda(0,\gamma) \leq s_0$ *and* $\lambda(1,\gamma) \leq s_1$*. We denote the set of superscores for a regular game* $\mathcal{G}$ *by* $\mathcal{S}$*.*

Definition 20. *A prediction strategy* $\wp : \Sigma^* \to (-\infty, \infty]$ *is called a* superloss process *if the following conditions hold.*

1. $\wp(\Lambda) = 0$*, where* Λ *is the empty string.*
2. *For every string* x*, the pair* $(\wp(x0) - \wp(x), \wp(x1) - \wp(x))$ *is a superscore with respect to the game.*

[6] In Kalnishkan et. al. [11], [10], the concept is called a superprediction.

3. $\wp$ is upper semicomputable.

A superloss process K is *universal* if for any superloss process $\wp$ there is a constant C such that for every string x, $K(x) \leq \wp(x) + C$. It follows that the difference in loss between any two universal superloss processes is bounded by a constant. Hence we may pick a particular universal superloss process $\mathcal{K}$ and call $\mathcal{K}(x)$ the *predictive complexity* of the string x with respect to the game $\mathcal{G}$.

When we consider regular games, it is not necessary that an optimal strategy exists on Σ^* which incurs at most an additive loss when compared to any other prediction process. However, Vovk [16] and Vovk and Watkins [17] introduced the concept of *mixability* to ensure that one such universal process exists.

Definition 21. *Let $\beta \in (0,1)$. Consider the homeomorphism $h_\beta : (-\infty, \infty]^2 \to [0,\infty)^2$ specified by $h_\beta(x,y) = (\beta^x, \beta^y)$. A regular game $\mathcal{G}$ with set of superscores $\mathcal{S}$ is called β-*mixable *if the set $h_\beta(\mathcal{S})$ is convex. A game $\mathcal{G}$ is called* mixable *if it is β-mixable for some $\beta \in (0,1)$.*

We call a strategy ζ *computable* if there is a program M such that for any $w \in \Sigma^*$ and any $m \in \mathbb{N}$, $M(w,m)$ outputs a rational $r \in \Gamma$ and $|\lambda(w,r) - \lambda(w,\zeta(w))| < \frac{1}{2^m}$. Similarly, a loss function λ is computable if there is a program L such that for every $m \in \mathbb{N}$, $a \in \Sigma$ and $r \in \Gamma \cap \mathbb{Q}$, $L(a,r,m)$ outputs a rational r such that $|L(a,r,m) - \lambda(a,r)| \leq 2^{-m}$. We assume that λ has a computable modulus of continuity – that is, there is a function $h_\lambda : \mathbb{N} \to \mathbb{N}$ such that $h(n) = m$ implies that if $|x-y| < 2^{-m}$, then $|\lambda(b,x) - \lambda(b,y)| \leq 2^{-n}$. If the loss function is computable and proper, with a computable modulus of continuity, we call $(\Sigma^\infty, \Gamma, \lambda)$ a *computable* game.

Theorem 22. *[17] If a game $\mathcal{G}$ with set of superscores $\mathcal{S}$ is mixable, then $\mathcal{G}$ has a predictive complexity.*

It is known that the logloss and the square loss games are mixable. The coincidence of logloss and Kolmogorov complexity [13] enables us to view predictive complexity as a generalization of Kolmogorov complexity. Absolute loss game is known not to be mixable [19].

We mention a loss bound which holds for mixable games. This is used in the proof of the theorem which follows.

Lemma 23. *[11] If $\mathcal{K}$ is predictive complexity of a mixable game $\mathcal{G}$, then there is a positive constant c such that $|\mathcal{K}(xb) - \mathcal{K}(x)| \leq c \ln n$ for all $n = 1, 2, \cdots$, strings $x \in \Sigma^n$ and bits b.*

We can now show that for a strongly regular mixable game $\mathcal{G}$, the predictive complexity rate on an infinite sequence of outcomes attains the generalized entropy of a computable stationary ergodic distribution P, almost everywhere.

