



# NORTHWESTERN UNIVERSITY

Computer Science Department

**Technical Report**  
**Number: NU-CS-2026-37**

July, 2026

## **Reliably Extracting Value from Unreliable Black-Box Models**

**Vaidehi Srinivas**

### **Abstract**

The big data revolution along with the success of black-box machine learning models have given us access to large volumes of information. These models are then used to inform all manner of algorithmic and decision-making tasks. However, black-box machine learning models can be unreliable in ways that are hard to predict, and we are far from having a full characterization of the failure modes of these methods. If the black-box model could be arbitrarily wrong, conventional worst-case analysis may suggest that the best an algorithm can do is to ignore it, as it could be as hurtful as it might be helpful. This is pessimistic, as model predictions are often useful and valuable, even if it is hard to predict failure. In many high-stakes applications it is unreasonable, if not irresponsible, to fully disregard model predictions.

An alternative is to evaluate an algorithm both on reliability, and on how well it uses the black-box model. This thesis develops new algorithmic methods that simultaneously utilize the model as well as possible when it is correct, while also remaining robust in the setting where the model is not useful, or even misleading. A key aspect of this thesis is simultaneously providing (1) pessimistic guarantees: that the method is reliable even when the black-box model is not, and (2) optimistic guarantees: that the method provably distills useful information if the black-box model provides it.

This work extends the areas of algorithms with predictions and conformal prediction, and develops new algorithmic techniques that make new connections to other areas of algorithms including robust statistics, and online algorithms.

**Theoretical Computer Science, Foundations of Machine Learning**

NORTHWESTERN UNIVERSITY

Reliably Extracting Value from Unreliable Black-Box Models

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL  
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Computer Science

By

Vaidehi Srinivas

EVANSTON, ILLINOIS

September 2026

## Abstract

The big data revolution along with the success of black-box machine learning models have given us access to large volumes of information. These models are then used to inform all manner of algorithmic and decision-making tasks. However, black-box machine learning models can be unreliable in ways that are hard to predict, and we are far from having a full characterization of the failure modes of these methods. If the black-box model could be arbitrarily wrong, conventional worst-case analysis may suggest that the best an algorithm can do is to ignore it, as it could be as hurtful as it might be helpful. This is pessimistic, as model predictions are often useful and valuable, even if it is hard to predict failure. In many high-stakes applications it is unreasonable, if not irresponsible, to fully disregard model predictions.

An alternative is to evaluate an algorithm both on reliability, and on how well it uses the black-box model. This thesis develops new algorithmic methods that simultaneously utilize the model as well as possible when it is correct, while also remaining robust in the setting where the model is not useful, or even misleading. A key aspect of this thesis is simultaneously providing

- (1) pessimistic guarantees: that the method is reliable even when the black-box model is not, and
- (2) optimistic guarantees: that the method provably distills useful information if the black-box model provides it.

This work extends the areas of algorithms with predictions and conformal prediction, and develops new algorithmic techniques that make new connections to other areas of algorithms including robust statistics, and online algorithms.

## Acknowledgements

First and foremost, thank you to my advisor, Aravindan Vijayaraghavan. I have been very fortunate to get his advice about everything from technical subjects, to writing, teaching, navigating collaborations, the job market, and more. One indicator of how helpful Aravindan's advice is the industry I have made of passing it on second-hand to friends and colleagues. I am especially grateful for his guidance in how to choose and frame research problems and directions. His advice has often helped me find the right path, and helped me learn how to do this myself.

One way in which I really look up to Aravindan is his kindness and generosity. He always does his best to look out for all of us, whether it is his own students, members of the theory group, or the broader community. I have (many times) called him, freaking out about sections of papers that were coming together dangerously close to their deadlines, and he has calmly and reassuringly talked me out of my panic, helped me figure out what I needed to do, and given me feedback, even for projects he was not directly involved in— and even when he was on leave with his newborn.

Over the years, Aravindan has been someone that I can talk to about anything. I am very lucky to have had an advisor who is a great teacher, mentor, and friend. I was recently at a conference, where I met someone who said, “Ah, you are Aravindan's student. You must be so lucky!” I could not agree more. I have every intention of continuing to bother him for advice for many years to come— you can't get rid of me that easily.

Thank you also to Hari Vijayaraghavan, for being a great friend and board game partner, and graciously allowing us to use his office.

Thank you to my thesis committee members: Avrim Blum, Chao Gao, Samir Khuller, and

Jason Hartline. I worked with Avrim as part of a summer exchange on a project that eventually became part of this thesis. I learned a lot from him about how to take the spark of an idea and build it into concrete theory. I have worked closely with Chao for a large part of graduate school, meeting weekly for almost two years. I've learned a lot from him about high-dimensional statistics, and more broadly how to be precise and organized about my thoughts when doing research. I am very lucky to have been mentored by Samir throughout graduate school. From the first day that I arrived at Northwestern, Samir has looked out for me, keeping me up to date on interesting opportunities, and encouraging me at every milestone. Especially in the early days, it was incredibly powerful that he believed in me, even when I did not believe in myself. I have not directly worked on a research project with Jason, but I have learned a lot from him about how to do research. The first course I took at Northwestern was Jason's mechanism design class, and ever since then I have been very lucky to be immersed in the thriving economics and computation community here at Northwestern. This exposure, and our day-to-day spirited discussions about problems, have really shaped my research taste, and the way I think about new directions.

Thank you to my collaborators on the work that became this dissertation: Avrim Blum, Shaddin Dughmi, Chao Gao, Quanquan Liu, Liren Shan, and Aravindan Vijayaraghavan. I have learned a lot from Shaddin about learning theory in our time working on this project, and conversation during his time visiting Northwestern. Quanquan was an early mentor for me, when she was a postdoc at Northwestern, and I was a junior graduate student. Learning about dynamic algorithms from her was like traveling through the jungle with an expert navigator. Starting from me knowing nothing, she showed me the patterns and ideas in a very technical area. I have been really lucky to learn from her, and to call her a friend. Liren has been a mainstay of my entire time in graduate school— first as my senior and officemate at Northwestern, and later as a collaborator when he moved to TTIC. I have bothered him

with homework, with research, and even with a paper on my qualifying exam reading list that just didn't make any sense, and Liren has always managed to make complicated concepts simple, and convince me that everything will be ok.

Thank you to Anupam Gupta, Christian Schulz, Klaus Sutner, and David Woodruff, who were my research mentors before I came to Northwestern. Many of the concepts and skills I learned in these early projects have been my foundation in doing research. Thank you also to Anil Ada, my undergraduate academic advisor, who encouraged me to pursue theory in the first place, and was a big influence in my decision to pursue graduate school.

I have been very lucky to collaborate with many other amazing people. Thank you to Dionysis Arvanitakis, Aditya Bhaskara, Antares Chen, Paul Chichura, Alex Drlica-Wagner, Eric Evert, Anxin (Bob) Guo, Rachel Hur, Alex Ji, Liam O'Carroll, Lorenzo Orecchia, Tanmay Sinha, Erasmo Tani, Ziyu (Neil) Xu, and Samson Zhou. I have learned a lot from all of them, and I've had a lot of fun doing it.

I could not imagine a better place to grow up academically than the Northwestern CS theory group. I would like to thank the faculty Edith Elkind, Julia Gaudio, Jason Hartline, Samir Khuller, Konstantin Makarychev, Sidhanth Mohanty, Dima Pasechnik, Miki Racz, and Aravindan Vijayaraghavan. They have created an environment of cross-pollination of ideas across areas, which has really broadened my perspective on what it means to do theory. Each of these people has also gone out of their way to look out for me and support me in my academic journey, and been extremely generous with their effort and time. I am very grateful to have received their mentorship and support, and I am lucky to have been a student in a group that feels like a big team that works together.

I have been extremely lucky to share my time in the Northwestern theory group with many other wonderful people including Dionysis Arvanitakis, Dmitrii Avdiukhin, Aditya Bhaskara,

Gruia Calinescu, Matthew Casey, Shuwen Chai, Sayak Chakrabarty, Vaggos Chatziafratis, Aidao Chen, Rajni Dabas, Sami Davies, Jinshuo Dong, Shaddin Dughmi, Abhratanu Dutta, Eric Evert, Charlie Guan, Anxin (Bob) Guo, Yu He, He Jia, Victor Jourdain, Sanchit Kalhan, Yiduo Ke, Pooja Kulkarni, Yingkai Li, Mozhengfu Liu, Quanquan Liu, Vasilis Livanos, Sheng Long, Michail Mamakos, Indira Munoz, Aniket Murhekar, Liam O’Carroll, Ilias Papanikolaou, Phawin Prongpaophan, Shravas Rao, Aravind Reddy Talla, Lev Reyzin, Emmi Rivkin, Xingchen Sha, Anant Shah, Liren Shan, Dravy Sharma, Tanmay Sinha, Pattara Sukprassert, Madhav Suresh, Alex Tang, Matthew vonAllmen, Chang Wang, Nathan White, Yifan Wu, Sheng Yang, Chenhao Zhang, Jifan Zhang, and Shirley Zhang. The theory group always felt like somewhere where there was something new and exciting going on, and I have learned a lot from being around all of these people.

Vaggos, Sami, Jinshuo, Eric, Quanquan, and Shravas were postdocs when I was a junior graduate student and basically nothing made sense to me. I learned a huge amount from pestering them with my incessant basic questions, and each of them was incredibly helpful and kind. As one example, in my first year of graduate school, when we were still hybrid from the pandemic, Shravas would set up watch parties for talks in the theory lab multiple times a week. We would sit together with our microphone muted, and I would ask him questions about every slide, which he would patiently answer—telling me when I was missing something important vs. when that detail might not be important to get the main idea, and helping me learn when it was a good idea to just unmute and ask the speaker.

Aravind, Liren, Pat, and Chenhao were senior students when I started. They very quickly took me under their wing, and taught me how to be a graduate student. They also set a model of what it means to be involved in the group, organizing events for the theory group, the department, and IDEAL. I was very lucky to be in such a warm, welcoming environment, and to have them as close friends.



Charlie, Yiduo, Sheng, Anant, Matthew vonAllmen, and Yifan formed what I think of as my cohort. We have been through a lot together, from starting in the pandemic, to the myriad challenges and uncertainties of graduate school, through the job market. Graduate school has rarely been easy, but it has been a lot better for going through it together.

After my second year of graduate school, it felt like an asteroid hit the theory group when most of the students and postdocs graduated or moved to new positions at the same time. I was convinced that I would never have friends again. I was very lucky to be proved completely wrong by the new blood, including Dionysis, Matthew Casey, Shuwen, Rajni, Bob, Pooja, Vasilis, Ilias, Phawin, Emmi, Tanmay, and Xingchen who became the life and soul of the theory group, and close friends. It is very hard to leave them behind and know how much fun they will be having without me.

I have been very lucky to have been part of Aravindan's group, which has come to feel like family. Abhratanu, Aidao, Aravind, Alex, and Sanchit<sup>1</sup> were my extremely welcoming seniors. After two years of being the baby of the group, I wasn't really sure why Aravindan *needed* new students. However, I have to admit that Dionysis, Bob, and Tanmay<sup>2</sup> were excellent additions. We have come to form a posse, and I have no doubt that everyone is jealous of our hive mind.

The theory group has become home in my time in graduate school. I will miss our spontaneous arguments about inconsequential things, and tangents in a 4pm conversation that go on until after midnight. I have met some of my closest friends in my time here, and I hope we continue to be close for years to come.

In addition to the theory group, I have been supported by many mentors and friends in the

---

<sup>1</sup>Thank you also to Sanchit for proving that it was possible to graduate as Aravindan's student, without having a name that begins with "A."

<sup>2</sup>Thank you also to Tanmay, an avid reader of thesis acknowledgements, for proofreading this section.

computer science department at Northwestern. Thank you to Jessica Hullman for collaborating on coorganizing a workshop. Thank you to Emma Alexander and Sara Sood for their mentorship and encouragement over the years. Thank you to Madeline Agaton, Bella Barrios, James Devine, Indira Munoz, Dru Redmond, Jensen Smith, Xiaolin Wang, and Katie Winters for their help managing every aspect of academic life. Thank you to Melissa Chen, Walker Demel, Donna Hooshmand, Hyeok Kim, and Mukund Kuthalam for their friendship. Thank you to the whole CSPAC team for everything they do to support the Ph.D. students, and also being a wonderful group of people to hang out with.

A very special aspect of doing my graduate studies in Chicago was being a part of IDEAL (Institute for Data, Econometrics, Algorithms, and Learning). I have gotten to meet many amazing people working in the foundations of machine learning, both in Chicago and elsewhere, through the workshops, reading groups, special courses, and through my time in the summer exchange program. It has shaped the way that I think about research, and had a very tangible impact on what I work on: all of [Part I](#) and at least half of [Part II](#) of this thesis were triggered by IDEAL collaborations.

As a part of this Chicago-area connection, I am also lucky to have made some very good friends. Thank you to Anmol Kabra, Gene Li, Naren Manoj, Tushant Mittal, Max Ovsiankin, Kumar Kshitij Patel, Kavya Ravichandran, Han Shao, and Shashank Srivastava for adopting me as one of their own. And thank you to Avrim for declaring me an official *friend of TTIC* in his capacity as chief academic officer.

I have been lucky to have other close friends and family that have been fellow-travelers on the Ph.D. journey. Thanks to Rhea Jain, Aditya Mahadevan, Akshay Narayan, Nandeeka Nayak, Kabir Peshawaria, Deepti Raghavan, Utthara Rameshbabu, Maya Sankar, Anita Shlesinger, and Ziyu (Neil) Xu, for their camaraderie and commiseration over the highs and

lows of graduate school.

Anyone who has spoken to me for longer than three minutes knows that I am extremely lucky to have a large close-knit extended family that has looked out for me and taken care of me at every turn. Unfortunately, if I list everyone who is important, it will double the length of the document, but please know that I am humbled by and grateful for your love and support.

Thank you to my mom, Sripriya Chari, for her support and encouragement, and being a voice of reason in my life. Thank you to my dad, Sampath Srinivas, for, though I hate to admit it, excellent advice. It is actually on his advice that I applied to Northwestern in the first place, which has come to be an extremely fruitful decision. Thank you to my brother, Vishnu Srinivas, who said of my decision to go to graduate school “Wait, they’re going to pay you? To sit in a room and be frustrated? But you’re already doing that for free!” Thank you to the “geeps,” Ananthachari Thiruvengkatachari and Indumalini Ananthachari, for their support and unwavering interest in what I have been studying, and for staying more on top of my deadlines than I do. Thank you to my other grandmother, Hemalatha Sampathkumaran, who is the O.G. statistician in the family, I like to think I am following in her footsteps.

Thank you very much to my uncle and aunt, Sri and Sonali Srinath, who have been my *in loco parentis* for the last five years. From picking me up when I first landed in Chicago, to picking me up again when my car broke down, I have relied on them through all of the challenges of graduate school, and life in general.

As a final thought, I have come to strongly believe that the sciences, including the mathematical sciences, are fundamentally human endeavors. We may use whatever tools we have at our disposal, but in the end, science is about people, coming to understand and make

sense of the world in which we live, together.

At the global scale, it has been very exciting to have a front row ticket to and play some small part in contributing to the breadth of human knowledge. But more importantly, at the local scale, this journey has been about learning from and learning with so many truly wonderful people. I am very grateful to have had this opportunity of a lifetime. Thank you, everyone, very much from the bottom of my heart.

## Technical Acknowledgements

[Chapter 2](#), [Chapter 3](#), [Chapter 4](#), and [Chapter 5](#) are based on joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan. [Chapter 7](#) is based on joint work with Shaddin Dughmi and Aravindan Vijayaraghavan. [Chapter 9](#) is based on joint work with Quanquan C. Liu. [Chapter 10](#) is based on joint work with Avrim Blum.

I thank my committee members Avrim Blum, Chao Gao, Jason Hartline, Samir Khuller, and Aravindan Vijayaraghavan for their valuable feedback on the technical portions and presentation of this dissertation.

My work in this thesis has been supported by the National Science Foundation (NSF) under grants CCF-1652491, CCF-1934931 and CCF-2154100, the NSF-funded Institute for Data, Econometrics, Algorithms and Learning (IDEAL) through the grants NSF ECCS-2216970 and ECCS-2216912, and the Northwestern University Presidential Fellowship. I also gratefully acknowledge the support of NSF-Simons AI-Institute for the Sky (SkAI) via grants NSF AST-2421845 and Simons Foundation MPS-AI-00010513, and the support of the NSF-Simons National Institute for Theory and Mathematics in Biology (NITMB) via grants NSF DMS-2235451 and Simons Foundation MP-TMPS-00005320.

# Contents

|          |                                                                                           |           |
|----------|-------------------------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                                                       | <b>23</b> |
| 1.1      | Volume Optimal Conformal Prediction . . . . .                                             | 26        |
| 1.2      | Solving Related Instances . . . . .                                                       | 27        |
| 1.3      | Conclusions . . . . .                                                                     | 30        |
| <b>I</b> | <b>Volume Optimality in Conformal Prediction</b>                                          | <b>31</b> |
| <b>2</b> | <b>Introduction to Volume Optimality in Conformal Prediction</b>                          | <b>32</b> |
| 2.1      | Goals of Conformal Prediction . . . . .                                                   | 32        |
| 2.2      | Introduction to Volume Optimality . . . . .                                               | 35        |
| 2.3      | Overview of Results . . . . .                                                             | 41        |
| 2.4      | The Score-Function Abstraction . . . . .                                                  | 45        |
| 2.5      | Meta-theorem for Volume Optimal Conformal Prediction . . . . .                            | 48        |
| 2.6      | Connections to Other Algorithmic and Statistical Problems . . . . .                       | 52        |
| 2.7      | Prior and Related Algorithmic Approaches . . . . .                                        | 54        |
| 2.8      | Uniform Convergence of Bounded VC Classes over Exchangeable Random<br>Variables . . . . . | 56        |

|                                                                      |           |
|----------------------------------------------------------------------|-----------|
| <i>CONTENTS</i>                                                      | 13        |
| <b>3 The Volume Optimality Framework</b>                             | <b>62</b> |
| 3.1 Introduction . . . . .                                           | 63        |
| 3.1.1 Results in Chapter 3 . . . . .                                 | 65        |
| 3.2 Unsupervised Setting . . . . .                                   | 67        |
| 3.2.1 Approximate Volume Optimality . . . . .                        | 67        |
| 3.2.2 Impossibility of Distribution-Free Volume Optimality . . . . . | 69        |
| 3.2.3 Distribution-Free Restricted Volume Optimality . . . . .       | 70        |
| 3.2.4 Conformalizing Dynamic Programming . . . . .                   | 74        |
| 3.3 Supervised Setting . . . . .                                     | 77        |
| 3.3.1 Problem Setting . . . . .                                      | 77        |
| 3.3.2 Distributional Conformal Prediction . . . . .                  | 79        |
| 3.3.3 DCP meets DP . . . . .                                         | 80        |
| 3.3.4 Theoretical Guarantees . . . . .                               | 82        |
| 3.4 Numerical Experiments . . . . .                                  | 83        |
| 3.4.1 Construction of Nested Systems . . . . .                       | 83        |
| 3.4.2 Comparison in Unsupervised Settings . . . . .                  | 84        |
| 3.4.3 Comparison in Supervised Settings . . . . .                    | 86        |
| 3.5 Chapter Acknowledgements . . . . .                               | 89        |
| <b>4 Competing with High-dimensional Balls</b>                       | <b>90</b> |
| 4.1 Introduction . . . . .                                           | 91        |
| 4.1.1 Results in this Chapter . . . . .                              | 91        |
| 4.1.2 Proper Learning: Separation Result and Algorithms . . . . .    | 93        |
| 4.1.3 Application to Conformal Prediction . . . . .                  | 96        |
| 4.1.4 Prior and Related Algorithmic Approaches . . . . .             | 97        |
| 4.2 Technical Overview . . . . .                                     | 99        |

|          |                                                                                                     |            |
|----------|-----------------------------------------------------------------------------------------------------|------------|
| 4.2.1    | Proper Learning for Non-Worst-Case Instances . . . . .                                              | 102        |
| 4.2.2    | Reducing Worst-Case Improper Learning to Non-Worst-Case Proper Learning . . . . .                   | 105        |
| 4.3      | Proper Learning of Euclidean Balls . . . . .                                                        | 108        |
| 4.3.1    | Completing the Proof of Theorem 4.3.1 . . . . .                                                     | 116        |
| 4.3.2    | Better Volume Approximation for Almost Isotropic Distributions . . . . .                            | 120        |
| 4.4      | Finding Confidence Sets ( $\Gamma = 1 + d^{-1/2+o(1)}$ )-Competitive with Euclidean Balls . . . . . | 123        |
| 4.5      | Greedy Algorithm and Unions of Sets . . . . .                                                       | 128        |
| 4.6      | Hardness of Proper Learning . . . . .                                                               | 135        |
| 4.6.1    | NP-hardness of Proper Learning . . . . .                                                            | 135        |
| 4.6.2    | Computational Intractability even with Slack in Coverage . . . . .                                  | 137        |
| 4.7      | Application to Conformal Prediction . . . . .                                                       | 140        |
| 4.8      | Chapter Acknowledgements . . . . .                                                                  | 141        |
| <b>5</b> | <b>Competing with High-dimensional Ellipsoids</b>                                                   | <b>142</b> |
| 5.1      | Introduction . . . . .                                                                              | 143        |
| 5.1.1    | Results in this chapter . . . . .                                                                   | 146        |
| 5.1.2    | Technical Overview . . . . .                                                                        | 151        |
| 5.1.3    | Related Work . . . . .                                                                              | 162        |
| 5.2      | Algorithm and Analysis . . . . .                                                                    | 165        |
| 5.2.1    | SDP and KKT condition . . . . .                                                                     | 167        |
| 5.2.2    | Volume Approximation . . . . .                                                                      | 172        |
| 5.2.3    | Algorithmic guarantee . . . . .                                                                     | 178        |
| 5.3      | Algorithm for Subspace Recovery . . . . .                                                           | 183        |
| 5.4      | Computational Intractability . . . . .                                                              | 189        |
| 5.5      | Application to Convex Confidence Sets and Conformal Prediction . . . . .                            | 199        |

|                                                                    |            |
|--------------------------------------------------------------------|------------|
| <i>CONTENTS</i>                                                    | 15         |
| 5.5.1 Convex confidence sets . . . . .                             | 199        |
| 5.5.2 Conformal Prediction . . . . .                               | 202        |
| 5.6 Chapter Acknowledgements . . . . .                             | 203        |
| <b>6 Online Conformal Prediction</b>                               | <b>204</b> |
| 6.1 Introduction . . . . .                                         | 204        |
| 6.1.1 Results . . . . .                                            | 207        |
| 6.1.2 Technical Overview . . . . .                                 | 213        |
| 6.1.3 Related Work . . . . .                                       | 220        |
| 6.2 Algorithmic Guarantees for Arbitrary Sequences . . . . .       | 223        |
| 6.3 Algorithmic Guarantees for Exchangeable Sequences . . . . .    | 229        |
| 6.4 Lower Bound for Arbitrary Order Sequences . . . . .            | 242        |
| 6.5 Lower Bound Against “Best of Both Worlds” Algorithms . . . . . | 249        |
| 6.6 Future Directions . . . . .                                    | 258        |
| 6.7 Chapter Acknowledgements . . . . .                             | 260        |
| <b>7 Statistical Limits of Learning Prediction Sets</b>            | <b>261</b> |
| 7.1 Volumetric Learnability . . . . .                              | 261        |
| 7.2 Convex Sets are Volumetrically Learnable . . . . .             | 264        |
| <b>II Solving Related Instances</b>                                | <b>273</b> |
| <b>8 Introduction to Solving Related Instances</b>                 | <b>274</b> |
| 8.1 Solving Related Instances of a Problem . . . . .               | 274        |
| 8.2 Preliminaries: Algorithms with Predictions . . . . .           | 276        |
| 8.3 Data-driven Algorithm Design . . . . .                         | 279        |
| 8.4 Overview of Results in Chapters 9 and 10 . . . . .             | 281        |



|           |                                                                                                                   |            |
|-----------|-------------------------------------------------------------------------------------------------------------------|------------|
| <b>9</b>  | <b>Dynamic Algorithms with Predictions</b>                                                                        | <b>285</b> |
| 9.1       | Introduction . . . . .                                                                                            | 285        |
| 9.2       | Technical Overview and Summary of Contributions . . . . .                                                         | 292        |
| 9.2.1     | The Predicted-Updates Dynamic Model . . . . .                                                                     | 292        |
| 9.2.2     | Framework for Predicted-Updates Dynamic Algorithms . . . . .                                                      | 295        |
| 9.2.3     | Applications . . . . .                                                                                            | 302        |
| 9.2.4     | Comparison with Concurrent, Independent Work . . . . .                                                            | 303        |
| 9.3       | Offline Divide-and-Conquer to Fully-Dynamic Transformation . . . . .                                              | 304        |
| 9.3.1     | Preliminaries . . . . .                                                                                           | 304        |
| 9.3.2     | Algorithm Description . . . . .                                                                                   | 306        |
| 9.3.3     | Preprocessing and Scheduling of Predictions . . . . .                                                             | 309        |
| 9.3.4     | Work of a Single Call to RETRIGGER . . . . .                                                                      | 312        |
| 9.3.5     | Total Work Over All Calls to RETRIGGER . . . . .                                                                  | 321        |
| 9.3.6     | Boosting to High Probability . . . . .                                                                            | 324        |
| 9.3.7     | Putting Everything Together: the Final Theorem . . . . .                                                          | 326        |
| 9.4       | Incremental to Fully-Dynamic Transformation . . . . .                                                             | 330        |
| 9.4.1     | Comparison to Independent Concurrent Work of Peng and Rubinstein<br>[2023], van den Brand et al. [2023] . . . . . | 336        |
| 9.5       | Decremental to Fully-Dynamic Transformation . . . . .                                                             | 339        |
| 9.6       | Applications: Offline, Incremental, and Decremental to Fully Dynamic . . . .                                      | 342        |
| 9.7       | Future Directions . . . . .                                                                                       | 348        |
| 9.8       | Chapter Acknowledgements . . . . .                                                                                | 350        |
| <b>10</b> | <b>Competitive Strategies to Use “Warm Start” Algorithms with Predictions</b>                                     | <b>354</b> |
| 10.1      | Introduction . . . . .                                                                                            | 354        |
| 10.1.1    | Related work . . . . .                                                                                            | 363        |

|                                                                                  |            |
|----------------------------------------------------------------------------------|------------|
| 10.2 Preliminaries . . . . .                                                     | 367        |
| 10.3 Competing Against $k$ Fixed Points Offline . . . . .                        | 369        |
| 10.4 Competing with a Hypothesis Class of $k$ -wise Partitions Offline . . . . . | 375        |
| 10.4.1 Approximation . . . . .                                                   | 378        |
| 10.4.2 Learning the Hypothesis Class . . . . .                                   | 379        |
| 10.4.3 Algorithmic Guarantee . . . . .                                           | 386        |
| 10.5 Competing with Trajectories Online . . . . .                                | 387        |
| 10.5.1 Offline Baselines . . . . .                                               | 389        |
| 10.5.2 Reduction to $k$ -server . . . . .                                        | 394        |
| 10.5.3 Algorithm with Improved Runtime . . . . .                                 | 396        |
| 10.6 Future Directions . . . . .                                                 | 408        |
| 10.7 Chapter Acknowledgements . . . . .                                          | 409        |
| <b>11 Conclusion and Outlook</b>                                                 | <b>410</b> |
| 11.1 Outlook for Volume-Optimal Conformal Prediction . . . . .                   | 411        |
| 11.2 Outlook for Solving Sequences of Instances . . . . .                        | 413        |
| 11.3 Future Directions for Reliable Machine Learning with Unreliable Black Boxes | 415        |
| <b>References</b>                                                                | <b>420</b> |
| <b>A Details for Dynamic Algorithms with Predictions</b>                         | <b>446</b> |
| A.1 Preprocessing Proofs . . . . .                                               | 446        |
| A.2 Best-of-All-Worlds Backstop Proofs . . . . .                                 | 453        |
| <b>B Details for Volume Optimality Framework for Conformal Prediction</b>        | <b>455</b> |
| B.1 Additional Numerical Experiments . . . . .                                   | 455        |
| B.1.1 Construction of Nested Systems . . . . .                                   | 455        |
| B.1.2 Unsupervised Setting . . . . .                                             | 456        |

|             |                                                                       |            |
|-------------|-----------------------------------------------------------------------|------------|
| B.1.3       | Supervised Setting . . . . .                                          | 466        |
| B.2         | KDE Optimality Implies DP Optimality . . . . .                        | 481        |
| B.3         | Additional Proofs . . . . .                                           | 484        |
| B.3.1       | Proof of Theorem 3.2.2 . . . . .                                      | 484        |
| B.3.2       | Proof of Theorem 3.2.6 . . . . .                                      | 486        |
| B.3.3       | Proof of Theorem 3.3.3 . . . . .                                      | 486        |
| B.3.4       | Proofs of Proposition 3.2.4, Lemma B.2.3, Lemma B.3.1 and Lemma B.3.2 | 490        |
| <b>C</b>    | <b>Details for High-dimensional Ellipsoids</b>                        | <b>496</b> |
| C.1         | Computational Hardness of Obtaining Non-trivial Volume Approximation  |            |
| Guarantees  | . . . . .                                                             | 496        |
| C.2         | Proofs of Claims in Section 5.4 . . . . .                             | 497        |
| C.3         | Algorithm Details . . . . .                                           | 498        |
| C.3.1       | Preliminaries . . . . .                                               | 498        |
| C.3.2       | Bounding Ball . . . . .                                               | 499        |
| C.3.3       | Iterative Outlier Removal via Dual . . . . .                          | 500        |
| C.3.4       | Dual Equivalence . . . . .                                            | 504        |
| <b>Vita</b> |                                                                       | <b>507</b> |

# List of Figures

|     |                                                                               |     |
|-----|-------------------------------------------------------------------------------|-----|
| 2.1 | Organization of chapters in Part I . . . . .                                  | 41  |
| 3.1 | Conformalized-DP Comparison on Mixtures of Gaussians . . . . .                | 85  |
| 3.2 | Conformalized-DP Comparison to DCP-QR* . . . . .                              | 86  |
| 3.3 | Conformalized-DP Comparison to HPD-Split . . . . .                            | 87  |
| 4.1 | Most points in a ball are as close to the mean as the center . . . . .        | 102 |
| 4.2 | Points with high variance in some directions . . . . .                        | 106 |
| 6.1 | “Doubling” Argument for Online Conformal Prediction . . . . .                 | 227 |
| 6.2 | Lower Bound for Arbitrary-Order Sequences . . . . .                           | 244 |
| 6.3 | Arbitrary-Order Lower Bound, Symmetric Version . . . . .                      | 248 |
| 6.4 | Lower Bound for i.i.d. Sequences . . . . .                                    | 251 |
| 6.5 | Lower Bound for i.i.d. Sequences, Symmetric Version . . . . .                 | 258 |
| 9.1 | Problems with Improved Predicted-Updates Runtimes . . . . .                   | 302 |
| 9.2 | Dividers for Random Partition Tree . . . . .                                  | 314 |
| 9.3 | Shortest Window Containing Segment . . . . .                                  | 315 |
| 9.4 | Permanent Elements . . . . .                                                  | 331 |
| B.1 | Conformalized-DP Comparison with Conformalized KDE, Prediction Sets . . . . . | 458 |

|      |                                                                                                                          |     |
|------|--------------------------------------------------------------------------------------------------------------------------|-----|
| B.2  | Conformalized-DP Comparison with Conformalized KDE, Prediction Set Size                                                  | 458 |
| B.3  | Conformalized-DP Comparison on Censored Gaussian, Prediction Sets . . .                                                  | 460 |
| B.4  | Conformalized-DP Comparison on Censored Gaussian, Prediction Set Size .                                                  | 460 |
| B.5  | Conformalized-DP on Multimodal Data . . . . .                                                                            | 461 |
| B.6  | Conformalized KDE on Multimodal Data . . . . .                                                                           | 461 |
| B.7  | Conformalized-DP and Conformalized KDE Set Sizes on Multimodal Data .                                                    | 462 |
| B.8  | Conformalized-DP on ReLU Transformed Gaussian . . . . .                                                                  | 463 |
| B.9  | Method comparison on ReLU-transformed Gaussian . . . . .                                                                 | 463 |
| B.10 | Volume of Prediction Set Against Sample Size . . . . .                                                                   | 464 |
| B.11 | Volume of Prediction Set Against Coverage Probability . . . . .                                                          | 465 |
| B.12 | Simulated data of Romano et al. [2019] . . . . .                                                                         | 468 |
| B.13 | Simulated data from Izbicki et al. [2020a] . . . . .                                                                     | 469 |
| B.14 | Comparison of DCP-QR* and ConformalizedDP on skewed data . . . . .                                                       | 475 |
| B.15 | Comparison of Supervised Conformal Prediction Methods on Simulated Data<br>from Romano et al. [2019], Part 1 . . . . .   | 476 |
| B.15 | Comparison of Supervised Conformal Prediction Methods on Simulated Data<br>from Romano et al. [2019], Part 2 . . . . .   | 477 |
| B.16 | Comparison of Supervised Conformal Prediction Methods on Simulated Data<br>from Izbicki et al. [2020a], Part 1 . . . . . | 478 |
| B.16 | Comparison of Supervised Conformal Prediction Methods on Simulated Data<br>from Izbicki et al. [2020a], Part 2 . . . . . | 479 |

# List of Tables

|     |                                                                                   |     |
|-----|-----------------------------------------------------------------------------------|-----|
| 3.1 | Comparison of Supervised Conformal Methods on Data from Romano et al.<br>[2019]   | 87  |
| 3.2 | Comparison of Supervised Conformal Methods on Data from Izbicki et al.<br>[2020a] | 88  |
| B.1 | Conformalized KDE on Enzyme Dataset                                               | 466 |
| B.2 | Conformalized DP on Enzyme Dataset                                                | 466 |
| B.3 | Comparison of Supervised Conformal Methods on Dataset AirFoil                     | 480 |
| B.4 | Comparison of Supervised Conformal Methods on Dataset WineQuality                 | 481 |

# List of Algorithms

|      |                                                                              |     |
|------|------------------------------------------------------------------------------|-----|
| 2.1  | Split Conformal Prediction . . . . .                                         | 46  |
| 3.1  | Finding Minimum-Volume Union of Intervals with Dynamic Programming . . . . . | 74  |
| 4.1  | Algorithm for Finding a Dense Ball . . . . .                                 | 116 |
| 4.2  | Finding a Dense Ball for Isotropic Distributions . . . . .                   | 121 |
| 4.3  | Finding a Dense Ellipsoid . . . . .                                          | 125 |
| 4.4  | Greedy Algorithms for Finding Union of Sets . . . . .                        | 129 |
| 5.1  | Finding Confidence Ellipsoid . . . . .                                       | 178 |
| 5.2  | Robust Subspace Recovery . . . . .                                           | 184 |
| 6.1  | Meta-algorithm for Online Conformal Prediction . . . . .                     | 209 |
| 9.1  | Fully Dynamic with Predictions from Offline (part 1) . . . . .               | 351 |
| 9.2  | Fully Dynamic with Predictions from Offline (part 2) . . . . .               | 352 |
| 9.3  | Backstop Meta-Algorithm for $N$ Algorithms . . . . .                         | 352 |
| 9.4  | Expectation to High-Probability Boosting for Dynamic Algorithms . . . . .    | 353 |
| 10.1 | Using $k$ Predictions in Parallel . . . . .                                  | 370 |
| 10.2 | “Predict Yesterday’s Solution” . . . . .                                     | 393 |
| 10.3 | Quadratic Decay . . . . .                                                    | 398 |

# Chapter 1

## Introduction

The big data revolution and with the explosive recent successes of large foundation models have given us access to large volumes of information. This information, in the form of *black-box models*, is used to inform all manner of algorithmic and decision-making tasks.

However, black-box models are often unreliable in ways that are hard to predict. They can exhibit systematic biases, which sometimes reflect biases in their training data [see the survey of [Bommasani et al., 2021](#), §5.1]. They are often *miscalibrated* (*over-* and *under-confident*), a phenomenon that could be a byproduct of the training method [see the survey of [Geng, Cai, Wang, Koepl, Nakov, and Gurevych, 2024](#)]. Unexpected issues also arise because capabilities of these models seem to be linked in ways that we do not fully understand. For example, post-training adjustments to the way a model addresses one task often result in mysterious side-effects in how the model addresses seemingly unrelated tasks [[Betley, Cocola, Feng, Chua, Arditi, Sztyber-Betley, and Evans, 2025](#)].

In recent years, we have been discovering new techniques for designing foundation models,



and the new capabilities they unlock at an extraordinary pace. However, we are also discovering new issues equally rapidly. Because these models are already being deployed in high-stakes settings, this has quickly cemented *reliable machine learning* or *AI safety* as a defining scientific problem of our time.

While exciting work is being done on understanding the forces that drive foundation models, we are still decades away from having a full scientific understanding of how, when, and why these models work and fail. Because our current understanding of foundation models is largely heuristic, the reliability and safety patches that are deployed to these models in practice are often also heuristic. That is, the reliability and safety measures do not come with rigorous guarantees. This results in them being vulnerable to security attacks and *jailbreaks*, both in academic settings [see survey of [Yi, Liu, Sun, Cong, He, Song, Xu, and Li, 2024](#)], and in real-world settings [see e.g. [Schaul and Tiku, 2026](#)]. Thus these reliability measures are themselves not very reliable.

This large gap between the theory and practice of machine learning seems to lead us to the following choice when performing a task with access to an unreliable black-box model:

- (a) *Optimism*. Blindly trust the black-box model to make decisions, and achieve high performance most of the time, while failing unpredictably some of the time.
- (b) *Pessimism*. Ignore the black-box model, and fall back on an alternative method with a *worst-case guarantee*, and achieve mediocre performance all of the time.

Both options are unsatisfying. In high stakes situations, it is not clear whether it is more responsible to prioritize the potential benefit of trusting the model, or the reliability of ignoring it.

Recent lines of work in algorithms and statistics have explored a way to bridge these options

and achieve the “best of both worlds:”

- (c) *Cautious optimism*. Without knowing the quality of the prediction in advance, perform as well as a hypothetical alternative that did know the quality of the prediction.

Designing such methods requires a new algorithmic viewpoint. Traditionally in algorithm design, we fix an input model (e.g. *worst case*, *average case*, etc.) and then we design methods to optimize performance for that model. In this new viewpoint, we must design algorithms that simultaneously do well under multiple input models (e.g. accurate black-box predictions versus misleading black-box predictions). This requires us to establish simultaneous guarantees of

- (1) *Reliability*. The method achieves reasonable worst-case guarantees even when the black-box model is misleading,
- (2) *Informativeness*. The method provably takes advantage of the black-box model when it is accurate.

In this thesis, we develop two natural and complementary approaches. The first is a *task-agnostic* approach, where the goal is to “wrap” a black-box model in a layer that provably ensures reliability for any downstream use of the model. We do this by using the model to generate reliable and informative *confidence sets*. The second approach we develop is *task-specific*. In settings where we are repeatedly solving related instances of the same algorithmic task, we would like to use a learned model that can identify patterns in the instances that allow us to solve them more efficiently. This approach allows us to take advantage of structure in natural data, without sacrificing the reliability of algorithms with guarantees.

The work in this thesis builds on lines of work such as *algorithms with predictions*, *data driven algorithms*, and *conformal prediction*. Each of these frameworks was designed to

address different aspects of reliability in machine learning. In this thesis we will leverage the strengths of these frameworks, and also extend them to account for aspects in which they are insufficient.

The methods we design for these problems make new connections between diverse areas, including *data structures*, *online algorithms*, *optimization*, *convex geometry*, *high-dimensional probability*, *statistics* and *learning theory*, to establish a new toolkit to reason about reliable machine learning with unreliable black-box models.

## 1.1 Volume Optimal Conformal Prediction

[Part I](#) of this thesis focuses on a task-agnostic way of “wrapping” a black-box model with a layer that ensures reliability.

Consider a *regression* problem, where we want to predict some *target*  $Y$  from some *features*  $X$ . To do this, we have access to a *black-box* regression model  $f(\cdot)$  that gives us a prediction  $\hat{Y} = f(X)$  for  $Y$ . We would like to post-process  $f(\cdot)$  so that it satisfies a provable notion of *reliability*.

A popular paradigm that has seen a wealth of work in the last few years is *conformal prediction*. In conformal prediction, our goal is to provide a *prediction set*  $C(X)$  for  $Y$ , such that for an input *miscalibration* parameter  $\alpha \in [0, 1]$ , we have

$$\mathbb{P}[Y \in C(X)] \geq 1 - \alpha. \tag{1.1}$$

A key property of conformal methods is that they require access to only a small amount of *calibration data* to work, and they are designed to be *distribution free*, so that they are widely applicable in practice.

The property (1.1) is called *coverage*, and gives a provable notion of reliability. In addition to providing coverage, we would like that our conformal prediction algorithm is *efficient*, i.e., it outputs prediction sets that are as small as possible.

Typically in the literature, conformal methods provide provable guarantees of coverage. Efficiency, however, is usually evaluated empirically. In particular, this often requires new bespoke conformal methods for every new setting of interest, which goes against the distribution-free spirit of the problem. A few works also provide provable guarantees of efficiency: that they output the smallest possible prediction set for  $Y$  given  $X$  and  $f(\cdot)$ . However, these works are limited in the kinds of data distributions that they can handle (see “prior work” in [Section 2.2](#) for discussion).

In this thesis, we design conformal methods that are the first, to the best of our knowledge, to provide valid prediction sets of *minimum volume* in a distribution-free setting. In [Chapter 3](#) (based on [\[Gao, Shan, Srinivas, and Vijayaraghavan, 2025b\]](#)), we formalize the *volume optimality framework*, and provide guarantees for the setting of one-dimensional  $Y$ . In [Chapters 4](#) and [5](#) (based on [\[Gao, Shan, Srinivas, and Vijayaraghavan, 2025a, 2026\]](#)) we design algorithms for the setting of high-dimensional  $Y$ . In [Chapter 6](#) (based on [\[Srinivas, 2026\]](#)), we consider volume optimal conformal prediction in the online setting. In [Chapter 7](#), we explore the statistical limits of learning prediction sets.

An in-depth overview of [Part I](#) can be found in [Chapter 2](#).

## 1.2 Solving Related Instances

[Part II](#) of this thesis focuses on solving a sequence of related instances of a specific algorithmic task. For example, perhaps we have a routing problem that must be solved every day with

different demands. If the instances are related, we could hope to *learn* information that allows us to solve them more efficiently than tackling them one at a time with a worst-case algorithm.

This is a classic idea that is used in practice in many settings, including problems in computer vision that require learning a *prior* [Lu, Lin, Wu, Luo, Zheng, Xiong, and Wang, 2023], optimizing computer systems based on workloads [Kraska, Beutel, Chi, Dean, and Polyzotis, 2018, Akgun, Aydin, Burford, McNeill, Arkhangelskiy, and Zadok, 2023], and data compression schemes [Liu, Han, Liao, Li, Peng, Du, and Chen, 2024], to list just a few. With the advent of powerful foundation models, a hope is that this approach will be applicable to an even broader set of problems. However, a risk with such an approach is that these learned models for different tasks may not always generalize well to new instances. This requires us to develop new methods that can take advantage of these learned models, while also ensuring reliability in the case that they fail.

We explore this in two natural settings. The first leverages learned information about the *instances* in the sequence. The second leverages learned information about the *solutions* in the sequence.

**Dynamic algorithms with predictions.** In Chapter 9, based on [Liu and Srinivas, 2024], we show how to leverage unreliable black-box predictions to speed up *dynamic algorithms*. In the dynamic model, we solve a sequence of instances that changes slowly from one day to the next. An important aspect is that, while the *instances* only change in small increments, the *solutions* could be very different from one day to the next. The goal of the algorithm is to minimize the *total* time it takes to solve the entire sequence of instances.

Our work is in the paradigm of *algorithms with predictions*. We equip our algorithm with

black-box predictions of future updates to the instances. Our guarantee is twofold. If the predictions are even moderately correct, we prove that our algorithm will take advantage of them to significantly speed up computations. We also show that, in the spirit of worst-case analysis, even if the predictions of the black-box model are misleading, our algorithm will never perform worse than the best algorithm that ignores the model completely. This allows us to rigorously argue that we are taking advantage of the learned model, without sacrificing the worst-case guarantee.

We design a framework that applies to several dynamic problems including *all-pairs shortest paths*, *k-connectivity*, *depth-first-search trees*, *maxflow/mincut*, *strongly connected components*, *sparsest cut*, and *submodular maximization* (see [Figure 9.1](#) for detailed list).

**Fast algorithms for sequences of related instances.** In [Chapter 10](#), based on [[Blum and Srinivas, 2025](#)], we show how to solve sequences of instances of a problem faster than solving each from scratch, by learning information about the solutions on the fly, when possible.

The model we consider is more general than the setting of dynamic algorithms, in that the instances do not have to change slowly over time. We provide new methods that are able to solve these sequences quickly, if the *solutions* of the instances are related. We quantify how related the solutions are by their clustering cost (*k-server cost* in the online setting). This allows us to rigorously argue that we are taking advantage of learnable information. This work builds on and extends the areas of *algorithms with predictions* and *data driven algorithms*.

The framework we design applies to a broad range of problems that admit *warm-start* algorithms, including *bipartite matching*, *maxflow* via Ford-Fulkerson, and other problems with

local search algorithm such as *convex optimization*.

An in-depth overview of [Part II](#) of the thesis can be found in [Chapter 8](#).

## 1.3 Conclusions

In this thesis, we design a new toolkit for reliable algorithms that extract value from black-box machine learning models. We establish that this setting requires new kinds of methods that simultaneously guarantee *reliability* in the worst case, and provably take advantage of the black-box model when it is accurate. Addressing these two objectives simultaneously requires new algorithmic and statistical techniques.

We explore two different natural approaches to ensuring reliability. The first approach is *task-agnostic*, and “wraps” a black-box model in a layer that ensures reliability, regardless of the downstream task. The guarantees we establish are *distributional*, and show that the method will perform well on average, for any data distribution. The second approach is *task-specific*, and designs algorithmic methods that take unreliable black-box predictions into account to efficiently provide reliable output. The guarantees we establish hold in the *worst-case*, for even “one shot” of the algorithm.

This thesis provides a broad toolkit for designing and reasoning about these methods, that spans and connects techniques from diverse areas, including *data structures*, *online algorithms*, *optimization*, *convex geometry*, *high-dimensional probability*, *statistics*, and *learning theory*.

## Part I

# Volume Optimality in Conformal Prediction



## Chapter 2

# Introduction to Volume Optimality in Conformal Prediction

## 2.1 Goals of Conformal Prediction

*Conformal prediction* is a family of methods that has become one of the most popular ways to ensure reliability and perform *uncertainty quantification* for black-box models in recent years [Gammerman, Vovk, and Vapnik, 1998, Vovk, Gammerman, and Shafer, 2005]. The goal of conformal prediction is to take an unreliable black-box predictor  $f(\cdot)$ , and post-process it to satisfy a guarantee of reliability.

**Problem 2.1.1** (Conformal prediction (marginal coverage)).

Let  $(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1}) \in \mathcal{X} \times \mathcal{Y}$  be exchangeable random variables. That is, the data points  $(X_i, Y_i)$  are in uniformly random order (formally [Definition 2.1.2](#)).

A conformal predictor that achieves marginal coverage takes

- a target miscoverage rate  $\alpha \in (0, 1]$
- a black-box model  $\hat{Y} = f(X)$  to predict target variable  $Y \in \mathcal{Y}$  from feature variable  $X \in \mathcal{X}$ ,
- calibration data  $(X_1, Y_1), \dots, (X_n, Y_n)$ , and
- the test feature  $X_{n+1}$ ,

and outputs a prediction set  $C = C((X_1, Y_1), \dots, (X_n, Y_n), X_{n+1}) \subseteq \mathcal{Y}$  such that

$$\mathbb{P}_{(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1})} [Y_{n+1} \in C] \geq 1 - \alpha. \quad (\text{MC})$$

In addition, we would like the conformal predictor to be efficient, in that it outputs prediction sets  $C$  that are as small as possible.

**Definition 2.1.2** (Exchangeability). A sequence  $(Z_1, \dots, Z_n)$  is exchangeable if

$$(Z_1, \dots, Z_T) \stackrel{d}{=} (Z_{\sigma(1)}, \dots, Z_{\sigma(T)}),$$

for all permutations  $\sigma : [T] \rightarrow [T]$ , where  $\stackrel{d}{=}$  means they are equal in distribution.

Conformal prediction is posited as a way to ensure the reliability of a model. The coverage guarantee (MC) ensures the accuracy of the prediction sets that are output by the conformal predictor. This means that, in some sense, the model can be taken at “face value.”

Achieving marginal coverage (MC) alone is trivial. For example, a completely uninformative predictor that predicts the entire solution space,

$$C = C((X_1, Y_1), \dots, (X_n, Y_n), X_{n+1}) = \mathcal{Y}$$

achieves (MC). It is the added goal of efficiency that makes conformal prediction challenging.

Typically, conformal predictors are designed to provably satisfy the coverage guarantee (MC), and have their efficiency empirically validated on data. The most commonly studied settings are where the target space  $\mathcal{Y}$  is a discrete label space or where  $\mathcal{Y} = \mathbb{R}^d$ . Our focus will be on the Euclidean setting, in one dimension and many dimensions. In this setting, the efficiency or informativeness of a conformal predictor is usually measured by the Euclidean volume or *Lebesgue measure* of its prediction sets.

**Score functions and efficiency.** Most prior work designs these conformal predictors via the score-function abstraction, which we overview in [Section 2.4](#). This abstraction allows a designer to pick an arbitrary score function. The split conformal framework is then able to prove the coverage guarantee (MC), without requiring any properties of the score function. This makes the framework extremely flexible and applicable. However, the sizes of the prediction sets depend very heavily on the choice of score function. This results in having to hand-design new ad hoc score functions for each new domain of interest, which come with no guarantee of optimality.

The overhead of designing these score functions, along with the inefficiency of these methods, is a major bottleneck in the practical uptake of conformal methods, especially in high-dimensional settings. Ideally, we would like to design adaptive methods that are able to automatically fit any data distribution, without an expert having to guess the shape of the data via a hand-designed score function in advance.

**Uncertainty quantification.** Conformal prediction is also often used to perform *uncertainty quantification*. For a black-box model  $f(\cdot)$ , a popular method is to design a conformal “wrapper” for  $f(\cdot)$ . Then, the size of the prediction set  $C$  produced by the conformalized

model on an input  $X$  is taken to be a measurement of the model  $f(\cdot)$ 's uncertainty on the input  $X$ .

For the above method to be sound, we must have a guarantee that the size of the prediction set  $C$  is indicative of the true underlying uncertainty of the black-box model  $f(\cdot)$ , rather than inefficiency of the conformal method.

## 2.2 Introduction to Volume Optimality

In this thesis, we spearhead a new approach for conformal prediction. We explicitly formulate *efficient* conformal prediction as an optimization problem. Armed with this new formulation, we can design methods that have provable guarantees of optimality. In addition, the statistical and computation tractability of this problem is an interesting direction of study in its own right, with connections to many fundamental problems in algorithms and statistics.

**Problem 2.2.1** ( $\mathcal{C}$ -Restricted Volume-Optimal Conformal Prediction). *Let  $\mathcal{C}$  be a hypothesis class of prediction sets over  $\mathcal{Y} = \mathbb{R}^d$ , and let  $P$  be an arbitrary distribution over  $\mathbb{R}^d$ .*

*For a volume-approximation factor  $\mu \geq 1$  and a coverage slack factor  $\varepsilon > 0$ , A conformal predictor satisfying  $\mathcal{C}$ -restricted volume optimality, takes*

- *a target miscoverage rate  $\alpha \in (0, 1]$ , and*
- *$n$  samples  $Y_1, \dots, Y_n$  that are exchangeable with an unseen test example  $Y_{n+1}$*

*and outputs a prediction set  $C = C(Y_1, \dots, Y_n) \subseteq \mathbb{R}^d$  such that  $C$  achieves coverage,*

$$P(C) \geq 1 - \alpha. \tag{cov}$$

and, with high probability,  $C$  is approximately volume-optimal,

$$\text{vol}(C) \leq \mu \cdot \text{OPT}_{\mathcal{C}}(P, 1 - \alpha + \varepsilon), \quad (\text{VO})$$

where

$$\text{OPT}_{\mathcal{C}}(P, 1 - \alpha) := \inf\{\text{vol}(C) : C \in \mathcal{C}, P(C) \geq 1 - \alpha\}.$$

To design such conformal methods, we will reduce to the problem of learning small-volume confidence sets. This is relevant in both the empirical and the population settings.