Theorem 24. *Let $\mathcal{G} = (\Omega, \Gamma, \lambda)$ be a strongly regular computable mixable game with predictive complexity $\mathcal{K}$. Let $(\Omega, \mathcal{F}, P)$ be the probability space over the outcomes where P is a stationary ergodic distribution with generalized entropy H. Then*

$$\lim_{n\to\infty} \frac{\mathcal{K}(\omega_0^{n-1})}{n} = H, \tag{6}$$

for P-almost every $\omega \in \Omega$.

Proof. (A) Upper bound: Let $N \in \mathbb{N}$ be arbitrary, $\epsilon = \frac{1}{2^N}$, and let $\delta = \frac{1}{2^m}$ be the modulus of continuity for λ at error ϵ. Let $k \in \mathbb{N}$. For every string w of length $n > k$, we consider the "k-window" predictor $s_k^N(w)$ which outputs a dyadic rational number $\frac{r}{2^{m+1}}$, $0 \leq r \leq 2^{m+1}$, in Γ depending on w_{n-k}^{n-1}. Since the number of such dyadic rationals is finite, and s_k^N depends only on k bits, there are only finitely many such predictors. Let this number be M.

Let $p_r^N = \frac{1}{M}$, $0 \leq r \leq M-1$. These weights sum to 1, and the family s_r^N is finite, hence we can use the aggregating algorithm [17] to produce an algorithm A^k such that for each $0 \leq r \leq M-1$, there is a constant c_r such that for any string $w \in \Sigma^*$, $\text{Loss}(w, A^k(w)) \leq \text{Loss}(w, s_k^{\mathbb{N}}(w)) + c_r$. Let c be the maximum of the finite number of constants c_r. Then for every $w \in \Sigma^n$ and s_k^N, $\text{Loss}(w, A^k(w)) \leq \text{Loss}(w, s_k^N(w)) + c$. Let $\wp_k$ be the predictor that attains the optimal value $H_{1|k}$. By the choice of δ, we know that one of the predictors $s_r^N(w)$ incurs at most ϵ more error than $\wp_k$ on each bit of w. Thus, we have that

$$\text{Loss}(w, A^k(w)) \leq \text{Loss}(w, \wp_N(w)) + c + \epsilon N,$$

for every w.

The loss rate incurred by $\wp_k$ on N-long prefixes is

$$\frac{1}{N-k} \sum_{i=0}^{N-k-1} \lambda(\omega_{k+i}, \wp_k(\omega_i^{k+i-1})) < H_{1|k} + \epsilon,$$

by the Ergodic Theorem[7], for large enough N, and almost every ω. For large enough k, this quantity is within $H + \epsilon$ by Theorem 10. Thus, we have that for all large enough k, there is a constant c_k such that for almost every ω,

$$\mathcal{K}(\omega_0^{N-1}) \leq \text{Loss}(\omega_0^{N-1}, A^k(\omega_0^{N-1})) + c_k \leq NH + 3N\epsilon + O(1).$$

(B) We now establish the reverse inequality, $\lim_{n\to\infty} \frac{\mathcal{K}(\omega_0^{n-1})}{n} > H - \epsilon$ for $\epsilon > 0$. Since

$$(\mathcal{K}(\omega_0^{n-1} \cdot 0) - \mathcal{K}(\omega_0^{n-1}),\ \ \mathcal{K}(\omega_0^{n-1} \cdot 1) - \mathcal{K}(\omega_0^{n-1}))$$

is a superscore, we have $E(\eta_n | \omega_0^{n-1}) \geq H_{1|n}$ where $\eta_n = \mathcal{K}(\omega_0^n) - \mathcal{K}(\omega_0^{n-1})$.