**Problem 2.2.2** (Learning small confidence sets, empirical). *Fix a hypothesis class  $\mathcal{C}$ , a target miscoverage rate  $\alpha \in [0, 1]$ , coverage slack factor  $\delta \geq 0$ , and approximation factor  $\mu \geq 1$ .*

*Let  $y_1, \dots, y_n \in \mathbb{R}^d$  be an arbitrary dataset of points. The goal is to find a set  $\hat{C} \in \mathcal{C}$  such that  $\hat{C}$  approximately achieves the target coverage,*

$$\frac{1}{n} \sum_{i=1}^T \mathbf{1}[y_i \in \hat{C}] \geq 1 - \alpha - \delta,$$

*and  $\hat{C}$  is approximately volume optimal,*

$$\text{vol}(\hat{C}) \leq \mu \cdot \text{vol}(C^*),$$

*where  $C^* \in \mathcal{C}$  is the minimum volume set that achieves coverage  $1 - \alpha$  over the dataset.*

**Problem 2.2.3** (Learning small confidence sets, population). *Fix a hypothesis class  $\mathcal{C}$ , a target miscoverage rate  $\alpha \in [0, 1]$ , coverage slack factor  $\delta \geq 0$ , and approximation factor  $\mu \geq 1$ .*

Let  $y_1, \dots, y_n \sim_{i.i.d.} P$  for an arbitrary distribution  $P$  over  $\mathbb{R}^d$ . The goal is to find a set  $\hat{C} \in \mathcal{C}$  such that  $\hat{C}$  approximately achieves the target coverage,

$$P(\hat{C}) \geq 1 - \alpha - \delta,$$

and  $\hat{C}$  is approximately volume optimal,

$$\text{vol}(\hat{C}) \leq \mu \cdot \text{vol}(C^*),$$

where  $C^* \in \mathcal{C}$  is the minimum volume set that achieves coverage  $1 - \alpha$  over the  $P$ .

*Remark 2.2.4.* In the case that  $\mathcal{C}$  has bounded VC-dimension, a guarantee for the empirical problem ([Problem 2.2.2](#)) essentially immediately implies a guarantee for the population problem ([Problem 2.2.3](#)) via uniform convergence. See the proof of [Theorem 2.5.1](#) for details.

We discuss a few important aspects of this problem formulation.

**Distribution-freeness.** An important aspect of [Problem 2.2.1](#) is that the data is an arbitrary exchangeable sequence. This ensures that the user does not need to know anything about the shape of the data distribution in order to get volume optimality guarantees. This is especially important in our setting, where the values  $Y_i$  are typically the errors of a black-box model on datapoints. To address the setting of black-box models, we would like to avoid distributional assumptions on the types of errors that are made.

**Hypothesis class  $\mathcal{C}$ .** Unfortunately, [Problem 2.2.1](#) is not statistically tractable when competing against fully arbitrary prediction sets for arbitrary distributions. Informally, if one only sees samples from  $P$ , it is impossible to know whether the distribution is discrete and admits a prediction set of volume 0, or if the distribution is continuous and requires a predic-

tion set of finite measure. The formal statement can be found in [Theorem 3.2.2](#) in [Chapter 3](#). Thus to get a meaningful guarantee, we must restrict the problem in some way.

One option is to make parametric assumptions about the data distribution. This is unsatisfactory, since a core principle of conformal prediction is to be distribution-free. A common and important application of conformal methods is in *uncertainty quantification*, where the size of the conformal set is taken to be a measure of the uncertainty in the distribution. For such a measure to be meaningful, we would like that size of the conformal set is truly a property of the distribution being spread, rather than misspecification of the parametric model.

We take an alternate approach, where we constrain the form of the prediction sets, rather than the distribution. Thus our guarantees give competitive guarantees against a specified hypothesis class  $\mathcal{C}$ , and avoid the issue of misspecification. This is similar to the setting of *distribution-free PAC learning*. Similarly to the PAC case, we see that complexity of the problem is now governed by the complexity of the hypothesis class of prediction sets, rather than the complexity of the distribution or the black-box model.

Restricting the hypothesis class also allows us to explicitly constrain shapes of the prediction sets to be *interpretable*, or otherwise convenient for downstream tasks. This avoids producing prediction sets that achieve coverage but are difficult to describe or use downstream.

**Unsupervised setting.** The conformal prediction problem ([Problem 2.1.1](#)) is typically stated in the *supervised* setting, where the goal is to predict some target  $y \in \mathcal{Y}$  from features  $x \in \mathcal{X}$ . In the simplified statement of volume-optimal conformal prediction above ([Problem 2.2.1](#)), we have chosen to focus on the *unsupervised* case, where there are only targets  $y \in \mathcal{Y}$ .

The unsupervised setting can be thought of as a special case of the supervised setting, where all of the datapoints have the same feature  $X$ , and the data captures the remaining uncertainty in  $Y$  given  $X$ . Thus any supervised algorithm must also handle this case. To the best of our knowledge, volume-optimality even in this simplified case has not been studied prior to our work.

**Lifting to supervised setting.** Moreover, our results in the unsupervised setting can translate to supervised methods. Most existing approaches for the supervised conformal prediction problem go through a reduction to the unsupervised problem. A simple example is to do unsupervised conformal prediction on the *residuals*  $S_i = \|f(X_i) - Y_i\|$ . Then, a prediction set for the  $S_i$  and the value of  $X_{n+1}$  naturally translate to a prediction set for  $Y_{n+1}$ .

More generally, the standard paradigm for supervised conformal prediction is the score-function abstraction, which uses a hand-designed score function to collapse the feature-label pair  $(X_i, Y_i) \in \mathcal{X} \times \mathcal{Y}$  to a one-dimensional score  $S_i \in \mathbb{R}$ . The conformal method can then be seen as an unsupervised method run on the scores  $S_i$ . We overview this strategy in [Section 2.4](#).

Another group of strategies, including *distributional conformal prediction* [[Chernozhukov, Wüthrich, and Zhu, 2021](#)] and *conformalized quantile regression* [[Romano, Patterson, and Candès, 2019](#)], use a black-box predictor for the distribution of  $Y|X$ , and apply unsupervised methods to these distributions. In [Section 3.3](#) in [Chapter 3](#) we will see that the methods that we design for the unsupervised setting can also be ported to the supervised setting in this way.

Overall, volume optimality in the unsupervised setting is a fundamental problem that needs



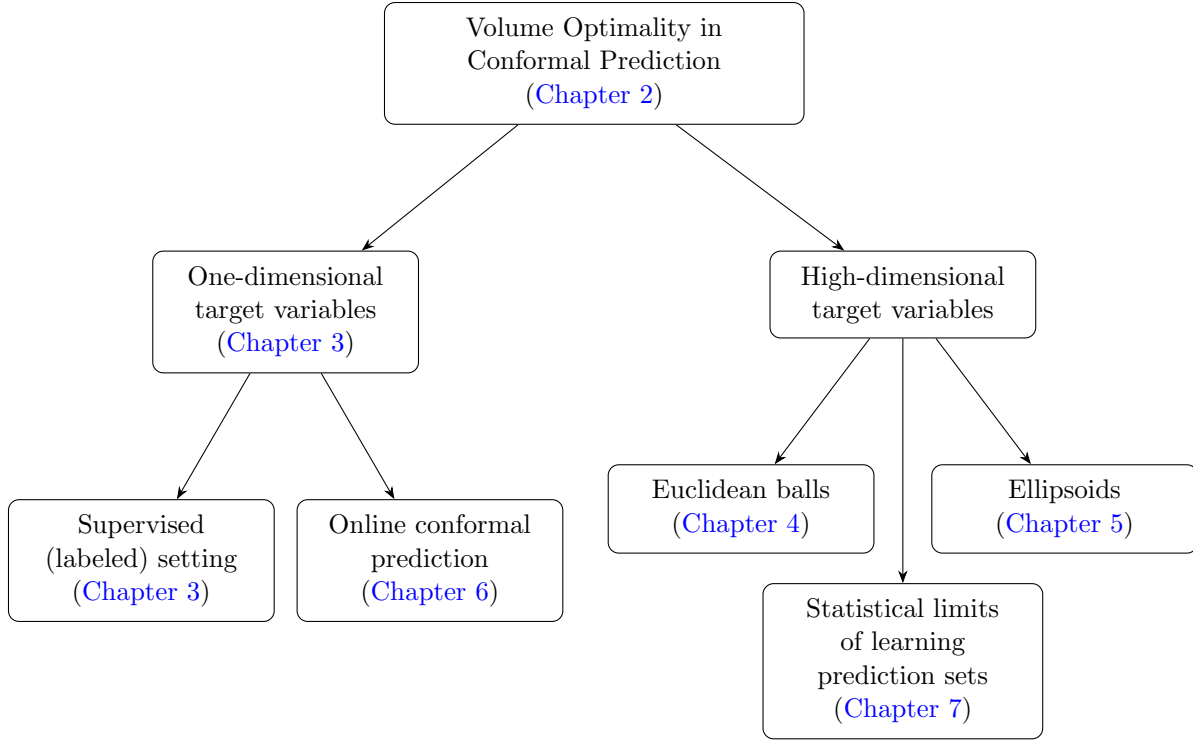
to be understood before considering the supervised setting. We will see examples of why this problem is interesting in its own right, and how it can apply to the supervised setting. We also note that there are many interesting questions left to explore related to the extension to the supervised setting.

**Previous work.** The volume of the prediction set in conformal prediction is also sometimes referred to as “efficiency” has been stated as an important consideration in many prior works [see e.g., [Shafer and Vovk, 2008](#), [Vovk, Fedorova, Nouretdinov, and Gammernan, 2016](#), [Angelopoulos and Bates, 2023](#)]. However, most works that we are aware of do not give theoretical guarantees of volume optimality, and mainly reason about volume control through empirical evaluations.<sup>1</sup>

Notable exceptions include the works of [Lei, Robins, and Wasserman \[2013\]](#) in the unsupervised setting, [Lei \[2014\]](#), [Vovk, Fedorova, Nouretdinov, and Gammernan \[2016\]](#), [Sadinle, Lei, and Wasserman \[2016\]](#) in the classification setting, [Lei, Rinaldo, and Wasserman \[2015\]](#) for functional data and recent works of [Izbicki, Shimizu, and Stern \[2020a,b\]](#) and [Kiyani, Pappas, and Hassani \[2024\]](#) in the regression setting. As summarized by [Angelopoulos, Barber, and Bates \[2025\]](#), a sufficient condition that leads to volume optimality of conformal prediction is consistent estimation of the conditional density function of  $Y$  given  $X$ . This is essentially the strategy adopted by previous work [[Lei, Robins, and Wasserman, 2013](#), [Izbicki, Shimizu, and Stern, 2020a,b](#)]. In comparison, our method, by incorporating a framework of [Chernozhukov, Wüthrich, and Zhu \[2021\]](#), builds on the estimation of the conditional CDF via a new conformity score computed by dynamic programming, and thus also works in settings where good conditional density estimation is impossible or density does not even

---

<sup>1</sup>Some works also guarantee that the coverage is not much more than  $1 - \alpha$ , e.g.,  $\mathbb{P}\left(Y_{n+1} \in \widehat{C}(X_{n+1}) \mid X_{n+1}\right) \leq 1 - \alpha + o(1)$  to argue that the prediction set is not too big. However, smallness according to the measure of the  $Y_i$  does not necessarily reflect a small volume (or Lebesgue measure) for the set.

Figure 2.1: Organization of chapters in [Part I](#)

exist.

To the best of our knowledge, all prior works that provide guarantees of volume optimality require stronger distributional assumptions than exchangeability or even i.i.d.-ness of the data. Our framework and results are the first to provide *distribution-free* volume optimality guarantees.

## 2.3 Overview of Results

We provide an informal chart of the results and how they are related in [Figure 2.1](#).

**Meta-theorem for volume optimal conformal prediction.** We begin in [Section 2.5](#) by showing that how to reduce the problem of volume optimal conformal prediction ([Prob-](#)

lem 2.2.1) to fitting a small prediction set on an arbitrary dataset (Problem 2.2.2). This uses the standard analysis of *split conformal prediction*, which we overview in Section 2.4, and uniform convergence of bounded VC-classes over exchangeable random variables (Section 2.8). Many of the later results will focus efficient algorithms for this subroutine for different hypothesis classes of prediction sets.

**Volume optimality framework.** In Chapter 3 we lay out the problem of volume-optimal conformal prediction. In Theorem 3.2.2, we show that it is impossible to achieve volume optimality without some restriction on either the distribution of points or the prediction sets that we compete with. We then define the notion of *distribution-free restricted volume-optimality* in Section 3.2.3, which corresponds to Problem 2.2.1, and we observe that this notion is statistically tractable for any hypothesis class  $\mathcal{C}$  of prediction sets that has bounded VC-dimension. As a proof of concept, we show that this problem is algorithmically tractable in the one-dimensional setting  $\mathcal{Y} = \mathbb{R}$ , where we compete with the concept class that is a union of  $k$  intervals. This is done via a simple dynamic programming procedure (Theorem 3.2.6). We show that our unsupervised notion of volume-optimality has some natural consequences for the supervised setting, when the algorithm is given a black-box predictor for the c.d.f. of the conditional distribution  $y \mid x$ , via the existing framework of *distributional conformal prediction* (Theorem 3.3.3). Finally, we provide experiments to validate that our new methods outperform the state-of-the-art for in many settings (Section 3.4).

**High-dimensional setting.** In Chapter 4 and Chapter 5 we design efficient algorithms for the high-dimensional setting. Specifically, we compete with the hypothesis class of high-dimensional balls and high-dimensional ellipsoids respectively. Since both of these classes have bounded VC-dimension, these problems are statistically tractable via a simple procedure that simply fits the smallest volume ball (resp. ellipsoid) that captures at least the target

fraction of the data. However, computing these minimizers is algorithmically challenging. For arbitrary data, this problem, both for balls and ellipsoids, is computationally intractable (Theorem 4.6.1, Theorem 5.4.2).

For high-dimensional balls, we give a volume approximation of  $\exp(\tilde{O}(\sqrt{d}))$  by outputting a confidence set in the shape of an ellipsoid (Theorem 4.1.1). We think of this as an analogue of *improper learning*, and it allows us to sidestep strong computational hardness of the corresponding *proper learning* problem of approximating the best ball with another ball. The design and analysis of this algorithm goes through a novel connection to *robust estimation*.

For high-dimensional ellipsoids, we see that any volume-approximation captures the hard problem of *robust subspace recovery*. Essentially, an algorithm that achieves a multiplicative volume-approximation against the best prediction ellipsoid must be able to tell when a large fraction of the dataset lies on a low-dimensional subspace (and therefore the dataset admits a confidence ellipsoid of volume 0). We sidestep this issue by designing an ellipsoid that competes with the minimum volume confidence ellipsoid with bounded *condition number* (or *aspect ratio*)  $\beta$ . This allows us to get volume approximations that have a mild dependence on the parameter  $\beta$ . In this setting, we are able to design bicriteria algorithms that trade off the volume approximation factor with an approximation factor in the additional miscoverage they incur (Theorem 5.2.7). The algorithm and analysis develop new insights into the rich primal-dual structure of the *minimum volume enclosing ellipsoid*.

As a consequence of our method to find confidence ellipsoids, we are also able to develop an algorithm for the robust subspace recovery problem for *worst-case* data in certain regimes. This is the first guarantee for this problem that makes no distributional assumptions, to the best of our knowledge.

**Online conformal prediction.** Conformal prediction guarantees are usually given under the assumption that the test data point  $Y_{n+1}$  and the calibration data  $Y_1, \dots, Y_n$  are *exchangeable*. This is because the goal of conformal prediction is to design methods that can be applicable to real data, which may not conform to strong distributional assumptions.

While exchangeability is a weaker notion than, say, requiring the data to be i.i.d., it still imposes strong restrictions on the kinds of data that we can design methods for. For example, exchangeability imposes that there can be no time-dependent distribution shift. To address this limitation and provide methods that are truly distribution free, some works have considered an *online* model for conformal prediction [Gibbs and Candes, 2021]. In this model, the algorithm sees a stream of *arbitrary* datapoints  $Y_1, \dots, Y_T$ . On each day  $t$ , the algorithm must output a prediction set  $C_t$  before  $Y_t$  is revealed. The goal of the algorithm is to achieve coverage *on average*: that is, on close to a  $(1 - \alpha)$  fraction of days, the point should fall within the prediction set  $Y_t \in C_t$ .

In [Chapter 6](#), we initiate the study of the online conformal prediction problem with respect to volume optimality. That is, we would like to achieve coverage on average, using prediction sets that compete in volume with the smallest-volume *fixed* prediction set that achieves the target coverage over the points, in hindsight. We show that this problem admits very different guarantees from the offline case of conformal prediction, and behaves quite differently from the related problem of *online learning*. We show that the algorithms developed in past work can be very inefficient with respect to volume, and design new methods that achieve optimal tradeoffs.

**Statistical limits of learning prediction sets.** In [Chapter 3](#) we make the observation that if a hypothesis class  $\mathcal{C}$  of prediction sets has bounded VC-dimension, then that is sufficient to ensure that we can learn  $(1 + \varepsilon)$ -approximately volume-optimal prediction sets

from  $\mathcal{C}$  statistically efficiently (where the sample complexity grows with the VC-dimension).

Surprisingly, while this condition is sufficient, it is not necessary. In [Chapter 7](#) we show that the family of convex sets, which has unbounded VC-dimension, is still learnable in the sense of learning volume-optimal prediction sets.

## 2.4 The Score-Function Abstraction

Here, we overview the *score-function abstraction*, one of the most common ways to design conformal prediction algorithms. It is not necessary to understand this family of methods to understand the results in this thesis. We provide this as a comparison point, to highlight the advantages of the new methods that we design. This also serves to introduce *split conformal prediction*, which we will use to design our volume optimal conformal prediction methods.

**Split conformal prediction.** The split conformal method ([Algorithm 2.1](#)) essentially splits the data into a *training set* and a *holdout set*. On the training set, it learns a black-box *score function*  $s : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ , which tries to capture the shape of the data. For a particular  $x \in \mathcal{X}$ , one can think of the score function as ranking the values in  $\mathcal{Y}$  by how much that  $(X, Y)$  pair “conforms” to the data distribution. The method will choose a prediction set that favors  $Y \in \mathcal{Y}$  with low scores.

The split conformal method estimates the one-dimensional score distribution on the hold-out data. For a particular feature  $X_{n+1}$ , it will predict the set of  $Y$  that would result in  $(X_{n+1}, Y_{n+1})$  having a score in the bottom  $(1 - \alpha)$ -fraction of scores.

An advantage of this abstraction is that it provides a guarantee of coverage using only the exchangeability of the  $(X_i, Y_i)$  and no properties of the learned score function at all. The proof of this is simple and elegant, but in the interest of concision, instead of repeating it here,

---

**Algorithm 2.1** Split Conformal Prediction

---

**Require:** Calibration data  $(X_1, Y_1), \dots, (X_n, Y_n) \in \mathcal{X} \times \mathcal{Y}$ , test feature  $X_{n+1} \in \mathcal{X}$ **Ensure:** Prediction set  $C$  contains unknown  $Y_{n+1}$  with probability  $\geq (1 - \alpha)$ 

- 1: Split data into *training set* and *holdout set* of size  $n/2$  each
- 2: On training set: learn a *score function*  $s : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$
- 3: On holdout set: compute  $n/2$  scores  $s(x_1, y_1), \dots, s(X_{n/2}, Y_{n/2})$  of each of the points  $(X_i, Y_i)$  in the holdout set
- 4: Take  $\tau$  to be the  $((1 - \alpha) + \varepsilon)$ th quantile of the scores  $s(X_i, Y_i)$
- 5: Predict

$$C := \{Y \in \mathcal{Y} : s(X_{n+1}, Y) \leq \tau\}$$


---

we refer the reader to Theorem 1.2 in this recent textbook on the theoretical foundations of conformal prediction [Angelopoulos, Barber, and Bates, 2025].

A disadvantage of this framework is that this makes it hard to reason about the *efficiency*, i.e., prediction set size, of these methods. It is straightforward to see that the score threshold  $\tau$  (Algorithm 2.1, line 4) is the tightest threshold that achieves coverage, once the score function is fixed. However, the prediction set  $C \subseteq \mathcal{Y}$  is found by inverting the score function into  $\mathcal{Y}$ , so the prediction set size depends very heavily on the choice of the score function.

The score-function abstraction also arises in other paradigms of conformal prediction. For example, in *full-conformal prediction* the goal is to be more sample-efficient by avoiding splitting the data into training and holdout samples. In online conformal prediction, methods like *adaptive conformal inference* [Gibbs and Candes, 2021] achieve coverage on average over time by predicting sets that are thresholds of the score function. For a more in depth overview of this abstraction and how it is used, we refer the reader to Angelopoulos, Barber, and Bates [2025]. While they use score functions in different ways, most modern conformal methods go through thresholding a black-box score-function that was designed by an expert for a specific application domain.

**Disadvantages of the score function abstraction.** The result of this is that choosing the score function is one of the main concerns in the conformal prediction literature, and a large body of work focuses on designing new score functions by hand, to allow for efficient prediction sets on their domain of interest.

On the theoretical side, while the score function abstraction makes arguing about coverage much easier, it is not clear that it is helpful abstraction when designing efficient prediction sets. This is because it encodes the problem of designing potentially complicated high-dimensional prediction sets into a one-dimensional score space. This can destroy the high-dimensional structure that we want to take advantage of.

One can see the results in this thesis as designing new conformal methods that avoid the score function abstraction, and directly argue about the  $\mathcal{Y}$  space, instead of the score space  $\mathbb{R}$ , to achieve volume optimality.

**Nested set viewpoint.** We can equivalently think of a score function as a nested family of sets. These sets correspond to the *sublevel sets* of the score function, with the inner sets corresponding to the low values of the score function and the outer sets corresponding to the high values of the score function. This view was advocated by [Gupta, Kuchibhotla, and Ramdas, 2022], and we expand on this in Chapter 3. This perspective is helpful for us in designing volume optimal conformal methods, as we can directly work with the sets that the method outputs. From this viewpoint, one can alternately think of our results as a new paradigm for learning score functions that ensures volume-optimality.



## 2.5 Meta-theorem for Volume Optimal Conformal Prediction

Using uniform convergence over classes of bounded VC-dimension exchangeable data, we can use any subroutine that can find approximately volume optimal prediction sets on some empirical dataset (Problem 2.2.2), to construct an approximately volume optimal conformal prediction algorithm (Problem 2.2.1). We do this in the model of *split conformal prediction*. Once we have this theorem, our results in Chapter 3, Chapter 4, and Chapter 5 focus on designing the volume-approximation subroutine for different hypothesis classes  $\mathcal{C}$ .

**Theorem 2.5.1** (Conformal prediction from volume approximation). *Let  $\mathcal{C}$  be a set family of bounded VC dimension over  $\mathbb{R}^d$ . Let  $\mathcal{A}_{\mathcal{C}}$  be a subroutine that takes a target miscoverage rate  $\alpha \in [0, 1]$ , and a set of points  $\mathbf{Y} = \{Y_1, \dots, Y_n\} \in \mathbb{R}^d$  and outputs a set  $\hat{C} \in \mathcal{C}$  such that  $\hat{C}$  approximately achieves the target coverage on the set,*

$$\frac{|\mathbf{Y} \cap \hat{C}|}{n} \geq 1 - \alpha - \delta_{\mathcal{A}},$$

*and  $\hat{C}$  is approximately volume-optimal on the set,*

$$\text{vol}(\hat{C}) \leq \mu_{\mathcal{A}} \min_{C \in \mathcal{C}: \frac{|\mathbf{Y} \cap C|}{n} \geq 1 - \alpha} \text{vol}(C),$$

*where the approximation parameters  $\delta_{\mathcal{A}} \geq 0$  and  $\mu_{\mathcal{A}} \geq 1$  are fixed attributes of  $\mathcal{A}_{\mathcal{C}}$ .*

*Let  $Y_1, \dots, Y_{2n}, Y_{2n+1}$  be exchangeable data. We design a conformal method, that given data  $Y_1, \dots, Y_{2n}$ , outputs a prediction set  $\hat{C} \in \mathcal{C}$ , such that*

- *regardless of the number of samples  $n$ ,  $\hat{C}$  achieves coverage on  $Y_{2n+1}$  on average*

(marginal coverage)

$$\mathbb{P} \left[ Y_{2n+1} \in \widehat{C} \right] \geq 1 - \alpha,$$

- With high probability  $1 - \text{poly}(n)$ ,  $\widehat{C}$  ensures volume optimality

$$\text{vol}(\widehat{C}) \leq \mu_{\mathcal{A}} \text{vol}(C_{(1-\alpha+\frac{1}{n}+\delta_{\mathcal{A}}+\delta_{\text{UC}})}^*),$$

where  $C_{(1-\alpha+\frac{1}{n}+\delta_{\mathcal{A}}+\delta_{\text{UC}})}^*$  is the smallest volume set that achieves coverage  $1 - \alpha + \frac{1}{n} + \delta_{\mathcal{A}} + \delta_{\text{UC}}$  over the full set of data  $Y_1, \dots, Y_{2n+1}$ , and  $\delta_{\text{UC}} = \widetilde{O}(\sqrt{\text{VC}(\mathcal{C})/n} + \sqrt{\log n/n})$ , where  $\widetilde{O}(\cdot)$  hides polylogarithmic factors in the VC-dimension of  $\mathcal{C}$ ,  $\text{VC}(\mathcal{C})$ .

*Remark 2.5.2* (Improper learning). The above guarantee also holds for *improper learners*. That is, consider set families  $\mathcal{C}_{\text{compete}} \subseteq \mathcal{C}_{\text{learn}}$ , where  $\text{VC}(\mathcal{C}_{\text{learn}})$  is bounded. Let  $\mathcal{A}_{\mathcal{C}}$  be an algorithm that achieves a volume approximation against sets in  $\mathcal{C}_{\text{compete}}$  by outputting sets in  $\mathcal{C}_{\text{learn}}$ . Then the conformal algorithm will achieve coverage and volume optimality against the class  $\mathcal{C}_{\text{compete}}$ , with coverage slack that depends on  $\text{VC}(\mathcal{C}_{\text{learn}})$ . We write the proof for  $\mathcal{C}_{\text{compete}} = \mathcal{C}_{\text{learn}}$  for ease of exposition, but the proof works essentially unchanged for the case where one is a subset of the other.

*Proof.* Our method proceeds via split conformal prediction with the following score function. We use the first  $n$  samples  $Y_1, \dots, Y_n$ , along with our subroutine  $\mathcal{A}_{\mathcal{C}}$ , to calculate an approximately minimum volume set  $\widehat{C}$  that captures at least  $1 - \alpha' - \delta_{\mathcal{A}}$  probability mass of the empirical samples, and achieves a  $\mu_{\mathcal{A}}$  approximation factor in volume to the best set that captures  $1 - \alpha'$  fraction of the samples. Here  $\alpha'$  is a modified miscoverage parameter that we will set later.

We then define the score function via a nested set family that are its level sets as in [Gupta, Kuchibhotla, and Ramdas, 2022]. The nested set family contains  $\widehat{C}$  and the set  $\mathcal{Y}$  – the entire

domain that is guaranteed to achieve coverage 1. The set family may also have other subsets and supersets of  $\widehat{C}$  that are chosen adaptively with respect to the samples  $Y_1, \dots, Y_n$ . These are not necessary for our theoretical guarantees, but can only improve the performance in practice.

By the standard analysis of split conformal prediction [see e.g. Theorem 1.2 in [Angelopoulos, Barber, and Bates, 2025](#)], we have that regardless of the score function, the split conformal method guarantees marginal coverage over exchangeable data. That is, it outputs a prediction set  $\widehat{C}$  such that

$$\mathbb{P} \left[ Y_{2n+1} \in \widehat{C} \right] \geq 1 - \alpha.$$

Thus it only remains to show that approximate volume optimality holds with high probability.

Let  $\widehat{C}$  be the prediction set found by the subroutine on data  $Y_1, \dots, Y_n$ , with parameter  $\alpha'$  which we will specify later. Let  $P_n$  be the empirical distribution on  $Y_1, \dots, Y_n$ ,

$$P_n = \frac{1}{n} \sum_{i=1}^n \delta_{Y_i}.$$

By the properties of the subroutine, we have that  $\widehat{C}$  achieves (approximate) coverage,  $P_n(\widehat{C}) \geq 1 - \alpha' - \delta_{\mathcal{A}}$  and  $\widehat{C}$  gives us a volume approximation guarantee  $\text{vol}(\widehat{C}) \leq \mu_{\mathcal{A}} \text{vol}(\widetilde{C}_{(1-\alpha')} )$ , where  $\widetilde{C}_{(1-\alpha')}$  is the optimal set in our family  $\mathcal{C}$  for the observed samples  $Y_1, \dots, Y_n$ ,

$$\widetilde{C}_{(1-\alpha')} = \underset{C \in \mathcal{C}: P_n(C) \geq 1-\alpha'}{\operatorname{argmin}} \text{vol}(C).$$

Let  $P_{2n+1}$  denote the empirical distribution over the (realized)  $2n + 1$  exchangeable samples

$$P_{2n+1} = \frac{1}{2n+1} \sum_{i=1}^{2n+1} \delta_{Y_i}.$$

Then by [Lemma 2.8.1](#) we have with probability at least  $1 - \text{poly}(n)$  we

$$\sup_{C \in \mathcal{C}} |P_n(C) - P_{2n+1}(C)| \leq \delta_2, \quad (2.1)$$

where  $\delta_2 = \tilde{O}(\sqrt{\text{VC}(\mathcal{C})/n} + \sqrt{\log n/n})$ , where  $\tilde{O}(\cdot)$  hides polylogarithmic factors in  $\text{VC}(\mathcal{C})$ .

Conditioned on [\(2.1\)](#), we have that

$$P_{2n+1} \geq 1 - \alpha' - \delta_{\mathcal{A}} - \delta_2$$

Let  $C_{(1-\alpha'+\delta_2)}^*$  be the smallest volume set that achieves coverage  $1 - \alpha' + \delta_2$  on the true distribution, i.e.

$$C_{(1-\alpha'+\delta_2)}^* = \underset{C \in \mathcal{C}: P_{2n+1}(C) \geq 1-\alpha'+\delta_2}{\text{argmin}} \text{vol}(C)$$

Conditioned on [\(2.1\)](#), we have that

$$P_n(C_{(1-\alpha'+\delta_2)}^*) \geq 1 - \alpha'.$$

Thus, we have that  $C_{(1-\alpha'+\delta_2)}^*$  was a feasible choice for  $\tilde{C}_{(1-\alpha')}$ , and we can establish the volume bound

$$\text{vol}(\hat{C}) \leq \mu_{\mathcal{A}} \text{vol}(\tilde{C}_{(1-\alpha')}) \leq \mu \text{vol}(C_{(1-\alpha'+\delta_2)}^*).$$

Now we must argue that the approximately volume optimal  $\widehat{C}$  survives the conformalizing wrapper. Let  $P_{[n+1,2n]} = \frac{1}{n} \sum_{i=n+1}^{2n} \delta_{Y_i}$  be the empirical distribution on the calibration samples. Again applying uniform convergence we have that for  $\delta_2 = \widetilde{O}(\sqrt{\text{VC}(\mathcal{C})/n} + \sqrt{\log n/n})$ , with probability at least  $1 - \text{poly}(n)$ , we have

$$\sup_{C \in \mathcal{C}} |P_{[n+1,2n]}(C) - P_{2n+1}(C)| \leq \delta_2$$

Conditioned on this, as well as (2.1), we have that

$$P_{[n+1,2n]}(\widehat{C}) \geq 1 - \alpha' - \delta_1 - 2\delta_2$$

Taking  $\alpha' = \alpha - \delta_{\mathcal{A}} - 2\delta_2 - \frac{1}{n}$  ensures that the empirical coverage of  $\widehat{C}$  on the calibration samples  $Y_{n+1}, \dots, Y_{2n}$  will be at least  $1 - \alpha + \frac{1}{n}$ , and therefore the set output by split conformal prediction will be either  $\widehat{C}$  or a subset of  $\widehat{C}$ . Thus, with total probability at least  $1 - \text{poly}(n)$ , we have that the set output by split conformal prediction has volume at most

$$\text{vol}(\widehat{C}) \leq \mu_{\mathcal{A}} \text{vol}(C_{(1-\alpha'+\delta_2)}^{\star}) = \mu_{\mathcal{A}} \text{vol}(C_{(1-\alpha+\frac{1}{n}+\delta_{\mathcal{A}}+3\delta_2)}^{\star}).$$

Setting  $\delta_{\text{UC}} = 3\delta_2$  establishes the claim.  $\square$

## 2.6 Connections to Other Algorithmic and Statistical Problems

Learning small confidence sets (Problem 2.2.3) has implications in many statistical inference tasks, ranging from estimation to hypothesis testing. We will summarize a few important applications below.

1. *Conformal prediction.* The main focus of this exposition has been on the application of learning small confidence set (Problem 2.2.3) to conformal prediction (Problem 2.2.1). In Theorem 2.5.1, we make this connection formal: a subroutine for finding volume optimal confidence sets, from a family of bounded VC-dimension, on arbitrary datasets, immediately implies a volume-efficient conformal predictor.
2. *Robust estimation.* Suppose  $\hat{C}$  is a  $\delta$ -level confidence set with some  $\delta$  close to zero. It was suggested by Sager [1978, 1979] that one can use some point in  $\hat{C}$  as a mode estimator. More generally, robust location and scatter estimation through learning minimum volume sets is advocated by Rousseeuw [1984, 1985]. Specifically, Rousseeuw first computes  $\hat{C}$  as the minimum volume ellipsoid (MVEE), and then the center and the shape of the  $\hat{C}$  are used as a robust location estimator and a robust scatter matrix estimator. Other applications of MVEE are discussed in [Van Aelst and Rousseeuw, 2009].
3. *Testing unimodality.* The original motivation of Hartigan [1987] in learning a minimum volume convex set is to test whether a distribution is unimodal. The paper suggested testing statistic  $\sup_{\lambda} \sup_{C \in \mathcal{C}} E_{\lambda}(C)$  that maximizes the excess mass functional (2.3). The idea was further developed and generalized by Müller and Sawitzki [1991], Polonik [1995], Cheng and Hall [1998].
4. *Anomaly detection.* The problem of anomaly detection, also known as outlier or novelty detection, aims to find a set that is not typical. Given a distribution  $\mathcal{D}$  with density  $p$ , its anomaly region is  $\{y \in \mathbb{R}^d : p(y) < t\}$ , the complement of a density level set. Thus, the problem is mathematically equivalent to learning a confidence set. We refer the readers to [Chandola, Banerjee, and Kumar, 2009, Ruff, Kauffmann, Vandermeulen, Montavon, Samek, Kloft, Dietterich, and Müller, 2021] for comprehensive review of

this area.

5. *Goodness-of-fit.* Consider a null hypothesis  $H_0 : P = P_0$ . Whether data is generated from  $P_0$  can be determined by the value of  $|\mathbb{P}_n(C) - P_0(C)|$  for some set  $C$  that is informative. The idea of using minimum volume sets in the goodness-of-fit test was considered by Polonik [1999]. In particular, let  $\hat{C}_\alpha$  (resp.  $C_\alpha$ ) be a  $(1 - \alpha)$ -level confidence set computed from  $\mathbb{P}_n$  (resp.  $P_0$ ). The statistic

$$\sup_{\alpha} \left( |\mathbb{P}_n(\hat{C}_\alpha) - P_0(\hat{C}_\alpha)| + |\mathbb{P}_n(C_\alpha) - P_0(C_\alpha)| \right)$$

was proposed and its asymptotic property was analyzed by Polonik [1999].

## 2.7 Prior and Related Algorithmic Approaches

The problem of learning minimum volume sets has a long history in statistical learning and computer science. It also goes under the names of learning density level sets, anomaly detection, or distribution support estimation in the literature. Learning algorithms that have been studied in the literature can roughly be categorized into the following three classes.

1. *Excess mass functional.* The empirical version of our problem is

$$\min \text{vol}(C) \quad \text{s.t. } C \in \mathcal{C} \text{ and } \mathbb{P}_n(C) \geq \delta - \gamma, \quad (2.2)$$

where  $\mathbb{P}_n$  is the empirical distribution and  $\gamma$  is some slack parameter [Scott and Nowak, 2005]. A more general objective function called the excess mass functional proposed by Hartigan [1987] is defined by

$$E_\lambda(C) = \mathbb{P}_n(C) - \lambda \text{vol}(C), \quad (2.3)$$

where the parameter  $\lambda > 0$  is determined by the confidence level. In fact, maximizing the excess mass functional over  $C \in \mathcal{C}$  is equivalent to Equation (2.2). This approach has been well studied in the literature. The original work [Hartigan, 1987] considers  $\mathcal{C}$  being the collection of convex sets. The collection of ellipsoids is considered by Nolan [1991]. Convergence rates of the maximizer of Equation (2.3) are investigated by Tsybakov [1997] in a nonparametric setting for star-shaped sets and convex sets.

2. *One-class classification.* Another type of algorithms aim to learn a nonparametric function  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ , and the confidence set is taken as  $\{y \in \mathbb{R}^d : f(y) \geq 0\}$ . This approach was pioneered by Schölkopf, Platt, Shawe-Taylor, Smola, and Williamson [2001] and they model the function  $f$  in a feature space by

$$f(y) = \text{sign} \left( w^T \Phi(y) - \rho \right), \quad (2.4)$$

where  $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^k$  is some feature mapping. The parameters  $w$  and  $\rho$  are solutions to the following optimization problem,

$$\max_{w, \rho, \xi} \frac{1}{2} \|w\|^2 + \frac{1}{\lambda n} \sum_{i=1}^n \xi_i - \rho \quad \text{s.t.} \quad w^T \Phi(Y_i) - \rho + \xi_i \geq 0, \quad (2.5)$$

where  $\xi \in \mathbb{R}^n$  is a margin vector and  $\lambda$  is determined by the confidence level. The optimization (2.5) is known as the one-class support vector machine (SVM) in the literature, which is equivalent to another optimization problem called support vector data description (SVDD) [Tax and Duin, 2004] using some particular feature mapping. Recasting learning minimum volume sets as solving classification is natural. The excess mass functional (2.3) already suggests that the problem is equivalent to testing between two measures, empirical distribution against volume (uniform). Formally, this connection was established by Steinwart, Hush, and Scovel [2005]. Unlike traditional



classification, here all the data points are generated from the same distribution, which leads to the name “one-class classification” in the literature. Compared with directly learning confidence sets through (2.2) or (2.3), the one-class classification perspective is more flexible, and can be easily combined with deep learning models [Ruff, Vandermeulen, Goernitz, Deecke, Siddiqui, Binder, Müller, and Kloft, 2018].

3. *Density estimation plug-in.* The population version of the excess mass functional is  $P(C) - \lambda \text{vol}(C)$ , which is maximized by the density level set

$$\{y \in \mathbb{R}^d : p(y) \geq t\},$$

by Neyman–Pearson lemma, since the density function  $p$  can be regarded as the likelihood ratio between  $P$  and volume [Garcia, Kutalik, Cho, and Wolkenhauer, 2003]. Thus, it is natural to replace  $p$  by some density estimator  $\hat{p}$ . The plug-in strategy has been considered by Hyndman [1996], Park, Huang, and Ding [2010], Lei, Robins, and Wasserman [2013] among others. Under certain nonparametric setting, the optimality of the plug-in strategy can be proved [Rigollet and Vert, 2009].

## 2.8 Uniform Convergence of Bounded VC Classes over Exchangeable Random Variables

We show that exchangeability suffices for standard uniform convergence bounds based on VC dimension. While this is already known, we provide a proof for completeness. Most of the results on learning confidence sets in this thesis go through uniform convergence, and this shows that these results generalize from i.i.d. data to exchangeable data.

We will consider a domain  $\mathcal{X}$ , and a hypothesis class  $\mathcal{H}$  comprised of functions of the form

$h : \mathcal{X} \rightarrow \{0, 1\}$ .

For the sake of completeness, we write this theorem in terms of *classification error* (0-1 error). This is only stronger than ensuring uniform convergence of coverage, which can be thought of as the classification error with respect to all labels being 1.

Given a set of sample points  $\{S_1, \dots, S_T\}$  where  $S_i = (x_i, y_i) \in \mathcal{X} \times \{0, 1\}$  and a hypothesis  $h \in \mathcal{H}$ , we consider the 0-1 loss,

$$\ell(h, \{S_1, \dots, S_T\}) = \frac{1}{T} \sum_{i=1}^T \|h(x_i) - y_i\|.$$

For an exchangeable sequence of samples  $\mathbf{S} = (S_1, \dots, S_T)$ , we will overload notation and define  $\ell(h, \mathbf{S}) = \ell(h, \{S_1, \dots, S_T\})$ .

**Lemma 2.8.1** (Uniform Convergence for Exchangeable Sequences). *Let  $\mathcal{H}$  be a hypothesis class of functions from a domain  $\mathcal{X}$  to  $\{0, 1\}$  of VC-dimension  $d$ . Let  $\mathbf{S} = (S_1, \dots, S_T)$  be drawn from an exchangeable distribution  $\mathbf{P} = (P_1, \dots, P_T)$ , with  $S_i = (x_i, y_i) \in \mathcal{X} \times \{0, 1\}$ . Then there exists a universal constant  $C$  such that for any fixed setting of  $t \leq T, \delta > 0, \varepsilon > 0$  such that*

$$t \geq C \left( \frac{d \log(d/\varepsilon) + \log(1/\delta)}{\varepsilon^2} \right),$$

*we have that with probability  $\geq 1 - \delta$  over the randomness of  $\mathbf{S}$ , for all  $h \in \mathcal{H}$  simultaneously,*

$$\ell(h, \mathbf{S}) - \varepsilon \leq \ell(h, \{S_1, \dots, S_t\}) \leq \ell(h, \mathbf{S}) + \varepsilon.$$

*Remark 2.8.2* (Exchangeability vs. sampling with replacement). Note that the conclusion of [Lemma 2.8.1](#) is stated with respect to  $\ell(h, \mathbf{S})$  for the *realized*  $\mathbf{S}$  and *not* with respect to the expectation over the distribution  $\mathbb{E}_{\mathbf{S}' \sim \mathbf{P}} [\ell(h, \mathbf{S}')]$ . In general, these are not the same. This is

the correct object to consider because we want to argue that a classifier trained on a prefix of a sequence will generalize well to the suffix of the same sequence.

It is a fact that any finite sequence of exchangeable variables is expressible as a mixture of *urn* sequences (i.e., sampling with replacement) [Diaconis and Freedman, 1980]. In the case that  $\mathbf{P}$  is an urn distribution, we have that

$$\ell(h, \mathbf{S}) = \mathbb{E}_{\mathbf{S}' \sim \mathbf{P}} [\ell(h, \mathbf{S}')] \quad \text{deterministically, for any } \mathbf{S} \sim \mathbf{P}.$$

However, for a mixture, this may no longer be true. Take for example

$$\mathbf{P} = \begin{cases} (0, \dots, 0) & \text{w.p. } \frac{1}{2} \\ (1, \dots, 1) & \text{w.p. } \frac{1}{2} \end{cases}.$$

For any  $\mathbf{S} \sim \mathbf{P}$ , and any hypothesis  $h : \mathcal{X} \rightarrow \{0, 1\}$ , we have that the loss of  $h$  on a prefix of  $\mathbf{S}$  is exactly the same as the loss of  $h$  on the full  $\mathbf{S}$ . However, it is not the same as the expected loss on  $\mathbf{P}$ , which is always  $\frac{1}{2}$ .

In some sense, when we think of  $\mathbf{S}$  as one round of sampling with replacement, then there are many samples, so we can learn about the distribution. However, if we think of  $\mathbf{S}$  as drawn from a mixture distribution, it is as though we only see one draw from the mixture, so we have no hope of learning anything about it. However, our performance is only evaluated according to the single shot, so from our perspective, we may as well condition on the component of the mixture that  $\mathbf{S}$  is drawn from, and treat it as sampling with replacement.

To prove [Lemma 2.8.1](#), we follow the standard symmetrization argument for uniform convergence bounds, with some extra care to make them work with exchangeability. We use the following Hoeffding concentration bound for sampling with replacement.

**Lemma 2.8.3** (Hoeffding [1963], Serfling [1974]). *Let  $X_1, \dots, X_n$  be a sequence of exchangeable random variables sampled without replacement with  $X_i \in [0, 1]$  and mean  $\mathbb{E}[X_1] = \mu$ . Then we have the following upper tail bound*

$$\mathbb{P} \left[ \frac{1}{n} \sum_{i=1}^n X_i - \mu \geq \lambda \right] \leq \exp \left( -n\lambda^2/2 \right),$$

*and we get an identical bound the lower tail.*

We now proceed with the proof of Lemma 2.8.1.

*Proof.* Every finite length exchangeable distribution can be expressed as a mixture of *urn distributions* (sampling without replacement) [Diaconis and Freedman, 1980]. Thus, to establish the lemma, we can condition on the mixture component of  $\mathbf{P}$  that  $\mathbf{S}$  is drawn from, and assume without loss of generality that  $\mathbf{S}$  is sampled without replacement.

From the projection property of exchangeable sequences, the subsequence  $\mathbf{S} = (S_1, \dots, S_t)$  is also exchangeable. The proof of uniform convergence follows the standard symmetrization approach as given by Bousquet, Boucheron, and Lugosi [2004].

Let  $\mathbf{S}' = (S'_1, \dots, S'_t)$  be an independent sample from the same urn distribution as  $(S_1, \dots, S_t)$  (i.e.,  $\mathbf{S}'$  is sampled without replacement from the elements in the realized  $\mathbf{S}$ ).

First from Hoeffding's bound for sampling without replacement (Lemma 2.8.3), we have

$$\mathbb{P} \left[ \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) \geq \lambda \right] \leq \exp \left( -t\lambda^2/2 \right), \text{ and } \mathbb{P} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}) \geq \lambda \right] \leq \exp \left( -t\lambda^2/2 \right). \quad (2.6)$$

We first prove the following claim through symmetrization. The claim allows us to relate the error on one sample to the discrepancy between two independent samples from the

distribution.

**Claim 2.8.1.** *For any  $\lambda > 0$ , such that  $t\lambda^2 \geq 2$ ,*

$$\mathbb{P} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) > \lambda \right] \leq 2 \mathbb{P} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) > \lambda/2 \right] \quad (2.7)$$

*Proof.* Let  $h^* \in \mathcal{H}$  be a hypothesis that achieves the supremum on the left side of (2.7). We will lower bound the probability of the event on the right by showing that it is satisfied for  $h^*$ . For any  $h \in \mathcal{H}$ ,

$$\mathbf{1} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) > \lambda/2 \right] \geq \mathbf{1} \left[ (\ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) > \lambda) \wedge (\ell(h, \mathbf{S}) - \ell(h, \mathbf{S}'_t) < \lambda/2) \right].$$

Note that (conditioned on the mixture component of the exchangeable distribution, i.e. the multiset of values in  $\mathbf{S}$ )  $\mathbf{S}'_t$  is independent of  $\mathbf{S}_t$ . Taking expectation on both sides w.r.t.  $\mathbf{S}'_t$ ,

$$\mathbb{P}_{\mathbf{S}'_t} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) > \lambda/2 \right] \geq \mathbf{1} \left[ \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) > \lambda \right] \cdot \mathbb{P}_{\mathbf{S}'_t} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}) > -\lambda/2 \right]. \quad (2.8)$$

The above inequality holds for any fixed hypothesis  $h \in \mathcal{H}$ , including any  $h^*$  that achieves the supremum of the left side of (2.7). The event on the right does not depend on  $\mathbf{S}_t$ , and follows by applying the Chebyshev inequality on the exchangeable sequence  $\mathbf{S}'_t = (S'_1, \dots, S'_t)$ . Hence for any fixed hypothesis  $h$  (including  $h^*$ )

$$\mathbb{P}_{\mathbf{S}'_t} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}) \leq -\lambda/2 \right] \leq \frac{\text{Var}[\ell(h, \mathbf{S}'_t)]}{\lambda^2} \leq \frac{1}{4t\lambda^2} \leq \frac{1}{2}.$$

The variance upper bound for  $\ell(h, \mathbf{S}'_t)$  is due sampling with replacement. Substituting in

(2.8) and taking expectation w.r.t.  $\mathbf{S}_t$ , we have

$$\mathbb{P} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) > \lambda/2 \right] \geq \mathbb{P}_{\mathbf{S}_t} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) > \lambda \right] \times \frac{1}{2}.$$

By rearranging terms, we get the claim.  $\square$

We now upper bound the right side of (2.7). For a fixed  $h \in \mathcal{H}$ , we have from (2.6)

$$\begin{aligned} \mathbb{P}_{\mathbf{S}_t, \mathbf{S}'_t} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) \geq \frac{\lambda}{2} \right] &\leq \mathbb{P} \left[ \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}) \geq \frac{\lambda}{4} \right] + \mathbb{P} \left[ \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) \geq \frac{\lambda}{4} \right] \\ &\leq 2 \exp \left( - \frac{t\lambda^2}{32} \right). \end{aligned}$$

Now we observe that for a given hypothesis  $h \in \mathcal{H}$ ,  $\ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t)$  is an empirical sum and only depends on the  $2n$  samples  $S_1, \dots, S_t, S'_1, \dots, S'_t$ . Hence the number of distinct hypothesis that we need to union bound over is at most the shattering number  $N(2t, d) = \sum_{j=1}^d \binom{2t}{d}$ . Hence

$$\begin{aligned} \mathbb{P} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}) - \ell(h, \mathbf{S}_t) > \lambda \right] &\leq 2 \mathbb{P}_{\mathbf{S}_t, \mathbf{S}'_t} \left[ \sup_{h \in \mathcal{H}} \ell(h, \mathbf{S}'_t) - \ell(h, \mathbf{S}_t) \geq \frac{\lambda}{2} \right] \\ &\leq 4N(2t, d) \cdot \exp \left( - \frac{t\lambda^2}{32} \right) \leq \delta, \end{aligned}$$

for our choice of  $t$  and  $\lambda$ . This finishes the proof.  $\square$

## Chapter 3

# The Volume Optimality Framework

This chapter is based on *Volume Optimality in Conformal Prediction with Structured Prediction Sets*, which is joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan, and appeared at the International Conference on Machine Learning (ICML) 2025 [Gao, Shan, Srinivas, and Vijayaraghavan, 2025b].

In this chapter, we study conformal prediction with volume optimality in both the unsupervised setting and the supervised setting. We begin by formulating a new definition of *restricted volume optimality*, which is tractable in a distribution-free sense.

For the one dimensional setting, where  $\mathcal{Y} = \mathbb{R}$ , we achieve this definition with new conformity score computed via dynamic programming. In the supervised setting, when consistent estimation of the conditional CDF is available, we prove that the proposed method not only achieves conditional coverage, but the output of prediction set also has approximate conditional volume optimality with respect to the class of unions of  $k$  intervals.

Our method is especially suitable to settings where the data generating process is multi-

modal or has a mixture structure. The numerical experiments show that the performance of the method is quite insensitive to the choice of  $k$ , whenever it is not smaller than the number of modes of the distribution.

### 3.1 Introduction

Conformal inference has emerged as a powerful black-box method for quantifying uncertainty in model predictions, providing confidence sets or prediction sets that contain the true value with a specified probability [Gammerman, Vovk, and Vapnik, 1998, Vovk, Gammerman, and Shafer, 2005]. Recall the standard conformal prediction problem introduced in Chapter 2.

**Problem 2.1.1** (Conformal prediction (marginal coverage)).

*Let  $(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1}) \in \mathcal{X} \times \mathcal{Y}$  be exchangeable random variables. That is, the data points  $(X_i, Y_i)$  are in uniformly random order (formally Definition 2.1.2).*

*A conformal predictor that achieves marginal coverage takes*

- *a target miscoverage rate  $\alpha \in (0, 1]$*
- *a black-box model  $\hat{Y} = f(X)$  to predict target variable  $Y \in \mathcal{Y}$  from feature variable  $X \in \mathcal{X}$ ,*
- *calibration data  $(X_1, Y_1), \dots, (X_n, Y_n)$ , and*
- *the test feature  $X_{n+1}$ ,*

*and outputs a prediction set  $C = C((X_1, Y_1), \dots, (X_n, Y_n), X_{n+1}) \subseteq \mathcal{Y}$  such that*

$$\mathbb{P}_{(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1})} [Y_{n+1} \in C] \geq 1 - \alpha. \quad (\text{MC})$$



In addition, we would like the conformal predictor to be efficient, in that it outputs prediction sets  $C$  that are as small as possible.

The above definition assumes *exchangeability* of the random variables. We recall the definition of exchangeability.

**Definition 2.1.2** (Exchangeability). *A sequence  $(Z_1, \dots, Z_n)$  is exchangeable if*

$$(Z_1, \dots, Z_T) \stackrel{d}{=} (Z_{\sigma(1)}, \dots, Z_{\sigma(T)}),$$

for all permutations  $\sigma : [T] \rightarrow [T]$ , where  $\stackrel{d}{=}$  means they are equal in distribution.

Here the probability  $\mathbb{P}$  refers to the joint distribution over all  $n + 1$  pairs of observations  $(X_1, Y_1), \dots, (X_n, Y_n), (X_{n+1}, Y_{n+1})$  including the test sample  $(X_{n+1}, Y_{n+1})$ . Unlike traditional approaches, conformal inference is distribution-free, relying only on the assumption of exchangeability of the joint distribution  $\mathbb{P}$  over the  $(n + 1)$  samples.