Now we can apply the martingale strong law of large numbers, Theorem VII.5.4 of Shiryaev [15] and get

$$\begin{aligned} \frac{\mathcal{K}(\omega_0^{n-1})}{n} = \quad & \tfrac{1}{n} \textstyle\sum_{i=0}^{n-1} \eta_i \quad = \frac{1}{n} \sum_{i=0}^{n-1} E(\eta_i | \omega_0^{i-1}) + o(1) \\ & \geq \tfrac{1}{n} \textstyle\sum_{i=0}^{n-1} H_{1|n} + o(1) = H + o(1), \end{aligned}$$

where the last equality is obtained by Theorem 10. □

[7] Applied on $f(\omega) = \lambda(\omega_k, \wp_k(\omega_0^{k-1}))$ with the left-shift transformation.

Acknowledgments. The authors would like to thank John Hitchcock and Vladimir V'yugin for helpful discussions, and to four anonymous reviewers for their corrections, suggestions in improving the presentation, and for pointing out errors in the previous version of the proof of Theorem 24, part (A).

References

[1] Billingsley, P.: Ergodic Theory and Information. John Wiley & Sons (1965)

[2] Cesa-Bianchi, N., Lugosi, G.: Prediction, Learning and Games. Cambridge University Press (2006)

[3] Chernov, A., Kalnishkan, Y., Zhdanov, F., Vovk, V.: Supermartingales in prediction with expert advice. Theor. Comput. Sci. 411(29-30), 2647–2669 (2010)

[4] Downarowicz, T.: Entropy in Dynamical Systems. New Mathematical Monographs. Cambridge University Press (2011)

[5] Feder, M.: Gambling using a finite state machine. IEEE Transactions on Information Theory 37, 1459–1461 (1991)

[6] Feder, M., Merhav, N., Gutman, M.: Universal prediction of individual sequences. IEEE Transations on Information Theory 38, 1258–1270 (1992)

[7] Folland, G.B.: Real Analysis. Wiley (1999)

[8] Fortnow, L., Lutz, J.H.: Prediction and dimension. Journal of Computer and System Sciences 70, 570–589 (2005)

[9] Grünwald, P.D., Dawid, A.P.: Game theory, maximum entropy, minimum discrepancy and robust bayesian decision theory. Annals of Statistics 32(4), 1367–1433 (2004)

[10] Kalnishkan, Y., Vovk, V., Vyugin, M.V.: Generalised Entropy and Asymptotic Complexities of Languages. In: Bshouty, N.H., Gentile, C. (eds.) COLT. LNCS (LNAI), vol. 4539, pp. 293–307. Springer, Heidelberg (2007)

[11] Kalnishkan, Y., Vovk, V., Vyugin, M.V.: Loss functions, complexities, and the Legendre transformation. Theor. Comput. Sci. 313(2), 195–207 (2004)

[12] Khinchin, A.Y.: Mathematical Foundations of Information Theory. Dover Publications (1957)

[13] Li, M., Vitányi, P.M.B.: An Introduction to Kolmogorov Complexity and its Applications, 3rd edn. Springer, Berlin (2008)

[14] Merhav, N., Feder, M.: Universal prediction. IEEE Transactions on Information Theory 44(6), 2124–2147 (1998)

[15] Shiryaev, A.N.: Probability, 2nd edn. Graduate Texts in Mathematics, vol. 95. Springer (1995)

[16] Vovk, V.: A game of prediction with expert advice. Journal of Computer and System Sciences, 153–173 (1998)

[17] Vovk, V.G., Watkins, C.: Universal portfolio selection. In: COLT, pp. 12–23 (1998)

[18] Vyugin, M.V., V'yugin, V.V.: Predictive Complexity and Information. In: Kivinen, J., Sloan, R.H. (eds.) COLT 2002. LNCS (LNAI), vol. 2375, pp. 90–105. Springer, Heidelberg (2002)

[19] V'yugin, V.: Suboptimal measures of predictive complexity for absolute loss function. Information and Computation 175, 146–157 (2006)

Author Index