While most conformal methods provide guarantees on coverage, they do not provide any control on the size or volume of the prediction sets.<sup>1</sup> In fact, the choice of  $\widehat{C}(X_{n+1}) = \mathcal{Y}$  also satisfies the coverage requirement. Consequently, the size of these sets is often validated empirically, without formal guarantees. This raises the important question of *volume optimality*, which is the focus of this chapter:

**Question 3.1.1** (Volume-optimal prediction sets). *Given calibration samples  $(X_1, Y_1), \dots, (X_n, Y_n)$  drawn i.i.d. from a distribution  $P$ , can we find among all data-dependent sets  $\widehat{C} \subset \mathcal{Y}$  satisfying the desired coverage requirement for  $(X_{n+1}, Y_{n+1}) \sim P$ , the one with the smallest volume, as quantified by the Lebesgue measure  $\text{vol}(\widehat{C}) = \lambda(\widehat{C})$ ?*

The volume of the prediction set in conformal prediction is also referred to as “efficiency” has

---

<sup>1</sup>Although conformal prediction ensures that the coverage probability does not exceed  $1 - \alpha + 1/n$ , it does not prevent overly conservative predictions.

been stated as an important consideration in many prior works [see e.g., [Shafer and Vovk, 2008](#), [Vovk, Fedorova, Nouretdinov, and Gammernan, 2016](#), [Angelopoulos, Barber, and Bates, 2025](#)]. However, most works that we are aware of do not give theoretical guarantees of volume optimality, and mainly reason about volume control through empirical evaluations.<sup>2</sup>

Notable exceptions include the works of [Lei, Robins, and Wasserman \[2013\]](#) in the unsupervised setting, [Lei \[2014\]](#), [Vovk, Fedorova, Nouretdinov, and Gammernan \[2016\]](#), [Sadinle, Lei, and Wasserman \[2016\]](#) in the classification setting, [Lei, Rinaldo, and Wasserman \[2015\]](#) for functional data and recent works of [Izbicki, Shimizu, and Stern \[2020a,b\]](#) and [Kiyani, Pappas, and Hassani \[2024\]](#) in the regression setting. As summarized by [Angelopoulos, Barber, and Bates \[2025\]](#), a sufficient condition that leads to volume optimality of conformal prediction is consistent estimation of the conditional density function of  $Y$  given  $X$ . This is essentially the strategy adopted by previous work [[Lei, Robins, and Wasserman, 2013](#), [Izbicki, Shimizu, and Stern, 2020a,b](#)]. In comparison, our method, by incorporating a framework of [Chernozhukov, Wüthrich, and Zhu \[2021\]](#), builds on the estimation of the conditional CDF via a new conformity score computed by dynamic programming, and thus also works in settings where good conditional density estimation is impossible or density does not even exist.

### 3.1.1 Results in [Chapter 3](#)

**An Impossibility Result.** We first prove an impossibility result in a one-dimensional setting where any distribution-free method that satisfies the coverage requirement can only find a trivial solution whose volume is sub-optimal. See [Theorem 3.2.2](#) for a formal statement.

---

<sup>2</sup>Some works also guarantee that the coverage is not much more than  $1 - \alpha$ , e.g.,  $\mathbb{P}\left(Y_{n+1} \in \widehat{C}(X_{n+1}) \mid X_{n+1}\right) \leq 1 - \alpha + o(1)$  to argue that the prediction set is not too big. However, smallness according to the measure  $\mathbb{P}$  does not necessarily reflect a small volume (or Lebesgue measure) for the set.

This result provides an explanation for the lack of such volume-optimality guarantees in the conformal prediction literature, and also motivates our new notion of volume-optimality that we introduce in this work.

**Structured Prediction Sets and Restricted Volume Optimality.** Motivated by the impossibility result, our goal is to find a prediction set  $\hat{C} \in \mathcal{C} \subset 2^{\mathcal{Y}}$  whose volume is competitive with the optimum volume of any set in the family  $\mathcal{C}$  as given by

$$\text{OPT}_{\mathcal{C}}(P, 1 - \alpha) = \inf_{C \in \mathcal{C}} \{\text{vol}(C) : P(C) \geq 1 - \alpha\}. \quad (3.1)$$

As long as  $\mathcal{C}$  has bounded VC-dimension, for any distribution  $P$  we can obtain good empirical estimates of the probability measure of any set  $C \in \mathcal{C}$  via a standard uniform concentration inequality, which allows us to overcome the impossibility result in [Theorem 3.2.2](#). In the rest of the chapter, we focus on the setting when  $\mathcal{Y} = \mathbb{R}$  and  $\mathcal{C} = \mathcal{C}_k$  which is the collection of unions of  $k$  intervals.

**Conformalized Dynamic Programming.** Equipped with our new notion of volume optimality, we propose a new conformity score function for  $\mathcal{Y} = \mathbb{R}$  based on dynamic programming. The proposed method is shown to not only achieve approximate conditional coverage as in [Chernozhukov, Wüthrich, and Zhu \[2021\]](#) and [Romano, Patterson, and Candès \[2019\]](#), but also conditional volume optimality with respect to unions of  $k$  intervals, as long as a reasonable estimator of the conditional CDF is available. Our method of learning a predictive set via CDF can be regarded an extension of the framework of [Izbicki, Shimizu, and Stern \[2020a\]](#), [Chernozhukov, Wüthrich, and Zhu \[2021\]](#).

## 3.2 Unsupervised Setting

### 3.2.1 Approximate Volume Optimality

Suppose  $Y_1, \dots, Y_n, Y_{n+1}$  are independently drawn from a distribution  $P$  on  $\mathbb{R}$ . The goal is to predict  $Y_{n+1}$  based on the first  $n$  samples  $Y_1, \dots, Y_n$ . To be specific, we would like to construct a data-dependent set  $\hat{C} = \hat{C}(Y_1, \dots, Y_n)$  such that

$$\mathbb{P}(Y_{n+1} \in \hat{C}) \geq 1 - \alpha. \quad (3.2)$$

Among all data-dependent sets that satisfy (3.2), our goal is to find the one with the smallest volume, quantified by the Lebesgue measure  $\text{vol}(\hat{C}) = \lambda(\hat{C})$ . When the distribution  $P$  is known, one can directly minimize  $\lambda(C)$ , subject to  $P(C) \geq 1 - \alpha$  without even using the data. In particular, when  $P \ll \lambda$ , an optimal solution is given by the density level set

$$C_{\text{opt}} = \left\{ \frac{dP}{d\lambda} > t \right\} \cup D,$$

for some  $t > 0$  and  $D$  is some subset of  $\{dP/d\lambda = t\}$ .

In general,  $P$  may not be absolutely continuous and the density need not exist. Nonetheless, we can still define the optimal volume by

$$\text{OPT}(P, 1 - \alpha) = \inf\{\text{vol}(C) : P(C) \geq 1 - \alpha\}.$$

Note that without any assumption on  $P$ , the above optimization problem may not have a unique solution. Moreover, it is possible that the infimum cannot be achieved by any measurable set. Therefore, a natural relaxation is to consider approximate volume optimality.

**Problem 3.2.1** (Approximate Volume-Optimal Conformal Prediction). *For an arbitrary*

distribution  $P$  over  $\mathcal{Y}$ , a target miscoverage rate  $\alpha \in [0, 1)$ , and a parameter  $\varepsilon > 0$ , approximately volume-optimal conformal method sees  $n$  i.i.d. samples from  $P$   $Y_1, \dots, Y_n$ , and outputs a prediction set  $C$  that

(1) achieves coverage,

$$P[C] \geq 1 - \alpha. \quad (3.3)$$

(2) achieves volume optimality,

$$\text{vol}(C) \leq \text{OPT}(P, 1 - \alpha + \varepsilon), \quad (3.4)$$

where

$$\text{OPT}(P, 1 - \alpha) := \inf\{\text{vol}(C) : P(C) \geq 1 - \alpha\}.$$

The notion of volume optimality defined in [Problem 3.2.1](#) is quite different from those considered in the literature. A popular quantity that has already been studied is the volume of set difference  $\text{vol}(\widehat{C} \Delta C_{\text{opt}})$  [[Lei, Robins, and Wasserman, 2013](#), [Izbicki, Shimizu, and Stern, 2020a](#), [Chernozhukov, Wüthrich, and Zhu, 2021](#)]. However, this much stronger notion requires that the optimal solution  $C_{\text{opt}}$  must not only exist but also be unique. Usually additional assumptions need to be imposed in the neighborhood of the boundary of  $C_{\text{opt}}$  in order that the set difference vanishes in the large sample limit. In comparison, the definition (3.4) only requires the volume to be controlled, which can be achieved even if  $\widehat{C}$  is not close to  $C_{\text{opt}}$ , or when  $C_{\text{opt}}$  does not even exist. Indeed, from a practical point of view, any set with coverage and volume control would serve the purpose of valid prediction. Insisting the closeness to a questionable target  $C_{\text{opt}}$  comes at the cost of unnecessary assumptions on the data generating process.

Another notion considered in the literature is close to our formulation [Problem 3.2.1](#). Instead of relaxing the coverage probability level from  $1 - \alpha$  to  $1 - \alpha + \varepsilon$ , one can consider the following approximate volume optimality,

$$\text{vol}(\widehat{C}) \leq \text{OPT}(P, 1 - \alpha) + \varepsilon. \quad (3.5)$$

Results of interval length optimality in the sense of (3.5) have been studied by [Chernozhukov, Wüthrich, and Zhu \[2021\]](#), [Kiyani, Pappas, and Hassani \[2024\]](#). However, the  $\varepsilon$  in (3.5) is usually proportional to the scale of the distribution  $P$ , or may depend on  $P$  in some other ways. In comparison, the  $\varepsilon$  in [Problem 3.2.1](#) has the unit of probability, and as we will show later, can be made independent of the distribution  $P$ , which leads to more natural and cleaner theoretical results with fewer assumptions.

### 3.2.2 Impossibility of Distribution-Free Volume Optimality

It is known that conformal prediction achieves the coverage property (3.3) in a distribution-free sense, meaning that (3.3) holds uniformly for all distributions  $P$ . One naturally hopes that the approximate volume optimality (3.4) can also be established in a distribution-free way. Perhaps not surprisingly, this goal is too ambitious. The theorem below rigorously proves the impossibility of the task. The detailed proof is deferred to [Section B.3.1](#).

**Theorem 3.2.2.** *Consider observations  $Y_1, Y_2, \dots, Y_n, Y_{n+1}$  sampled i.i.d. from a distribution  $P$  on  $\mathbb{R}$ . Suppose  $\widehat{C} = \widehat{C}(Y_1, \dots, Y_n)$  satisfies  $\mathbb{P}(Y_{n+1} \in \widehat{C}) \geq 1 - \alpha$  for all distribution  $P$ . Then, for any  $\varepsilon \in (0, \alpha)$ , there exists some distribution  $P$  on  $\mathbb{R}$ , such that the expected volume of the prediction set is at least*

$$\mathbb{E} \text{vol}(\widehat{C}) \geq \text{OPT}(P, 1 - \alpha + \varepsilon).$$

The above impossibility result can be regarded as a consequence of a nonparametric testing lower bound. Consider the following hypothesis testing problem,

$$\begin{aligned} H_0 : \quad & P = P_0 \\ H_1 : \quad & P \in \{P : \text{TV}(P, P_0) > 1 - \delta\}. \end{aligned}$$

It is well known that a testing procedure with both vanishing Type-1 and Type-2 errors does not exist without further constraining the alternative hypothesis, even when  $\delta$  is arbitrarily close to 0 [LeCam and Schwartz, 1960, Barron, 1989]. In the setting of distribution-free inference with simultaneous coverage and volume guarantees, two different probability measures naturally arise. The coverage guarantee is defined with respect to the joint distribution  $P^{n+1}$ , which governs the full dataset of  $n$  training samples and one test point. In contrast, the expected volume of the prediction set is measured under the product distribution  $P^n \otimes \lambda$ , where  $P^n$  represents the joint distribution of  $n$  training samples. When restricting the support of  $P$  to the unit interval  $[0, 1]$ ,  $\lambda$  becomes the uniform probability, and thus both  $P^{n+1}$  and  $P^n \otimes \lambda$  are probability distributions. It turns out achieving approximate volume optimality is related to hypothesis testing between  $P^{n+1}$  and  $P^n \otimes \lambda$  with total variation separation.

### 3.2.3 Distribution-Free Restricted Volume Optimality

The impossibility result implies a volume lower bound  $\text{OPT}(P, 1 - \alpha + \varepsilon)$ , where the coverage level  $1 - \alpha + \varepsilon$  can be arbitrarily close to 1. This means that, at least in the worst case, the volume cannot be smaller than that of the support of  $P$ .

To avoid this impossibility, we introduce *restricted volume optimality* (restated from Chapter 2).

**Problem 2.2.1** (*C*-Restricted Volume-Optimal Conformal Prediction). *Let  $\mathcal{C}$  be a hypothesis*

class of prediction sets over  $\mathcal{Y} = \mathbb{R}^d$ , and let  $P$  be an arbitrary distribution over  $\mathbb{R}^d$ .

For a volume-approximation factor  $\mu \geq 1$  and a coverage slack factor  $\varepsilon > 0$ , A conformal predictor satisfying  $\mathcal{C}$ -restricted volume optimality, takes

- a target miscoverage rate  $\alpha \in (0, 1]$ , and
- $n$  samples  $Y_1, \dots, Y_n$  that are exchangeable with an unseen test example  $Y_{n+1}$

and outputs a prediction set  $C = C(Y_1, \dots, Y_n) \subseteq \mathbb{R}^d$  such that  $C$  achieves coverage,

$$P(C) \geq 1 - \alpha. \tag{cov}$$

and, with high probability,  $C$  is approximately volume-optimal,

$$\text{vol}(C) \leq \mu \cdot \text{OPT}_{\mathcal{C}}(P, 1 - \alpha + \varepsilon), \tag{VO}$$

where

$$\text{OPT}_{\mathcal{C}}(P, 1 - \alpha) := \inf\{\text{vol}(C) : C \in \mathcal{C}, P(C) \geq 1 - \alpha\}.$$

In this chapter, we focus on the setting where  $\mathcal{Y} = \mathbb{R}$ . In this setting, a natural class of prediction sets is the set of unions of  $k$  intervals, which we denote  $\mathcal{C}_k$ ,

$$\mathcal{C}_k = \left\{ S : \bigcup_{i=1}^k [u_i, v_i], u_i \leq v_i \in \mathbb{R} \right\}.$$

We give efficient algorithms that are able to achieve volume approximation factor  $\mu = 1$  for this class of prediction sets.

*Remark 3.2.3.* We remark that we are still in a distribution-free setting, since no assumption is imposed on  $P$ . Instead, the restriction only constrains the shape of the prediction set. From



a practical point of view, it is reasonable to require that  $\widehat{C} \in \mathcal{C}_k$ , since a more complicated prediction set would be hard to interpret. Moreover, as long as  $P$  admits a density function with at most  $k$  modes, the two notions match,

$$\text{OPT}_{\mathcal{C}_k}(P, 1 - \alpha) = \text{OPT}(P, 1 - \alpha).$$

More generally, it can be shown that

$$\text{OPT}_{\mathcal{C}_k}(P, 1 - \alpha) \leq \text{OPT}(P, 1 - \alpha + \varepsilon),$$

for some  $\varepsilon \in (0, \alpha)$ , whenever  $P$  can be approximated by a distribution with at most  $k$  modes. This, in particular, includes the situation where the density of  $P$  can be well estimated by a kernel density estimator. A rigorous statement is given in [Appendix B.2](#).

Given the observations  $Y_1, \dots, Y_n$ , we define the empirical distribution  $\mathbb{P}_n = \frac{1}{n} \sum_{i=1}^n \delta_{Y_i}$ . To achieve restricted volume optimality, one can use

$$\widehat{C} = \underset{C \in \mathcal{C}_k}{\text{argmin}} \{ \text{vol}(C) : \mathbb{P}_n(C) \geq 1 - \alpha \}. \quad (3.6)$$

According to its definition, the prediction set [\(3.6\)](#) satisfies both  $\mathbb{P}_n(\widehat{C}) \geq 1 - \alpha$  and  $\text{vol}(\widehat{C}) = \text{OPT}_k(\mathbb{P}_n, 1 - \alpha)$ . The coverage and volume guarantees under  $P$  can be obtained via

$$\sup_{C \in \mathcal{C}_k} |\mathbb{P}_n(C) - P(C)| = O_P \left( \sqrt{\text{VC}(\mathcal{C})/n} \right), \quad (3.7)$$

with  $\text{VC}(\mathcal{C}) = O(k)$ . Therefore, approximate optimality can be achieved by [\(3.6\)](#) whenever [\(3.7\)](#) holds.

In particular, for general prediction set families  $\mathcal{C}$ , this means that bounded VC dimension of

$\mathcal{C}$  is sufficient to achieve a volume approximation of  $\mu = 1$  statistically efficiently. However, different choices of  $\mathcal{C}$  may or may not admit computationally efficient algorithms to achieve small volume approximations.

For  $\mathcal{C} = \mathcal{C}_k$ , a naive exhaustive search to find (3.6) requires exponential computational time. We show that an efficient dynamic programming algorithm (Algorithm 3.1) can solve (3.6) approximately with some additional slack  $\gamma \in [1/n, \alpha)$ , which controls the trade-off between volume approximation accuracy and the computational complexity.

The dynamic programming table  $\text{DP}(i, j, l)$  stores the minimum volume of  $i$  intervals that cover  $l\gamma n$  points in  $Y_{(1)}, \dots, Y_{(j)}$  and the right endpoint of the rightmost interval is at  $Y_{(j)}$ , where  $Y_{(1)}, \dots, Y_{(n)}$  are training data points  $Y_1, \dots, Y_n$  sorted in non-decreasing order. For each state in DP table, we enumerate all possible left endpoint of the rightmost interval and the right endpoint of the previous interval (if it exists). After filling the table, a standard backtracking procedure is used to construct the final prediction set  $\hat{C}_{\text{DP}}$ .

Theoretical guarantees of Algorithm 3.1 are given in the following proposition.

**Proposition 3.2.4.** *For any  $\gamma \in [1/n, \alpha)$ , Algorithm 3.1 computes a prediction set  $\hat{C}_{\text{DP}} \in \mathcal{C}_k$  by dynamic programming with time complexity  $O(n^3 k / \gamma)$  such that*

1.  $\mathbb{P}_n(\hat{C}_{\text{DP}}) \geq 1 - \alpha;$
2.  $\text{vol}(\hat{C}_{\text{DP}}) \leq \text{OPT}_k(\mathbb{P}_n, 1 - \alpha + \gamma).$

Together with (3.7), the coverage and volume guarantees of the dynamic programming can also be generalized from  $\mathbb{P}_n$  to  $P$ .

**Algorithm 3.1** Dynamic Programming Solving (3.6)

---

**Require:** data points  $Y_1, \dots, Y_n \in \mathbb{R}$ , coverage level  $1 - \alpha \in (0, 1)$  and slack  $\gamma \in (0, \alpha)$ , number of intervals  $k$

**Ensure:**  $k$  intervals that cover  $\lceil (1 - \alpha)n \rceil$  points with minimum volume

- 1: Sort data in non-decreasing order  $Y_{(1)} \leq \dots \leq Y_{(n)}$
- 2: Set  $\text{DP}(i, j, l) = \infty$  for all  $i \in [k], j \in [n], l \in [1/\gamma]$
- 3: **for**  $i = 1$  **to**  $k$ ,  $j = 1$  **to**  $n$ ,  $l = 1$  **to**  $\lceil 1/\gamma \rceil$  **do**
- 4:     **for**  $j' = i$  **to**  $j$  **do**
- 5:         **for**  $j'' = i - 1$  **to**  $j' - 1$  **do**
- 6:             Set  $l' = l - \lfloor (j - j' + 1)/(\gamma n) \rfloor$
- 7:             **if**  $l' < 0$  **and**  $i = 1$  **then**
- 8:                  $\text{DP}(i, j, l) = \min\{\text{DP}(i, j, l), Y_{(j)} - Y_{(j')}\}$
- 9:             **end if**
- 10:            **if**  $\text{DP}(i - 1, j'', l') \neq \infty$  **then**
- 11:                 $\text{DP}(i, j, l) = \min\{\text{DP}(i, j, l), Y_{(j)} - Y_{(j')} + \text{DP}(i - 1, j'', l')\}$
- 12:            **end if**
- 13:         **end for**
- 14:     **end for**
- 15: **end for**
- 16: Find the minimum volume among all  $\text{DP}(k, j, \lceil (1 - \alpha)/\gamma \rceil)$  for  $j = 1, \dots, n$  and backtrack to construct the prediction set  $\hat{C}_{\text{DP}}$ .
- 17: Return the set  $\hat{C}_{\text{DP}}$ .

---

### 3.2.4 Conformalizing Dynamic Programming

Having understood the generalization ability of dynamic programming, we are ready to conformalize the procedure to achieve a finite-sample coverage property. For simplicity, we will adopt the framework of split conformal prediction, though in principle full conformal prediction can also be applied here.

In the split conformal prediction framework, the data set is split into two halves. The first half is used to compute a conformity score, and the second half determines the quantile level. For convenience of notation, let us assume, from now on, that the sample size is  $2n$ . The split conformal procedure is outlined below.

1. Compute a score function  $q(\cdot)$  using  $Y_1, \dots, Y_n$ .

2. Evaluate  $q(Y_{n+1}), \dots, q(Y_{2n})$ , and order them as  $q_1 \leq \dots \leq q_n$ .
3. Output the prediction set

$$\hat{C} = \{y : q(y) \geq q_{\lfloor (n+1)\alpha \rfloor}\}. \quad (3.8)$$

By the exchangeability of  $Y_{n+1}, \dots, Y_{2n}, Y_{2n+1}$ , the prediction set  $\hat{C}$  satisfies

$$\mathbb{P}(Y_{2n+1} \in \hat{C}) \geq 1 - \alpha,$$

where the above probability is over the randomness of  $(Y_1, \dots, Y_n)$ , that of  $(Y_{n+1}, \dots, Y_{2n})$ , and that of  $Y_{2n+1}$ .

To conformalize the dynamic programming that approximately computes (3.6), we will first compute a nested system  $S_1 \subset \dots \subset S_m \subset \mathbb{R}$  using the data  $Y_1, \dots, Y_n$ . The nested system is required to satisfy the following assumption.

**Assumption 3.2.5.** *The sets  $S_1 \subset \dots \subset S_m \subset \mathbb{R}$  are measurable with respect to the  $\sigma$ -field generated by  $Y_1, \dots, Y_n$ . Moreover, for some positive integer  $k$ , some  $\alpha \in (0, 1)$  and some  $\delta, \gamma$  such that  $3\delta + \gamma + n^{-1} \leq \alpha$ , we have*

1.  $\mathbb{P}_n(S_j) = \frac{j}{m}$  and  $S_j \in \mathcal{C}_k$  for all  $j \in [m]$ .
2. There exists some  $j^* \in [m]$ , such that
 
$$\mathbb{P}_n(S_{j^*}) \geq 1 - \alpha + n^{-1} + 3\delta \text{ and}$$

$$\text{vol}(S_{j^*}) \leq \text{OPT}_k(\mathbb{P}_n, 1 - \alpha + \frac{1}{n} + 3\delta + \gamma).$$

Here,  $\mathbb{P}_n$  denotes the empirical distribution  $\frac{1}{n} \sum_{i=1}^n \delta_{Y_i}$  of the first half of the data.

To construct a nested system  $\{S_j\}_{j \in [m]}$  that satisfies the above assumption, one only needs

to make sure that there exists one subset  $S_{j^*}$  in the system that is computed by the dynamic programming (Algorithm 3.1) with confidence level  $1 - \alpha + n^{-1} + 3\delta$  and slack parameter  $\gamma$ . The rest of the sets in the system can be constructed just to satisfy  $\mathbb{P}_n(S_j) = \frac{j}{m}$ . In Section 3.4.1, we will present a greedy expansion/contraction algorithm that satisfies Assumption 3.2.5.

With a nested system  $\{S_j\}_{j \in [m]}$  satisfying Assumption 3.2.5, we can define the conformity score as

$$q(y) = \sum_{j=1}^m \mathbb{I}\{y \in S_j\}. \quad (3.9)$$

The equivalence between nested system and conformity score was advocated by Gupta, Kuchibhotla, and Ramdas [2022]. Intuitively,  $q(y)$  quantifies the depth of the location  $y$ . A higher score implies that  $y$  is covered by more sets in the nested system, and thus the location should be more likely to be included in the prediction set. Applying the standard split conformal framework, our prediction set based on conformalized dynamic programming is defined by (3.8) with the conformity score (3.9).

**Theorem 3.2.6** (Unsupervised volume optimal conformal prediction). *Consider exchangeable observations  $Y_1, \dots, Y_{2n}, Y_{2n+1}$  on  $\mathbb{R}$ . Let  $\hat{C}_{\text{CP-DP}}$  be the split conformal prediction set (with training and calibration samples  $Y_1, \dots, Y_{2n}$ ) defined by the score (3.9) based on a nested system  $\{S_j\}_{j \in [m]}$  satisfying Assumption 3.2.5. Suppose the parameter  $\delta$  in Assumption 3.2.5 satisfies  $\delta \gg \sqrt{\frac{k + \log n}{n}}$ . Then the following properties hold.*

1. Coverage:

$$\mathbb{P}\left(Y_{2n+1} \in \hat{C}_{\text{CP-DP}}\right) \geq 1 - \alpha.$$

2. *Restricted volume optimality:*

$$\text{vol}(\widehat{C}_{\text{CP-DP}}) \leq \text{OPT}_k \left( P, 1 - \alpha + \frac{1}{n} + \delta + \gamma \right),$$

with probability at least  $1 - \text{poly}(n)$ .

We emphasize that [Theorem 3.2.6](#) guarantees both distribution-free coverage and distribution-free volume optimality properties. In practice,  $k$  can be chosen to be a constant for prediction interpretability. By setting  $\gamma = O\left(\sqrt{\frac{\log n}{n}}\right)$ , the volume sub-optimality is at most  $\frac{1}{n} + 4\delta + \gamma = O\left(\sqrt{\frac{\log n}{n}}\right)$ .

*Proof.* Direct application of our meta-theorem for volume optimal conformal prediction, [Theorem 2.5.1](#), using the dynamic programming procedure as a subroutine.  $\square$

### 3.3 Supervised Setting

#### 3.3.1 Problem Setting

In this section, we consider conformal prediction with labeled data. Suppose data points  $(X_1, Y_1), \dots, (X_{2n}, Y_{2n}), (X_{2n+1}, Y_{2n+1})$  are *i.i.d.* drawn from a distribution  $P$  on  $\mathcal{X} \times \mathcal{Y}$  with  $\mathcal{Y} = \mathbb{R}$ . Using the first  $2n$  samples, our goal is to compute a prediction set  $\widehat{C}(x)$  for each  $x \in \mathcal{X}$ . We will study the following properties for the prediction set.

1. *Marginal Coverage:*

$$\mathbb{P} \left( Y_{2n+1} \in \widehat{C}(X_{2n+1}) \right) \geq 1 - \alpha,$$

where the probability  $\mathbb{P}$  is jointly over all  $2n + 1$  pairs of observations.

## 2. Conditional Coverage:

$$\mathbb{P}\left(Y_{2n+1} \in \widehat{C}(X_{2n+1}) \mid X_{2n+1}\right) \geq 1 - \alpha, \quad (3.10)$$

with high probability.

It is well known that the conditional coverage property cannot be achieved without additional assumptions on  $P$  [Vovk, 2012, Lei and Wasserman, 2014, Foygel Barber, Candes, Ramdas, and Tibshirani, 2021]. Therefore, some form of relaxation of (3.10) is necessary.

In addition to the coverage properties listed above, we will also extend the notion of restricted volume optimality (VO) from the unsupervised setting to the supervised setting. Define the conditional CDF by

$$F(y \mid x) = \mathbb{P}(Y_{2n+1} \leq y \mid X_{2n+1} = x).$$

The conditional restricted optimal volume is given by

$$\begin{aligned} & \text{OPT}_k(F(\cdot \mid x), 1 - \alpha) \\ &= \inf \left\{ \text{vol}(C) : \int_C dF(\cdot \mid x) \geq 1 - \alpha, C \in \mathcal{C}_k \right\}. \end{aligned}$$

With this definition, we can list the following volume requirement.

## 3. Conditional Restricted Volume Optimality:

$$\begin{aligned} & \text{vol}(\widehat{C}(X_{2n+1})) \\ & \leq \text{OPT}_k(F(\cdot \mid X_{2n+1}), 1 - \alpha + \varepsilon), \end{aligned} \quad (3.11)$$

with high probability, for some  $\varepsilon \in (0, \alpha)$ .

Similar to the conditional coverage property (3.10), the conditional restricted volume optimality (3.11) is only required for a typical value of the design point. We will show that based on an extension of *distributional conformal prediction* [Chernozhukov, Wüthrich, and Zhu, 2021], these two properties can be achieved under the same assumption.

### 3.3.2 Distributional Conformal Prediction

Conformal prediction based on estimating the conditional CDF has been considered independently by [Izbicki, Shimizu, and Stern, 2020a, Chernozhukov, Wüthrich, and Zhu, 2021]. We will briefly review the version by Chernozhukov, Wüthrich, and Zhu [2021], and then extend it to serve our purpose. Suppose  $\widehat{F}(y \mid x)$  is an estimator of the conditional CDF, which is computed from the first half of the data  $(X_1, Y_1), \dots, (X_n, Y_n)$ . The prediction set proposed by Chernozhukov et al. [2021] is

$$\widehat{C}_{\text{DCP}}(X_{2n+1}) = \left\{ y \in \mathbb{R} : \left| \widehat{F}(y \mid X_{2n+1}) - \frac{1}{2} \right| \leq \widehat{t} \right\},$$

where  $\widehat{t}$  is an appropriate quantile of

$$\left\{ \left| \widehat{F}(Y_{n+1} \mid X_{n+1}) - \frac{1}{2} \right|, \dots, \left| \widehat{F}(Y_{2n} \mid X_{2n}) - \frac{1}{2} \right| \right\}.$$

Since  $\widehat{C}_{\text{DCP}}(X_{2n+1})$  is in the form of split conformal prediction, the marginal coverage property is automatically satisfied. When  $\widehat{F}(y \mid x)$  is close to  $F(y \mid x)$  in some appropriate sense, it was proved by Chernozhukov et al. [2021] that asymptotic conditional coverage also holds. However, in general,  $\widehat{C}_{\text{DCP}}(X_{2n+1})$  is not optimal in terms of its volume. A modification was also proposed in Chernozhukov et al. [2021] to achieve volume optimality within the class of intervals. Though not explicitly stated in Chernozhukov et al. [2021], we believe that the DCP procedure essentially achieves (3.11) for  $k = 1$ . Our goal is to achieve the conditional



restricted volume optimality for a general  $k$  by combining the ideas of DCP and dynamic programming (DP).

### 3.3.3 DCP meets DP

To achieve (3.11) for a general  $k$ , we will modify the DCP procedure by considering a different conformity score that generalizes (3.9) to the supervised setting. Recall that  $\widehat{F}(y \mid x)$  is an estimator of the conditional CDF, and it is computed from the first half of the data  $(X_1, Y_1), \dots, (X_n, Y_n)$ . Our first step is to construct a nested system for each  $x \in \mathcal{X}$ . To be specific, for each  $x \in \mathcal{X}$ , we will construct a collection of sets  $\{S_j(x)\}_{j \in [m]}$  based on the function  $\widehat{F}(\cdot \mid x)$ . The requirement of the nested system is summarized as the following assumption.

**Assumption 3.3.1.** *The sets  $S_1(x) \subset \dots \subset S_m(x) \subset \mathbb{R}$  are measurable with respect to the  $\sigma$ -field generated by  $\widehat{F}(\cdot \mid x)$ . Moreover, for some positive integer  $k$ , some  $\alpha \in (0, 1)$  and some  $\delta, \gamma$  such that  $3\delta + \gamma + n^{-1} \leq \alpha$ , we have*

1.  $\int_{S_j(x)} d\widehat{F}(\cdot \mid x) = \frac{j}{m}$  and  $S_j \in \mathcal{C}_k$  for all  $j \in [m]$ .

2. There exists some  $j^* \in [m]$ , such that

$$\int_{S_{j^*}(x)} d\widehat{F}(\cdot \mid x) \geq 1 - \alpha + n^{-1} + 3\delta$$

$$\text{and } \text{vol}(S_{j^*}) \leq \text{OPT}_k(\widehat{F}(\cdot \mid x), 1 - \alpha + \frac{1}{n} + 3\delta + \gamma).$$

The construction of nested systems satisfying Assumption 3.3.1 is similar to that in the unsupervised setting. That is, one can apply dynamic programming (Algorithm 3.1) and obtain  $S_{j^*}(x)$ , and the rest of the sets can be constructed via the greedy expansion/contraction procedure described in Section 3.4.1 to satisfy  $\int_{S_j(x)} d\widehat{F}(\cdot \mid x) = \frac{j}{m}$ . The main difference here

is that [Algorithm 3.1](#) is directly applied to the data in the unsupervised setting, while we only have access to  $\widehat{F}(\cdot | x)$  in the supervised setting. This issue can be easily addressed by computing quantiles  $Y_1(x), \dots, Y_L(x)$  from  $\widehat{F}(\cdot | x)$  on a grid, and then apply [Algorithm 3.1](#) with  $Y_1(x), \dots, Y_L(x)$  as input. Indeed, since the distance between  $\widetilde{F}(\cdot | x)$  and  $\widehat{F}(\cdot | x)$  can be controlled by the size of the grid with  $\widetilde{F}(y | x) = \frac{1}{L} \sum_{l=1}^L \mathbb{I}\{Y_l(x) \leq y\}$ , [Assumption 3.3.2](#), which will be stated later in [Section 3.3.4](#), is also satisfied by  $\widetilde{F}(y | x)$  (with a slightly larger value of  $\delta$ ) by the triangle inequality.

The computational cost of constructing  $\{S_j(x)\}_{j \in [m]}$  for a single  $x \in \mathcal{X}$  is  $O(L^3 k / \gamma)$ . Note that there is no need to repeat the construction for each individual  $x \in \mathcal{X}$ . Since the split conformal framework only requires evaluating the conformity score at  $(X_{n+1}, Y_{n+1}), \dots, (X_{2n}, Y_{2n}), (X_{2n+1}, y)$ , it is sufficient to compute  $\{S_j(X_i)\}_{j \in [m]}$  for  $i = n + 1, \dots, 2n + 1$ , which leads to the total computational cost  $O(nL^3 k / \gamma)$ .

With nested systems satisfying [Assumption 3.3.1](#), the conformity score in the supervised setting is defined as

$$q(y, x) = \sum_{j=1}^m \mathbb{I}\{y \in S_j(x)\}.$$

Let  $q_1 \leq \dots \leq q_n$  be the order statistics computed from the set

$$\{q(X_{n+1}, Y_{n+1}), \dots, q(X_{2n}, Y_{2n})\}.$$

The prediction set for  $Y_{2n+1}$  is constructed as

$$\widehat{C}_{\text{DCP-DP}}(X_{2n+1}) = \{y : q(y, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}\}.$$

### 3.3.4 Theoretical Guarantees

We will show in this section that  $\hat{C}_{\text{DCP-DP}}(X_{2n+1})$  satisfies marginal coverage. Moreover, when  $\hat{F}(y | x)$  is close to  $F(y | x)$  in some appropriate sense, it also satisfies approximate conditional coverage and conditional restricted volume optimality. Given two CDFs  $\hat{F}$  and  $F$ , we define the  $(k, \infty)$  norm of the difference by

$$\|\hat{F} - F\|_{k, \infty} = \sup_{C \in \mathcal{C}_k} \left| \int_C d\hat{F} - \int_C dF \right|.$$

**Assumption 3.3.2.** *The estimated conditional CDF  $\hat{F}(y | x)$  satisfies*

$$\mathbb{P} \left( \|\hat{F}(\cdot | X_{2n+1}) - F(\cdot | X_{2n+1})\|_{k, \infty} \leq \delta \right) \geq 1 - \delta,$$

where  $\delta$  takes the same value as the one in [Assumption 3.3.1](#).

The theoretical properties of  $\hat{C}_{\text{DCP-DP}}(X_{2n+1})$  are given by the theorem below.

**Theorem 3.3.3.** *Consider i.i.d. observations  $(X_1, Y_1), \dots, (X_{2n}, Y_{2n}), (X_{2n+1}, Y_{2n+1})$  generated by some distribution  $P$  on  $\mathcal{X} \times \mathbb{R}$ . The conformal prediction set  $\hat{C}_{\text{DCP-DP}}(X_{2n+1})$  is computed from nested systems  $\{S_j(\cdot)\}_{j \in [m]}$  and  $\hat{F}(\cdot | \cdot)$  satisfying [Assumption 3.3.1](#) and [Assumption 3.3.2](#). Suppose the parameter  $\delta$  in the two assumptions satisfies  $\delta^2 \geq \frac{\log(2\sqrt{n})}{2n}$ . Then the following properties hold.*

1. *Marginal coverage,*

$$\mathbb{P} \left( Y_{2n+1} \in \hat{C}_{\text{DCP-DP}}(X_{2n+1}) \right) \geq 1 - \alpha.$$

2. *Approximate conditional coverage,*

$$\mathbb{P} \left( Y_{2n+1} \in \hat{C}_{\text{DCP-DP}}(X_{2n+1}) | X_{2n+1} \right) \geq 1 - \alpha - 3\delta,$$

with probability at least  $1 - \delta$ .

3. *Conditional restricted volume optimality,*

$$\begin{aligned} & \text{vol} \left( \widehat{C}_{\text{DCP-DP}}(X_{2n+1}) \right) \\ & \leq \text{OPT}_k \left( F(\cdot \mid X_{2n+1}), 1 - \alpha + \frac{1}{n} + 4\delta + \gamma \right), \end{aligned}$$

with probability at least  $1 - 2\delta$ .

Note that the marginal coverage does not depend on  $\delta$ , which reflects the estimation error of the conditional CDF in [Assumption 3.3.2](#). Therefore, the marginal coverage guarantee always holds and does not rely on [Assumption 3.3.2](#). [Theorem 3.3.3](#) can be regarded as a generalization of [Theorem 3.2.6](#). Indeed, when  $F(\cdot \mid x)$  does not depend on  $x$  and  $\widehat{F}(\cdot \mid x)$  is defined as the empirical CDF of  $Y_1, \dots, Y_n$ , [Theorem 3.3.3](#) recovers [Theorem 3.2.6](#). Moreover, since the volume optimality is over all sets that are unions of  $k$  intervals, it also covers the length optimality of intervals considered by [Chernozhukov, Wüthrich, and Zhu \[2021\]](#). The case  $k \geq 2$  will be important if the conditional density of  $Y$  given  $X$  has multiple modes; Gaussian mixture is a leading example.

## 3.4 Numerical Experiments

We complement our theoretical guarantees with an evaluation of our methods for both the unsupervised setting of [Section 3.2](#) and the supervised setting of [Section 3.3](#).

### 3.4.1 Construction of Nested Systems

We first describe a procedure that generates a nested system  $\{S_j\}_{j \in [m]}$  that satisfies [Assumption 3.2.5](#). The construction involves the following steps:

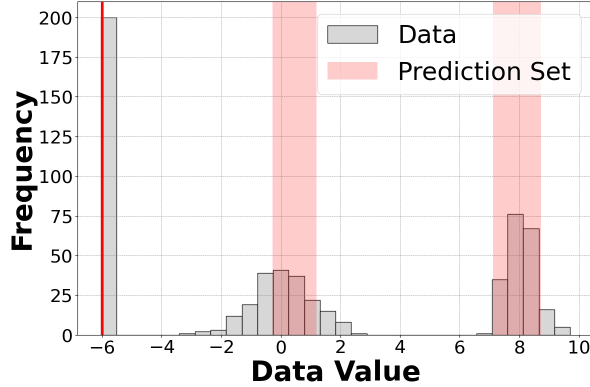
1. *Generate  $S_{j^*}$  by Dynamic Programming.* For  $j^* = \lceil (1 - \alpha + n^{-1} + 3\delta)m \rceil$ , we generate  $S_{j^*}$  by applying [Algorithm 3.1](#) with coverage level  $1 - j^*/m$  and slack  $\gamma = 1/m$ .
2. *Generate  $S_{j^*+1}, \dots, S_m$  by Greedy Expansion.* For each  $j > j^*$ , we iteratively identify the closest uncovered data point to the boundary of the current  $k$  intervals and expand the nearest interval to cover it. Once the intervals cover  $\lceil jn/m \rceil$  data points, we define the union as  $S_j$  and move on to the construction of  $S_{j+1}$ .
3. *Generate  $S_1, \dots, S_{j^*-1}$  by Greedy Contraction.* For each  $j < j^*$ , we iteratively remove a boundary point of the current  $k$  intervals that results in the maximum volume reduction. Once the intervals after contraction cover exactly  $\lceil jn/m \rceil$  data points, we define the union as  $S_j$  and move on to the construction of  $S_{j-1}$ .

In the supervised setting, the above procedure will be applied to quantiles  $Y_1(x), \dots, Y_L(x)$  computed from  $\hat{F}(\cdot \mid x)$  with  $L = m$  for all  $x \in \{X_{n+1}, \dots, X_{2n+1}\}$ .

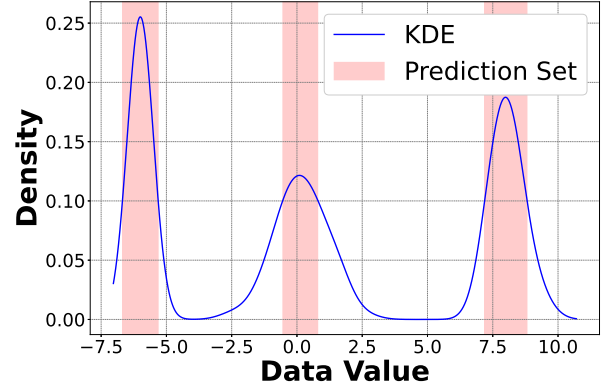
### 3.4.2 Comparison in Unsupervised Settings

The algorithm in [Section 3.2](#) is compared against the method based on kernel density estimation due to [Lei, Robins, and Wasserman \[2013\]](#) and evaluated on several different distributions. Though the original conformalized KDE was proposed in the full conformal framework, we will consider its split conformal version for a direct comparison. We believe the comparison between the full conformal versions of the two methods will lead to the same conclusion. For the conformalized DP method, the conformity score is constructed based on the nested system described in [Section 3.4.1](#) with  $m = 50$  and  $\delta = \sqrt{(k + \log n)/n}$ . The conformalized KDE is also in the form of (3.8), with the conformity score given by  $q_{\text{KDE}}(x) = \frac{1}{n\rho} \sum_{i=1}^n K\left(\frac{y - Y_i}{\rho}\right)$ , where  $K(\cdot)$  is the standard Gaussian kernel and  $\rho$  is the bandwidth parameter. Both methods involve a single tuning parameter,  $k$  for conformalized DP

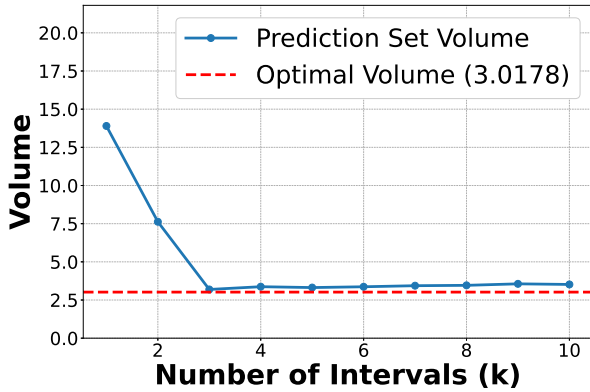
and  $\rho$  for conformalized KDE.



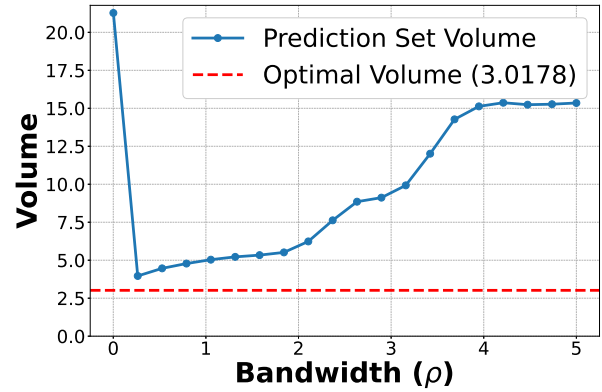
(a) The histogram of the dataset and the prediction set given by conformalized DP with  $k = 3$  intervals (The first interval is at  $[-6.03, -5.97]$ ). The volume of the prediction sets is 3.1438.



(b) The kernel density estimation with bandwidth  $\rho = 0.5$  and the prediction set given by conformalized KDE [Lei, Robins, and Wasserman, 2013]. The volume of the prediction sets is 4.4775.



(c) Volumes of prediction sets by conformalized DP is not sensitive to the choice of  $k$ .



(d) Volumes of prediction sets by conformalized KDE is highly sensitive to the choice of  $\rho$ .

Figure 3.1: Conformal prediction sets on the mixture of Gaussians data from  $P = \frac{1}{3}N(-6, 0.0001) + \frac{1}{3}N(0, 1) + \frac{1}{3}N(8, 0.25)$ . The coverage probability is 80%. The theoretically optimal volume is 3.0178.

Figure 3.1 summarizes the results using data generated from a mixture of Gaussians  $\frac{1}{3}N(-6, 0.0001) + \frac{1}{3}N(0, 1) + \frac{1}{3}N(8, 0.25)$ . Additional experiments on other distributions including standard Gaussian, censored Gaussian and ReLU-transformed Gaussian are presented in Appendix B.1.

### 3.4.3 Comparison in Supervised Settings

The algorithms for the supervised setting are compared against conformalized quantile regression (CQR) [Romano, Patterson, and Candès, 2019], distributional conformal prediction methods (DCP-QR and DCP-QR\*) [Chernozhukov, Wüthrich, and Zhu, 2021], and CD-Split and HPD-Split methods [Izbicki, Shimizu, and Stern, 2020b] against benchmark simulated datasets in Romano, Patterson, and Candès [2019], Izbicki, Shimizu, and Stern [2020a] (Figure 3.2 and Figure 3.3). The implementation details of all methods are given in Appendix B.1.

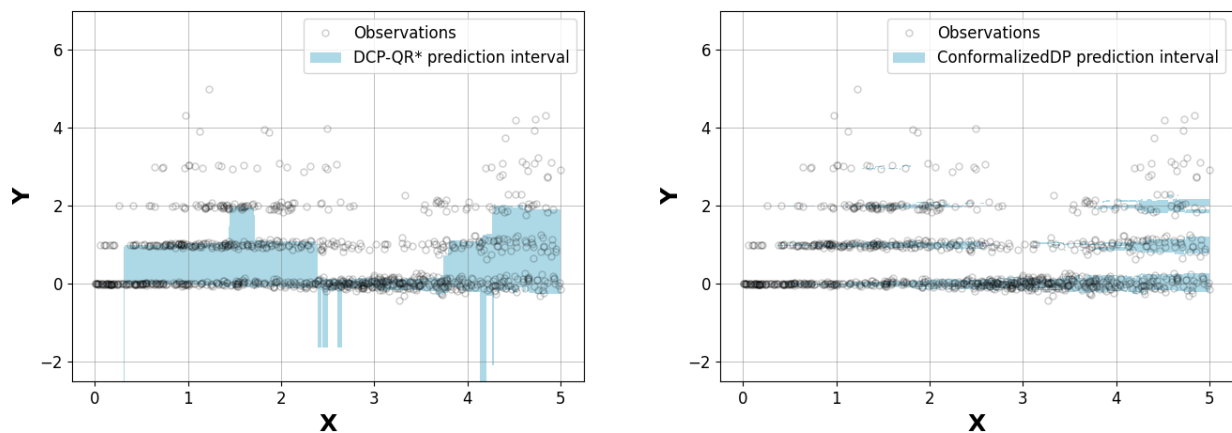


Figure 3.2: Results in the supervised setting on a synthetic data from [Romano, Patterson, and Candès, 2019] for target coverage 0.7. The left plot shows the output of DCP-QR\*, the state of the art method by Chernozhukov, Wüthrich, and Zhu [2021], which outputs prediction sets with average volume 1.29. The right plot shows the output of our method with  $k = 5$  intervals, which achieves a significantly improved average volume of 0.45.

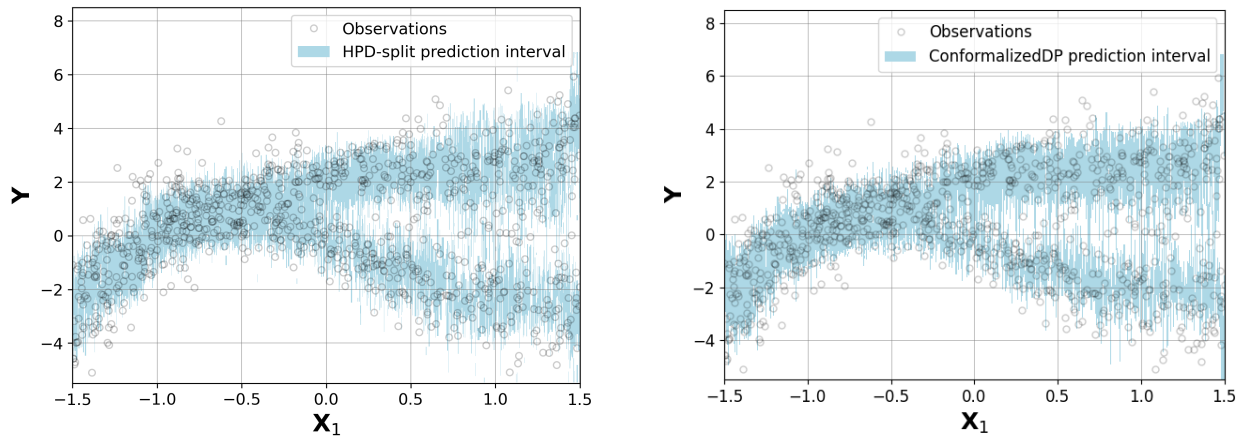


Figure 3.3: Results in the supervised setting on a synthetic data with 20 dimensional feature from [Izbicki, Shimizu, and Stern, 2020a] for target coverage 0.7. The left plot shows the output of HPD-Split method by Izbicki, Shimizu, and Stern [2020b], with average volume 3.60. The right plot shows the output of our method with  $k = 2$  intervals, which has an average volume 3.55.

Table 3.1: Comparison on simulated data from Romano, Patterson, and Candès [2019].

| Method                     | Average Volume | Empirical Coverage (%) |
|----------------------------|----------------|------------------------|
| CQR                        | 1.42           | 70.62                  |
| DCP-QR                     | 1.48           | 71.60                  |
| DCP-QR*                    | 1.29           | 71.06                  |
| CD-split                   | 1.83           | 69.94                  |
| HPD-split                  | 1.75           | 69.44                  |
| Conformalized-DP ( $k=1$ ) | 1.14           | 74.04                  |
| Conformalized-DP ( $k=5$ ) | <b>0.45</b>    | 72.36                  |



Table 3.2: Comparison on simulated data from [Izbicki, Shimizu, and Stern \[2020a\]](#).

| Method                     | Average Volume | Empirical Coverage (%) |
|----------------------------|----------------|------------------------|
| CQR                        | 4.10           | 71.54                  |
| DCP-QR                     | 4.04           | 70.85                  |
| DCP-QR*                    | 4.05           | 69.66                  |
| CD-split                   | 3.69           | 69.86                  |
| HPD-split                  | 3.60           | 69.64                  |
| Conformalized-DP ( $k=1$ ) | 4.00           | 68.98                  |
| Conformalized-DP ( $k=2$ ) | <b>3.55</b>    | 69.42                  |

As shown in [Table 3.1](#) and [Table 3.2](#), our method outperforms previous methods by outputting prediction sets that are unions of intervals. Among all other methods, the CD-split and HPD-split [[Izbicki, Shimizu, and Stern, 2020a](#)] are also able to produce unions of intervals. However, since these methods rely on consistent estimation of the conditional density, our conformalized DP still produces prediction sets with smaller volumes. The comparison is more pronounced on the first dataset (see [Figure 3.2](#) and [Table 3.1](#)), where it would be inappropriate to assume a smooth conditional density, but our method is based on conformalizing the estimation of conditional CDF, and thus works in much more general settings.

These experiments together demonstrate that our methods give a clear advantage over competitive procedures under these different conditions. More crucially, our methods also come with theoretical guarantees for both coverage and volume-optimality with respect to unions of  $k$  intervals.

Finally, we remark that our focus on the family of structured prediction sets conceptually differs from the predominant approach of designing new conformity score functions in existing

approaches. The prediction sets that are output e.g., unions of intervals, are often more interpretable. Apart from being more interpretable, the shift in focus to structured prediction sets, which is in the same spirit of [Gupta, Kuchibhotla, and Ramdas \[2022\]](#), is what allows us to overcome any sample efficiency concerns, to obtain *distribution-free volume-optimality*, through computationally efficient algorithms.

### 3.5 Chapter Acknowledgements

This chapter is based on joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan. The research started as part of the IDEAL special program on Reliable and Robust Data Science. We thank Rina Barber and Jing Lei for helpful discussions.

## Chapter 4

# Competing with High-dimensional Balls

This chapter is based on *Computing High-dimensional Confidence Sets for Arbitrary Distributions*, which is joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan, and appeared at the Conference on Learning Theory (COLT) 2025 [[Gao, Shan, Srinivas, and Vijayaraghavan, 2025a](#)].

In this chapter, we develop computationally efficient algorithms to find prediction sets that satisfy approximate  $\mathcal{C}$ -restricted volume optimality ([Problem 2.2.1](#)), for  $\mathcal{C}$  the family of Euclidean ( $\ell_2$ ) balls in  $\mathbb{R}^d$ . Our results also naturally extend to the family of unions of Euclidean balls in  $\mathbb{R}^d$ .

## 4.1 Introduction

**The family of Euclidean Balls.** Consider the setting where  $\mathcal{C}$  is the set of  $\ell_2$  balls (this has VC-dimension at most  $d$ ). This is a natural generalization of intervals in higher dimensions, and is motivated by applications [see e.g., Wang, Gao, Yin, Zhou, and Blei, 2023]. It is NP-hard to find the smallest set in  $\mathcal{C}$  that covers a  $\delta$  fraction of points from an arbitrary set [Shenmaier, 2015].<sup>1</sup> The famous work of Badoiu, Har-Peled, and Indyk [2002] on coresets for minimum volume enclosing balls [see also, Agarwal, Har Peled, and Varadarajan, 2004, Ding, 2020] gives a proper learning algorithm that runs in polynomial time and is  $\Gamma = 1 + O(1/\log d)$ -competitive i.e., a  $\exp(O(d/\log d))$  factor approximation in volume. To the best of our knowledge, these works using coresets [Badoiu, Har-Peled, and Indyk, 2002, Ding, 2020] remain the state-of-the-art guarantee for learning confidence sets competitive with Euclidean balls in  $d$  dimensions.

### 4.1.1 Results in this Chapter

Our main algorithmic result is an algorithm that is  $\Gamma \approx 1 + d^{-1/2}$  competitive against Euclidean balls.

**Theorem 4.1.1** (Learning confidence sets competitive with Euclidean balls). *There is a polynomial time algorithm that for any target coverage  $\delta \in (0, 1)$  and coverage slack  $\gamma \in (0, 1)$ , when given  $n = \Omega(d^2/\gamma^2)$  samples drawn i.i.d. from an arbitrary distribution  $\mathcal{D}$ , finds with high probability a set  $S \subset \mathbb{R}^d$  that is  $\Gamma = \exp(O_{\gamma,\delta}(d^{-1/2+o(1)}))$  competitive, i.e.,*

$$\mathbf{P}_{y \sim \mathcal{D}} [y \in S] \geq \delta,$$

---

<sup>1</sup>On the other hand, finding the minimum volume ball that encloses all the given  $n$  points is polynomial time solvable.

and

$$\text{vol}(S)^{1/d} \leq \text{vol}(B^*)^{1/d} (1 + O(d^{-1/2+o(1)}(\gamma\delta)^{-1})).$$

where  $B^*$  is the minimum volume ball that achieves  $\geq \delta + \gamma + O(\sqrt{d^2/n})$  coverage over  $\mathcal{D}$ .

*Remark 4.1.2.* Note that the guarantee above is stated slightly differently than in our original definition of conformal prediction with approximate restricted volume optimality, [Problem 2.2.1](#). First, we use  $\delta$  instead of  $1 - \alpha$  as our coverage parameter. We do this to emphasize that the methods in this chapter extend to the settings where  $\delta$  can be less than  $1/2$ . We also state the approximation factor in terms of  $\Gamma$ , which is the approximation in  $\text{vol}^{1/d}$  or “radius.” We find this a natural way to state volume approximations in terms of  $d$ -dimensional volume.

Please see [Section 4.4](#) for more details. To compare against existing methods, consider the setting when  $\delta, \gamma$  are constants. A straightforward solution that considers balls around the sample points that covers at least  $\delta$  fraction of the data points achieves a factor of  $\Gamma = 2$  (due to the triangle inequality).<sup>2</sup> Algorithms based on the powerful technique of coresets for minimum volume enclosing balls due to [Badoiu, Har-Peled, and Indyk \[2002\]](#) gives  $\Gamma = 1 + \tilde{O}(1/\log d)$ -competitive. In terms of the volume, the  $\Gamma \approx 1 + d^{-1/2}$  factor corresponds to volume approximation of

$$\Gamma^d = (1 + O_{\delta,\gamma}(d^{-1/2+o(1)}))^d = \exp(O_{\gamma,\delta}(d^{1/2+o(1)})),$$

which corresponds to a substantial improvement from the  $\exp(O_{\delta,\gamma}(d/\log d))$  volume approximation factor in existing work [[Badoiu, Har-Peled, and Indyk, 2002](#), [Ding, 2020](#)].

---

<sup>2</sup>In fact this strategy achieves a  $\Gamma = 2$  factor approximation (with  $\text{vol}^{1/d}$ ) for any set  $\mathcal{C}$  that consists of scalings and translations of a fixed convex shape. However, it is NP-hard to obtain a  $(2 - o(1))$  factor approximation even for the  $\ell_\infty$  ball [[Shenmaier, 2015](#)].

**Union of Balls  $\mathcal{C}_k$ .** We next consider the more general concept class  $\mathcal{C}_k$  which is the unions of  $k$  Euclidean balls. This family has VC-dimension of at most  $kd$ . We obtain a similar guarantee for unions of  $k = O(1)$  balls, by recursively applying the algorithm from [Theorem 4.1.1](#).

**Theorem 4.1.3** (Union of Balls). *Let  $\delta \in (0, 1), \gamma \in (0, 1), k \in \mathbb{N}$  be any constants. There is a polynomial time algorithm that for target coverage  $\delta \in (0, 1)$  and coverage slack  $\gamma \in (0, 1)$  when given  $n = \Omega(kd^2/\gamma^2)$  samples drawn i.i.d. from an arbitrary distribution  $\mathcal{D}$ , finds with high probability a set  $S \subset \mathbb{R}^d$  that is  $\Gamma = (1 + O_{\gamma,\delta}(d^{-1/2+o(1)}))$  competitive; more precisely, it satisfies  $\mathbf{P}_{y \sim \mathcal{D}}[y \in S] \geq \delta$ , and*

$$\text{vol}(S)^{1/d} \leq \text{vol}(C_k^*)^{1/d} (1 + O_{k,\delta}(d^{-1/2+o(1)})) \cdot \left( \frac{O(\log(k/\gamma))}{\gamma} \right)^{1/d}$$

where  $C_k^*$  is the minimum volume union of  $k$ -balls that achieves at least  $\delta + \gamma + O(\sqrt{kd^2/n})$  coverage over  $\mathcal{D}$ .

See [Corollary 4.5.2](#) for details. Our algorithms are surprisingly simple and conceptually different from existing algorithms based on coresets. It is based on a new connection to robust estimation that follows from a structural statement about the mean and the center of a large fraction of points in the optimal ball with the desired coverage. We give a detailed description of the ideas in the technical overview.

### 4.1.2 Proper Learning: Separation Result and Algorithms

The algorithm in [Theorem 4.1.1](#) outputs a set  $S \subset \mathbb{R}^d$  of small volume achieving the desired coverage, when there is a small ball achieving the required coverage. The set  $S$  may not be a ball, and may in general be an ellipsoid. Similarly the algorithm in [Theorem 4.1.3](#) outputs a union of  $O(k/\gamma)$  ellipsoids. These correspond to “improper” learners for confidence

sets. In contrast, the best *proper* learning algorithms achieve a worse competitive ratio of  $\Gamma = 1 + O(1/\log d)$ . The following theorem shows that such a loss is unavoidable assuming  $P \neq NP$ , unless we are willing to make additional assumptions on  $\mathcal{D}$ .

**Theorem 4.1.4** (NP-hardness of Proper Learning for Balls). *For any constant  $\varepsilon > 0$ , assuming  $P \neq NP$  there is no polynomial time learning algorithm that given samples from an arbitrary distribution over  $\mathbb{R}^d$  that outputs balls as confidence sets that are  $\Gamma \leq (1 + \frac{1}{d^\varepsilon})$  competitive with balls.*

See [Theorem 4.6.1](#) for a more formal statement. Note that the above result is a worst-case hardness result, and reflects the arbitrary nature of the distribution. This is reminiscent of NP-hardness of proper learning in supervised learning [see e.g., [Feldman, Gopalan, Khot, and Ponnuswami, 2009](#)]. The above theorem follows from a simple reduction that amplifies the existing NP-hardness result due to [Shenmaier \[2015\]](#). In [Theorem 4.6.3](#), we also show computational intractability (assuming the Small Set Expansion hypothesis of [Raghavendra and Steurer \[2010\]](#)) that shows a similar quantitative hardness, even when the coverage can be smaller by an arbitrary constant factor. Combined with our algorithmic result in [Theorem 4.1.1](#), this shows a formal separation between proper and improper learning for a natural geometric learning problem.

**Proper learning algorithms.** When we are required to output a confidence set from  $\mathcal{C}$ , the algorithm in [Theorem 4.1.1](#) can also be used to output a ball with a worse competitive ratio (which is expected due to [Theorem 4.1.4](#)). When we make no assumptions about the distribution  $\mathcal{D}$ , our algorithm can output a ball that is  $\Gamma = 1 + \tilde{O}(1/\log d)$  competitive. See [Theorem 4.3.1](#) in [Section 4.3](#) for a formal statement of the theorem. This guarantee recovers the guarantee from the coresnet-based algorithm which achieves  $\Gamma = 1 + \tilde{O}(1/\log d)$  [[Badoiu, Har-Peled, and Indyk, 2002](#)]. In the worst-case instances of the algorithm (and the hard

instances in [Theorem 4.1.4](#)), the points sampled from  $\mathcal{D}$  that lie inside the optimal ball  $B^*$  effectively lie on a lower-dimensional subspace.

On the other hand, our algorithm can give a significantly stronger guarantee when the points inside  $B^*$  are approximately isotropic. In the following theorem, we consider the setting where we have  $n$  points  $Y$  drawn from  $\mathcal{D}$ , and the guarantee depends on the covariance of the sampled points within the optimal ball  $B^*$ .

**Theorem 4.1.5** (Bounded variance implies better bounds). *Let  $Y \subseteq \mathbb{R}^d$  be a set of  $n$  points, and  $\delta, \gamma \in (0, 1)$ , such that there is an unknown ball  $B^* = B(c^*, R^*)$  with  $Y^* = B^* \cap Y$  satisfying  $|Y^*| \geq \delta|Y|$ , and  $Y^*$  is  $\beta(R^*)^2/d$ -isotropic for some  $\beta \geq 0$ . That is, if  $\mu^*$  is the mean of the points  $Y^*$ , then*

$$\Sigma_{Y^*} = \frac{1}{|Y^*|} \sum_{y \in Y^*} (y - \mu^*)(y - \mu^*)^\top \preceq \beta \frac{(R^*)^2}{d} I.$$

*Then we can find a ball  $\hat{B} = B(\hat{c}, \hat{R})$  such that  $|Y \cap \hat{B}| \geq (1 - \gamma)|Y^*|$ , and*

$$\text{vol}(\hat{B})^{1/d} \leq \text{vol}(B^*)^{1/d} \sqrt{1 + O\left(\frac{\beta}{\gamma \delta d}\right)},$$

*in polynomial time.*

In the above [Theorem 4.1.5](#),  $\Sigma_{Y^*}$  is a PSD matrix of dimension  $d \times d$  with  $\text{tr}(\Sigma_{Y^*}) \leq (R^*)^2$ , since  $Y^* \subseteq B^*$ . The isotropicity condition requires the maximum eigenvalue  $\lambda_{\max}(\Sigma_{Y^*}) \leq \beta(R^*)^2/d$  for some  $\beta > 0$ . Thus every  $Y^*$  satisfies the isotropic condition with  $\beta \leq d$ . When the points of  $Y^*$  are very spread out, this corresponds to a setting when  $\beta \leq 1$  (all the eigenvalues are equal). In this case, the theorem guarantees to find a ball with  $\Gamma = 1 + O(d^{-1})$ . In other words, it finds a ball whose volume is within  $O_{\delta, \gamma}(1)$  factor of the optimum, which is significantly stronger. Finally, the ball  $\hat{B}$  output by the algorithm also



has coverage  $(1 - \gamma)\delta - O(\sqrt{d/n})$  by standard concentration arguments [see e.g., [Devroye and Lugosi, 2001](#)] since the VC-dimension of  $d$ -dimensional Euclidean balls is  $d + 1$ .

All of our algorithmic results are distribution-free i.e., they do not make any assumptions on the distribution  $\mathcal{D}$ . While it may be impossible to learn the distribution or even estimate the density at a given point (these tasks often incur an exponential dependence on the dimension  $d$  even for smooth distributions), our results show that one can obtain polynomial time algorithms that find approximately optimal dense sets from  $\mathcal{C}$  covering at least  $\delta$  mass.

### 4.1.3 Application to Conformal Prediction

As a direct application of our result we get the following algorithm for conformal prediction in high dimensions. In conformal prediction, an algorithm is given training examples  $Y_1, \dots, Y_n$  in some set  $\mathcal{Y}$ , and a miscoverage rate  $\alpha$ , and must output a set  $\hat{C}$ , where

$$\mathbb{P}[Y_{n+1} \in \hat{C}] \geq 1 - \alpha,$$

for some unseen training example  $Y_{n+1}$ , assuming only that the training and test examples  $Y_1, \dots, Y_{n+1}$  are *exchangeable*.<sup>3</sup> There is a fairly simple way to “conformalize” our result to achieve this goal, resulting in the following algorithm that always outputs a valid conformal set when the data is exchangeable, and is additionally approximately volume optimal when the data is drawn i.i.d. from an (unknown) distribution  $\mathcal{D}$ .

**Theorem 4.1.6** (Conformal Prediction with Approximate Volume Optimality). *We have an algorithm for conformal prediction over examples from  $\mathcal{Y} = \mathbb{R}^d$ , that achieves approximate restricted volume optimality ([Problem 2.2.1](#)) with respect to the set  $\mathcal{B}$  of Euclidean balls. That is, we have an algorithm that, given training examples  $Y_1, \dots, Y_n \in \mathcal{Y}$  that are*

---

<sup>3</sup>That is,  $\mathbb{P}[Y_1 = y_1, \dots, Y_{n+1} = y_{n+1}] = \mathbb{P}[Y_1 = y_{\pi(1)}, \dots, Y_{n+1} = y_{\pi(n+1)}]$ , for all  $y_1, \dots, y_{n+1} \in \mathcal{Y}$  and permutations  $\pi : [n+1] \rightarrow [n+1]$ .

exchangeable with an unseen test example  $Y_{n+1} \in \mathcal{Y}$ , miscoverage rate  $0 < \alpha < 1$ , and coverage slack factor  $0 \leq \gamma \leq 1$ , outputs a set  $\hat{C}$  (not necessarily in  $\mathcal{B}$ ), such that

(a)  $\hat{C}$  achieves the target coverage on  $Y_{n+1}$ ,

$$\mathbb{P}[Y_{n+1} \in \hat{C}] \geq 1 - \alpha. \quad (4.1)$$

(b) if  $n = \Omega(d^2/\gamma^2)$ , then  $\hat{C}$  achieves approximate volume optimality with respect to Euclidean balls,

$$\text{vol}(\hat{C})^{1/d} \leq (1 + O_{\gamma,\delta}(d^{-1/2+o(1)})) \text{vol}(C^*)^{1/d},$$

where

$$C^* = \underset{C \in \mathcal{B}}{\text{argmin}} \text{vol}(C) \quad \text{s.t.} \quad \mathbb{P}[Y_{n+1} \in C] \geq 1 - \alpha + \gamma.$$

We provide an in depth discussion of this application in [Section 4.7](#). See also [Section 5.5](#) for more applications in high-dimensional statistics. We now review related literature and other methods that have been explored in literature, followed by an overview of the algorithm and techniques.

#### 4.1.4 Prior and Related Algorithmic Approaches

**Algorithms based on Coresets.** Coresets are a powerful algorithmic primitive for geometric problems involving an input set of data points [[Agarwal, Har Peled, and Varadarajan, 2004](#)]. A coreset for a problem is a small summation of the data (e.g., a subset of the points) such that solving the problem on the summation gives an approximate solution to the whole instance. Badiou, Har-Peled and Indyk proved a surprising result about coresets for the problem of minimum volume enclosing ball for a set of points – they showed that there is a

coreset of size  $1/\varepsilon$  (that is independent of the dimension) whose minimum volume enclosing ball approximates the minimum volume ball enclosing all the points up to a  $(1 + \varepsilon)$  factor in radius [Badoiu, Har-Peled, and Indyk, 2002].<sup>4</sup> Badoiu, Har-Peled, and Indyk [2002] show how to find a ball enclosing  $\delta - \gamma$  fraction of the given points in polynomial time whose radius is within a factor  $\Gamma = \left(1 + O_\gamma\left(\frac{(\log \log d)^2}{\log d}\right)\right)$  [see Theorem 4.2 of Badoiu, Har-Peled, and Indyk, 2002]. The subsequent work of Ding [2020] also uses coresets and seems to obtain a guarantee of  $\Gamma = \left(1 + O_\gamma\left(\frac{1}{\log d}\right)\right)$  [see (24) following Corollary 11 in Ding, 2020], though this is not stated explicitly. This corresponds to an approximation factor of  $\Gamma^d = \exp(\tilde{O}(d/\log d))$  for the volume. We are not aware of any other work that get a better competitive ratio  $\Gamma$  for balls.

**Comparison to robust mean estimation and covariance estimation.** There is a rich literature for developing robust algorithms to perform basic statistical estimation tasks in high-dimensions when  $1 - \delta$  fraction of the points are contaminated [see e.g., Huber, 2004, Diakonikolas and Kane, 2023]. This is usually studied in the context of parameter estimation problems like mean estimation or covariance estimation where a  $\delta$  fraction of points are *inliers* that are assumed to be drawn from a distribution with a location (mean) or covariance, while the rest of the points are arbitrary *outliers*. The inlier distribution is assumed to satisfy some nice property like bounded covariance and/or moments, or typically from a parametric family like Gaussians. Various contamination models have been considered. In the famous Huber contamination model and the stronger adversarial contamination model, a  $\delta = 1 - \eta > 1/2$  fraction of the points are inliers, while a  $\eta = 1 - \delta < 1/2$  fraction of the points are outliers [Huber, 1964, Diakonikolas and Kane, 2023]. The setting when  $\delta < 1/2$  has also been studied as list-decodable robust estimation [Charikar, Steinhardt, and Valiant, 2017]. Most relevant to us are the works in algorithmic robust statistics including the important

---

<sup>4</sup>The coreset size of  $\lceil 1/\varepsilon \rceil$  is also known to be tight [Agarwal, Har Peled, and Varadarajan, 2004].

works of [Diakonikolas, Kamath, Kane, Li, Moitra, and Stewart \[2016\]](#), [Lai, Rao, and Vempala \[2016\]](#). which gave polynomial time algorithms for robust mean and covariance estimation in high dimensions. Over the past decade there have been several polynomial time algorithms proposed based on outlier removal, sophisticated convex relaxations including sum-of-squares and many other techniques [please see [Diakonikolas and Kane, 2023](#), for an excellent book on the topic].

However, the robust estimation literature usually assumes various distributional assumptions – for mean estimation, they assume at the very least that the inlier distribution has bounded variance in every direction; while for covariance estimation one needs various fourth moment conditions, or other distributional assumptions like Gaussianity. Such assumptions are necessary to define a certifiable property of the inlier distribution in parameter estimation tasks.

In our setting, our main task is not one of parameter estimation. More importantly, the distribution of points is completely arbitrary i.e., we cannot make any assumptions on the distribution of both the inliers and the outliers. But in some settings we use powerful algorithms for list-decodable mean estimation [[Charikar, Steinhardt, and Valiant, 2017](#)] as a black-box subroutine once we reduce the task to mean estimation for bounded covariance distributions. Finally, algorithms for our problem can also be used to design robust estimators for various statistical tasks like mode estimation or robust support estimation, as detailed in [Section 5.5](#).

## 4.2 Technical Overview

Our goal is, given a set  $Y$  of samples from  $\mathcal{D}$ , to find a set  $S$  that captures  $(1 - \gamma)\delta|Y|$  points of  $Y$ , and has volume comparable to the minimum volume ball  $B^*$  that captures a subset of

points  $Y^* \subseteq Y$ , with  $|Y^*| \geq \delta|Y|$ . If we could estimate the center of  $B^*$  well, then we could find  $B^*$  approximately by guessing an appropriate radius. We can focus on the worst-case problem over the samples  $Y$  because we will choose our output set  $S$  to be an ellipsoid. Since the class of ellipsoids in  $d$  dimensions has bounded VC-dimension, the empirical coverage of all ellipsoids enjoys uniform convergence. This means that the coverage of *all* ellipsoids will simultaneously generalize from the empirical samples to the population setting, so it suffices to find an ellipsoid that approximately minimizes volume subject to a coverage constraint on the empirical samples.

Intuitively, one could interpret our problem as trying to recover the *center* of the points  $Y^* \subseteq Y, |Y^*| \geq \delta|Y|$  in a way that is robust to adversarial contamination, i.e., we only have access to  $Y$ . However, the center of the ball could depend on only a few points, i.e., a sphere in  $d$  dimensions can be defined by  $d + 1$  points.<sup>5</sup> Thus, it appears that the center is not a quantity that can be robust to outliers. This is in contrast to other quantities, such as the mean or median, that inherently depend on the whole dataset, and are not largely influenced by a few points.

In fact, the goal of recovering the center of  $B^*$  is too strong. Properly learning the minimum volume ball  $B^*$  within an approximation factor  $\Gamma = 1 + 1/d^\varepsilon$ , for any constant  $\varepsilon > 0$  is NP-hard. So we should not expect such a strategy to work in the worst case.

However, it turns out that for certain non-worst-case instances, it is indeed possible to recover  $B^*$  within a better approximation ratio than in the worst case. Our first main technical insight is that when the variance of the points is well-spread (i.e., the sampled points do not have too much of their total variance concentrated in any one direction), the *mean* of the points serves as a good proxy for the center. As a side result, this implies strong proper

---

<sup>5</sup>In fact the coreset is a subset of size only  $\lceil 1/\varepsilon \rceil$  that determines a ball that is within a  $(1 + \varepsilon)$  approximation in radius.

learning guarantees for non-worst-case instances, where the approximation ratio improves as the variance is more controlled (see [Theorem 4.3.1](#) and [Theorem 4.1.5](#) in [Section 4.3](#)).

This assumption on the variance of the sampled points is quite strong, and we cannot expect it to hold for arbitrary distributions. Our second main technical insight is that it is possible to precondition the sampled points by applying a linear transformation  $T$  to the points, to move them into a position that meets the strong variance criterion above. Then, we can solve the non-worst-case proper learning task in the transformed space, and apply the inverse transformation  $T^{-1}$  to the resulting ball to construct a confidence set for the original points.

The transformation  $T$  necessarily distorts the space, so when we apply the inverse transformation  $T^{-1}$  to a ball there are two things to keep in mind. First, the result will no longer be a ball, rather it will be an ellipsoid. This falls in the regime of improper learning, and allows us to sidestep the strong lower bounds against proper learning. Second, the transformation  $T$  “shrinks” the space, so the inverse transformation  $T^{-1}$  will expand the volume of the ball.

Our final result follows from carefully choosing the transformation  $T$  to balance two competing objectives:

- (a) We must control the variance of the transformed points to minimize the approximation ratio of the non-worst-case proper learning algorithm.
- (b) We must control the distortion of  $T$  to minimize the volume blow up from applying  $T^{-1}$  to the result of the proper learning algorithm.

Armed with these insights our final algorithm is quite simple: for each coarse set of candidate points (points contained in a ball centered at one of the sample points, there are  $O(n^2)$  such sets), we estimate a linear transformation  $T$ , we transform the points by  $T$ , we find the smallest ball centered at the mean of the transformed points that achieves the desired

coverage, and we apply  $T^{-1}$  to the resulting ball to get a candidate ellipsoid. Finally we take the minimum volume ellipsoid that is found over all coarse sets of candidates.

### 4.2.1 Proper Learning for Non-Worst-Case Instances

To tackle (a) above, our main observation is that we can approximate the center  $c^*$  of the minimum volume ball  $B^*$  containing  $Y^*$  by using the mean  $\mu^*$  of  $Y^*$ . It is not in general true that  $\mu^*$  should be near  $c^*$ . However, *most* points in  $Y^*$  must be approximately as close to  $\mu^*$  as they are to  $c^*$ . This is illustrated in Figure 4.1.

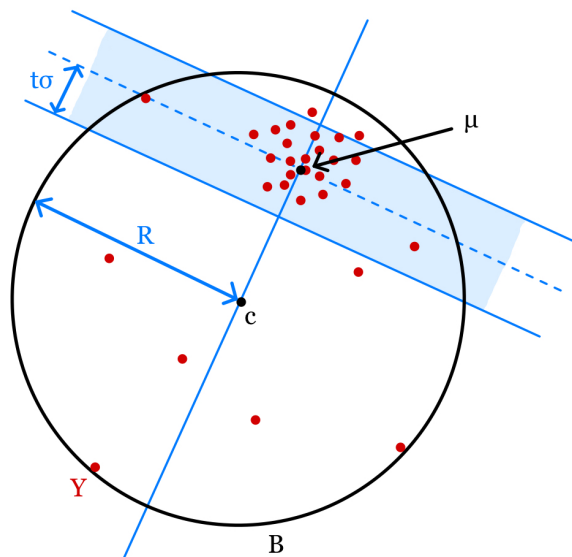


Figure 4.1: The figure shows the points  $Y$  (in red) with mean  $\mu$ , that are contained in ball  $B$  with center  $c$  and radius  $R$ . It is not necessarily the case that  $\mu$  is near  $c$ , as  $B$  can be defined by just a few points. However, Chebyshev's inequality tells us that *most* of the points  $Y$  (depicted here as the points in the shaded region) are within a few standard deviations ( $\sigma$ ) of  $\mu$  in the  $\mu - c$  direction. This allows us to bound the distance of these points to  $\mu$  as  $\leq \sqrt{R^2 + (t\sigma)^2}$ . Formally, this statement corresponds to [Lemma 4.2.1](#)

**Lemma 4.2.1** (For Bounded Variance Distributions, Most Points are Near the Mean). *Let  $Y$  be a set of points in  $\mathbb{R}^d$  that are contained in a ball  $B(c, R)$ . Let  $\mu_Y$  be the empirical mean of  $Y$ , and  $\Sigma_Y$  be the empirical covariance, where the largest eigenvalue of  $\Sigma_Y$  satisfies*

$\lambda_{\max}(\Sigma_Y) \leq \sigma^2$ . Let  $\hat{\mu}$  be an arbitrary point such that  $\|\mu_Y - \hat{\mu}\|_2 \leq \tau$ . Then for any  $t \geq 1$ , there exists a subset  $\hat{Y}$  such that  $|\hat{Y}| \geq (1 - \frac{1}{t^2}) |Y|$ , and for every point  $y \in \hat{Y}$ ,

$$\|y - \hat{\mu}\|_2^2 \leq \|y - c\|_2^2 + t^2(\sigma^2 + \tau^2).$$

*Proof.* As a warm-up, we can bound the number of points that deviate significantly from  $\mu_Y$  using Chebyshev's inequality. Let  $v_\mu = \frac{\mu_Y - c}{\|\mu_Y - c\|}$  be the unit vector in the direction of  $\mu_Y - c$ . We have that

$$\mathbf{P}_{y \sim \text{Unif}(Y)} [|\langle y - \mu_Y, v_\mu \rangle| \geq t\sigma] \leq \frac{1}{t^2}.$$

This means that there exists a subset  $\hat{Y} \subseteq Y$  such that  $|\hat{Y}| \geq (1 - \frac{1}{t^2}) |Y|$ , and for all  $y \in \hat{Y}$  we have  $|\langle y - \mu_Y, v_\mu \rangle| \leq t\sigma$ . For each  $y \in \hat{Y}$ , we can decompose  $y - \mu_Y$  into the portion that is parallel to  $v_\mu$ :  $\overline{(y - \mu_Y)} = \langle y - \mu_Y, v_\mu \rangle v_\mu$ , and the portion that is perpendicular to  $v_\mu$ :  $(y - \mu_Y)^\perp = (y - \mu_Y) - \overline{(y - \mu_Y)}$ . Let  $\overline{(y - c)}$  be the portion of  $y - c$  that is parallel to  $v_\mu$ :  $\overline{(y - c)} = \langle y - c, v_\mu \rangle v_\mu$ . By the choice of  $v_\mu$ , we have that  $\overline{(y - c)} = \overline{(y - \mu_Y)}$ . Thus,

$$\begin{aligned} \|y - \mu_Y\|_2^2 &= \|(y - \mu_Y)^\perp\|_2^2 + \|\overline{(y - \mu_Y)}\|_2^2 \\ &= \|(y - c)^\perp\|_2^2 + \|\overline{(y - \mu_Y)}\|_2^2 \\ &\leq \|y - c\|_2^2 + (t\sigma)^2. \end{aligned}$$

Instead of bounding the number of points that deviate significantly from  $\mu_Y$ , we would like to bound the number of points that deviate significantly from  $\hat{\mu}$ . We observe that a similar



argument goes through. We have that

$$\begin{aligned}
& \mathbf{E}_{y \sim \text{Unif}(Y)} [(y - \hat{\mu})(y - \hat{\mu})^\top] \\
&= \mathbf{E}_{y \sim \text{Unif}(Y)} [(y - \mu_Y + \mu_Y - \hat{\mu})(y - \mu_Y + \mu_Y - \hat{\mu})^\top] \\
&= \mathbf{E}_{y \sim \text{Unif}(Y)} [(y - \mu_Y)(y - \mu_Y)^\top] + \mathbf{E}_{y \sim \text{Unif}(Y)} [(y - \mu_Y)(\mu_Y - \hat{\mu})^\top] \\
&\quad + (\mu_Y - \hat{\mu})\mathbf{E}_{y \sim \text{Unif}(Y)} [(y - \mu_Y)^\top] + (\mu_Y - \hat{\mu})(\mu_Y - \hat{\mu})^\top \\
&= \Sigma_Y + (\mu_Y - \hat{\mu})(\mu_Y - \hat{\mu})^\top \\
&\preceq \Sigma_Y + \tau^2 I.
\end{aligned} \tag{4.2}$$

$$\tag{4.3}$$

Let  $v_{\hat{\mu}} = \frac{\hat{\mu} - c}{\|\hat{\mu} - c\|}$  be a unit vector in the direction of  $\hat{\mu} - c$ . By (4.3) we have that

$$\mathbf{P}_{y \sim \text{Unif}(Y)} \left[ |\langle y - \hat{\mu}, v_{\hat{\mu}} \rangle| \geq t\sqrt{\sigma^2 + \tau^2} \right] \leq \frac{1}{t^2}.$$

Thus there exists a subset  $\hat{Y} \subseteq Y$  such that  $|\hat{Y}| \geq (1 - \frac{1}{t^2})|Y|$ , and for all  $y \in \hat{Y}$  we have  $|\langle y - \hat{\mu}, v_{\hat{\mu}} \rangle| \leq t\sqrt{\sigma^2 + \tau^2}$ . For each  $y \in \hat{Y}$ , we can decompose  $y - \hat{\mu}$  into the portion that is parallel to  $v_{\hat{\mu}}$ :  $\overline{(y - \hat{\mu})} = \langle y - \hat{\mu}, v_{\hat{\mu}} \rangle v_{\hat{\mu}}$ , and the portion that is perpendicular to  $v_{\hat{\mu}}$ :  $(y - \hat{\mu})^\perp = (y - \hat{\mu}) - \overline{(y - \hat{\mu})}$ . Let  $\overline{(y - c)}$  be the portion of  $y - c$  that is parallel to  $v_{\hat{\mu}}$ :  $\overline{(y - c)} = \langle y - c, v_{\hat{\mu}} \rangle v_{\hat{\mu}}$ . By the choice of  $v_{\hat{\mu}}$ , we have that  $\overline{(y - c)} = \overline{(y - \hat{\mu})}$ . Thus we have

$$\begin{aligned}
\|y - \hat{\mu}\|_2^2 &= \|(y - \hat{\mu})^\perp\|_2^2 + \|\overline{(y - \hat{\mu})}\|_2^2 \\
&= \|(y - c)^\perp\|_2^2 + \|\overline{(y - \hat{\mu})}\|_2^2 \\
&\leq \|y - c\|_2^2 + t^2(\sigma^2 + \tau^2).
\end{aligned}$$

□

### 4.2.2 Reducing Worst-Case Improper Learning to Non-Worst-Case Proper Learning

Our goal is to use [Lemma 4.2.1](#) to get a guarantee for learning a small ellipsoid, that does not depend on the variance of the points. As a stepping stone to our population statement, we first prove the following statement about the sample points.

**Theorem 4.2.2** (Finding small volume ellipsoid for  $n$  points). *Suppose we are given a set of points  $Y \subseteq \mathbb{R}^d$ ,  $|Y| = n$ , and  $0 \leq \delta \leq 1, 0 \leq \gamma \leq 1$ , such that there exists a subset  $Y^* \subseteq Y$ ,  $|Y^*| \geq \delta|Y|$ , that is contained in an unknown ball  $B^* = B(c^*, R^*)$ . Then we can find an ellipsoid  $\hat{E}$  such that  $|\hat{E} \cap Y| \geq \delta(1 - \gamma)|Y|$ , and*

$$\text{vol}(\hat{E})^{1/d} \leq \text{vol}(B^*)^{1/d} \cdot (1 + O(d^{-1/2+o(1)}/\gamma\delta)).$$

Intuitively, the case where our non-worst-case proper learning algorithm fails is when the points have most of their variance concentrated in only a few directions, see for example [Figure 4.2](#). This makes it hard to estimate the position of the center accurately in the large variance directions. However, this also means that the variance of the points in the other directions must be low, so it is *easier* to estimate the position of the center in these directions. Applying the correct linear transformation to the points essentially allows us to “reweigh” the accuracy that we are aiming for across the different directions, and trade off error in the high variance directions for accuracy in the low variance directions.

Our strategy is to find a good preconditioner  $\widehat{M}^{-1/2}$ , for the points in  $Y$ , such that the transformed points have small variance. (Writing the preconditioner as  $\widehat{M}^{-1/2}$  is convenient, as we can then argue about the distortion in terms of the eigenvalues of  $\widehat{M}$ , which we will

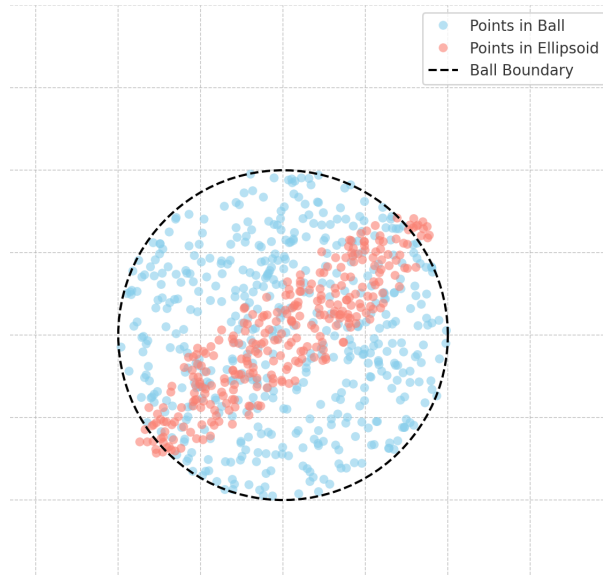


Figure 4.2: The figure shows a ball  $B^*$  of radius  $R^* = \sqrt{d}$  containing a set of points, whose the covariance matrix  $\Sigma_{Y^*}$  is not isotropic and has some directions of high variance. In this case, we can try to find a smaller ellipsoid containing many of the points.

choose to be positive definite.) Then, we can apply our procedure to find an approximately minimum volume ball on the transformed points, which will return a strong approximation since the variance of the points is controlled (via [Lemma 4.2.1](#)). Transforming this ball back into the original space will result in a confidence region that is shaped as an ellipsoid.

Our goal is to choose  $\widehat{M}$  so that the largest variance of the transformed points in any one direction is comparable to the *average* variance across all the directions (i.e., the variance is well-spread, and the distribution is “approximately isotropic”). We choose  $\widehat{M}$  to have eigenvectors aligned with those of the covariance  $\Sigma_{Y'}$  of the points  $Y'$ , where  $Y'$  is some subset of  $Y$  contained in a ball of radius  $R$ , so that the average variance across the  $d$  directions is at most  $R^2/d$ . We choose  $\widehat{M}$  to have the same eigenvectors as  $\Sigma_{Y'}$ , with eigenvalues  $a_i^2$  of  $\widehat{M}$  satisfying  $a_i^2 = d$  if  $\lambda_i \geq \widehat{\tau}^2 R^2/d$  and  $a_i^2 = 1$  if  $\lambda_i < \widehat{\tau}^2 R^2/d$ , where  $\widehat{\tau}$  is a parameter that we choose later.

Essentially, this means that  $\widehat{M}^{-1/2}$  will shrink the large variance directions by a factor  $\sqrt{d}$ , ensuring that the resulting variance in those directions is constant (depending only on  $R$ ). It does nothing for the small variance directions. This choice of  $\widehat{M}$  has the following nice properties.

- (i)  $\widehat{M} \succcurlyeq I$ , and therefore  $\widehat{M}^{-1/2}$  is “shrinking” (non-expanding) in all directions. This ensures that there exists a ball in the transformed space that contains  $Y^\star$  and has volume at most that of  $B^\star$  in the original space.
- (ii) We have that the variance of the points  $Y$  in the transformed space is bounded, i.e., the points in the transformed space are approximately isotropic:

$$\widehat{M}^{-1/2} \Sigma_Y \widehat{M}^{-1/2} \preccurlyeq O\left(\frac{\widehat{\tau}^2 R^2}{d}\right) I_d.$$

This allows us to bound the approximation ratio of applying [Lemma 4.2.1](#) to the transformed points.

- (iii) At most  $d/\widehat{\tau}^2$  eigenvalues of  $\widehat{M}$  are set to  $d$ . This bounds the distortion of  $\widehat{M}$ . That is, for any shape  $S$ ,

$$\frac{\text{vol}(S)}{\text{vol}(\widehat{M}^{-1/2} S)} \leq d^{d/\widehat{\tau}^2} \leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right).$$

Properties (i) and (ii) allow us to reduce to the non-worst-case setting, and property (iii) ensures that this translation does not lose too much in volume. We will set  $\widehat{\tau} = d^{1/4}$  to trade-off the different losses from (ii) and (iii) and obtain the final guarantee. The full proof of this theorem can be found in [Section 4.4](#).

*Remark 4.2.3.* Our main result ([Theorem 4.1.1](#)) gives a volume approximation guarantee of  $\exp(d^{1/2+o(1)})$  in the worst-case. At the same time, [Theorem 4.1.5](#) gives a  $O(1)$  factor

approximation guarantee in volume, when the distribution of points inside the optimal ball  $B^*$  is approximately isotropic. We leave as an open question whether or not there exists a polynomial time (improper) learning algorithm that obtains an  $O(1)$  volume approximation in the worst-case.

### 4.3 Proper Learning of Euclidean Balls

The main technical insight that allows us to get a proper learning algorithm for Euclidean balls in a non-worst-case setting is [Lemma 4.2.1](#), which tells us that for a set of points with bounded variance, the mean of the points is a good proxy for the center (up to a slack factor in coverage).

We use this result as a subroutine in our main result for improper learning. In this section we provide some additional results that extend our result for proper learning to other settings. In particular, [Theorem 4.3.1](#) gives a proper learning algorithm for *fully worst-case* instances, and achieves an approximation ratio of  $\left(1 + \tilde{O}_{\gamma,\delta}\left(\frac{1}{\log d}\right)\right)$  in radius, which is comparable to the factor of  $\left(1 + \tilde{O}_{\gamma,\delta}\left(\frac{1}{\log d}\right)\right)$  due to [[Badoiu, Har-Peled, and Indyk, 2002](#), [Ding, 2020](#)]. While we do achieve a slightly weaker approximation guarantee in this setting, our techniques are significantly different than the coresets based techniques of the prior work, and extend well to non-worst-case settings, as evidenced by our improper learning result. We also provide [Theorem 4.1.5](#), which shows a way to achieve a better approximation ratio, assuming non-worst-case properties only on the *inlier* points.

**Theorem 4.3.1** (Finding approximate minimum volume  $\delta$ -coverage ball). *Given a set of points  $Y \subseteq \mathbb{R}^d$ ,  $|Y| = n$ , and  $0 \leq \delta \leq 1$ ,  $0 \leq \gamma \leq 1$ , such that there exists a subset  $Y^* \subseteq Y$ ,  $|Y^*| \geq \delta|Y|$ , that is contained in an unknown ball  $B^* = B(c^*, R^*)$ , we can find a*

ball  $\widehat{B} = B(\widehat{c}, \widehat{R})$  such that  $|Y \cap \widehat{B}| \geq (1 - \gamma)|Y^*|$ , and

$$\text{vol}(\widehat{B})^{1/d} \leq \text{vol}(B^*)^{1/d} \left( 1 + O\left(\frac{\log \log d}{\gamma \delta \log d}\right) \right)$$

in polynomial time in  $n, d$ .

The algorithm for finding this small volume coverage ball is given in [Algorithm 4.1](#). By using the fact that the VC-dimension of Euclidean balls in  $d$ -dimensions is at most  $d + 1$ , we immediately get the following corollary.

**Corollary 4.3.2** (Population version of [Theorem 4.3.1](#)). *Suppose  $Y \subset \mathbb{R}^d$  is a set of  $n$  i.i.d. samples generated from an arbitrary distribution  $\mathcal{D}$ . For a target coverage  $\delta \in (0, 1)$  and a coverage approximation factor  $\gamma \in (0, 1)$  such that  $n = \Omega(d)$ , in  $\text{poly}(n, d)$  time, with high probability, we can find a ball  $\widehat{B}$  such that with high probability,*

$$\mathbf{P}_{y \sim \mathcal{D}} [y \in \widehat{B} \mid Y] \geq \delta,$$

and

$$\text{vol}(\widehat{B})^{1/d} \leq \text{vol}(B^*)^{1/d} \left( 1 + O\left(\frac{\log \log d}{\gamma \delta \log d}\right) \right)$$

where  $B^*$  is the minimum volume ball that achieves at least  $\delta + \gamma + O(\sqrt{d/n})$  coverage over  $\mathcal{D}$ .

Our approach starts with the following structural observation about the true high-density ball  $B^*$ . It is not in general true that the mean  $\mu^*$  of the points  $Y^* = B^* \cap Y$  should be near the center  $c^*$  of  $B^*$ . However, it turns out that if the mean is far from the center, taking a ball centered at the mean of  $B^*$  must still capture *most* of the points in  $Y^*$ . This is a simple consequence of Chebyshev's inequality—if we are able to control the variance of the points in  $Y^*$  ([Lemma 4.2.1](#), [Figure 4.1](#)). Thus our problem reduces to the following. Given only the

full set of points  $Y$ , we must estimate the mean of  $Y^*$ , while also controlling the variance of the points that we are taking the mean of.

The following simple lemma from robust statistics shows that the empirical mean of a set of points with bounded covariance that contains both inliers and outliers is close to the true mean.

**Lemma 4.3.3** (High probability event has close-by mean). *[see e.g., Lemma 2.1 of [Steinhardt, 2019](#)] Let  $X \in \mathbb{R}^d$  be a random variable with mean  $\mu \in \mathbb{R}^d$  and covariance matrix  $\Sigma$ , where the largest eigenvalue satisfies  $\lambda_{\max}(\Sigma) \leq \sigma^2$ . Then for any event  $E$  with probability at least  $\delta > 0$ , the conditional expectation satisfies:*

$$\|\mathbb{E}[X|E] - \mu\|_2 \leq \sigma \cdot \sqrt{2(1-\delta)/\delta}.$$

The following simple claim shows that the variance does not increase much when we condition on an event with non-negligible probability.

**Claim 4.3.1** (High probability event has similar variance). *Let  $X \in \mathbb{R}^d$  be a random variable whose maximum variance in any direction is at most  $\sigma^2$ , and let  $E$  be an event that occurs with probability at least  $\delta$ . Then the maximum variance in any direction of  $X$  conditioned on  $E$  is at most  $\frac{\sigma^2}{\delta}$ .*

*Proof.* This follows from a simple averaging argument. Let  $\mu$  be the mean of  $X$ . We have that the variance of  $X$  is at most  $\sigma^2$ . That is, for all unit vectors  $v$ ,

$$\mathbf{E} [\langle X - \mu, v \rangle^2] \leq \sigma^2.$$

Now, we can decompose this into the variance conditioned on  $E$ ,

$$\begin{aligned}
\sigma^2 &\geq \mathbf{E} [\langle X - \mu, v \rangle^2] \\
&\geq \mathbf{E} [\langle X - \mu, v \rangle^2 | E] \mathbf{P}[E] + \mathbf{E} [\langle X - \mu, v \rangle^2 | \neg E] \mathbf{P}[\neg E] \geq \mathbf{E} [\langle X - \mu, v \rangle^2 | E] \delta \\
\frac{\sigma^2}{\delta} &\geq \mathbf{E} [\langle X - \mu_E, v \rangle^2 | E],
\end{aligned}$$

where in the last line  $\mu_E$  is the mean of  $X$  conditioned on  $E$ , and the last line follows because the mean minimizes the squared deviation. Thus we can conclude that the variance of  $X$  given  $E$  is bounded by  $\frac{\sigma^2}{\delta}$  in every direction.  $\square$

To use the above observation to find an approximate center, we need to do two things. First, we need to control the standard deviation of the points that we are considering. In particular, to achieve a subconstant approximation of radius, we need that the standard deviation of our points is  $\frac{1}{\omega(1)} R^*$ . Second, we need to be able to find  $\mu^*$  approximately, which is akin to robust mean estimation.

In the first step, we use a simple search procedure that gives us a coarse estimate of  $B^*$ . That is, we find a ball  $B'$  such that  $Y^* \subseteq B'$  and the radius  $R'$  of  $B'$  is bounded by  $R^* \leq R' \leq 2R^*$ . In the worst case, we cannot expect that the variance is less than  $(R')^2$ , for example if all of the variance of  $Y'$  is concentrated in one direction. However, there can be at most  $\log d$  directions with variance higher than  $(R')^2 / \log d$ . Thus we can find a list of candidate centers for  $Y^*$  by estimating the location of the center separately in the high variance space and the low variance space. In the high variance space, we can grid search to find a list of potential candidates for the center. This is tractable since the high variance space has dimension  $\log d$ . In the low variance space we can estimate the location of  $\mu^*$ .

**Lemma 4.3.4** (Grid search). *Suppose we are given a ball  $B' = B(0, R')$  containing a set*



of points  $Y' \in \mathbb{R}^q$ , such that there exists a subset of points  $Y^* \in Y'$  that are contained in a smaller ball  $B^* = B(c^*, R^*)$  with  $c^* \in B'$ , where  $|Y^*| \geq \delta|Y'|$ .

Then for a tolerance  $0 < \tau < R'$ , there exists a list  $L$  of size  $\left(\frac{2+\tau/R'}{\tau/R'}\right)^q$ , such that  $L$  contains a point  $\hat{c} \in \mathbb{R}^q$  such that for every  $y \in Y^*$ ,

$$\|y - \hat{c}\| \leq \|y - c^*\| + \tau.$$

*Proof.* We first construct a  $\frac{\tau}{R'}$ -net for the unit ball in  $\mathbb{R}^q$ , denoted as  $L_0$ , with size  $\left(\frac{2+\tau/R'}{\tau/R'}\right)^q$ . This ensures that every point in the unit ball is at most distance  $\frac{\tau}{R'}$  from some point in the net  $L_0$ . Next, we scale the net  $L_0$  to  $L = R'L_0$ . This ensures that every point in  $B' = B(0, R')$  is at most distance  $\tau$  from some point in  $L$ . Thus, for the center  $c^* \in B'$ , there is a point  $\hat{c} \in L$  such that  $\|\hat{c} - c^*\| \leq \tau$ . By the triangle inequality, for every  $y \in Y^*$ , we have  $\|y - \hat{c}\| \leq \|y - c^*\| + \|\hat{c} - c^*\| \leq \|y - c^*\| + \tau$ .  $\square$

**Lemma 4.3.5** (Coarse-to-fine refinement). *Suppose we are given a ball  $B' = B(0, R')$  containing a set of points  $Y' \subseteq \mathbb{R}^d$ , where the  $(q+1)$ -th largest eigenvalue of the covariance matrix  $\Sigma_{Y'}$  of  $Y'$ , denoted as  $\lambda_{q+1}(\Sigma_{Y'})$ , is bounded by*

$$\lambda_{(q+1)}(\Sigma_{Y'}) \leq \sigma^2.$$

*Suppose there exists a subset of points  $Y^* \subseteq Y'$  are contained in a smaller ball  $B^* = B(c^*, R^*)$ , where  $|Y^*| \geq \delta|Y'|$  and  $R' \leq 2R^*$ . Let  $\gamma > 0$  be a coverage slack factor.*

*Then there exists a list  $C$  of size at most  $\left(\frac{3R'}{\tau}\right)^q$  and a set  $\hat{Y} \subseteq Y^*$  with  $|\hat{Y}| \geq (1 - \gamma)|Y^*|$ ,*

such that there exists a point  $\hat{c} \in C$  which for every  $y \in \hat{Y}$  satisfies

$$\begin{aligned} \|y - \hat{c}\|_2^2 &\leq \|\Pi_{\text{low}}(y - c^*)\|_2^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1-\delta)}{\delta}\right) + (\|\Pi_{\text{high}}(y - c^*)\|_2 + \tau)^2 \\ &\leq \left(\|y - c^*\|_2 + O\left(\frac{\sigma}{\sqrt{\gamma\delta}}\right) + \tau\right)^2, \end{aligned}$$

where  $\Pi_{\text{high}}$  is the projection onto the top- $q$  eigenspace of  $\Sigma_{Y'}$  and  $\Pi_{\text{low}}$  is the projection onto the space orthogonal to the top- $q$  eigenspace of  $\Sigma_{Y'}$ . For  $q = 0$ , we get a single estimate  $\hat{c}$ , and  $\hat{Y} \subseteq Y^*$  with  $|\hat{Y}| \geq (1 - \gamma)|Y^*|$ , that for every  $y \in \hat{Y}$  satisfies

$$\|y - \hat{c}\|_2^2 \leq \|y - c^*\|_2^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1-\delta)}{\delta}\right).$$

*Proof.* We start by considering  $\Sigma_{Y'}$ , the covariance matrix of  $Y'$  and partition its eigenvalues into two groups. The directions associated with the largest  $q$  eigenvalues of  $\Sigma_{Y'}$  are referred to as the “high variance directions”; while the directions corresponding to the remaining eigenvalues are the “low variance directions”. Let  $\Pi_{\text{high}} : \mathbb{R}^d \rightarrow \mathbb{R}^q$  denote the projection onto the “high variance” eigenspace and  $\Pi_{\text{low}} : \mathbb{R}^d \rightarrow \mathbb{R}^{d-q}$  denote the projection onto the “low variance” eigenspace. We estimate the mean  $\mu^*$  of  $Y^*$  separately in the high variance and low variance directions.

First, if  $q > 0$ , we estimate the location of  $\mu^*$  in the high variance directions. Let  $Y'_{\text{high}} = \Pi_{\text{high}}Y'$  denote the projection of points in  $Y'$  onto the high variance directions. Note that the set  $Y'_{\text{high}}$  lies within the  $q$ -dimensional ball  $B'_{\text{high}} = \Pi_{\text{high}}B'$ , which is centered at the origin and has a radius of at most  $R'$ . Next, consider the projection of  $Y^*$  onto the high variance directions, denoted by  $Y^*_{\text{high}} = \Pi_{\text{high}}Y^*$ . Then, this set  $Y^*_{\text{high}}$  is contained in the ball  $B^*_{\text{high}} = \Pi_{\text{high}}B^*$  centered at  $\Pi_{\text{high}}c^*$  with radius at most  $R^*$ . Since  $B^*_{\text{high}} \subseteq B'_{\text{high}}$ , and  $\mu^* \in B'_{\text{high}}$  as

well, by [Lemma 4.3.4](#), there exists a list  $L \subset \mathbb{R}^q$  with size  $|L| \leq \left(\frac{2+\tau/R'}{\tau/R'}\right)^q \leq \left(\frac{3R'}{\tau}\right)^q$  such that there is a point  $\widehat{\ell} \in L$  such that for every point  $y \in Y^*$ ,

$$\|\Pi_{\text{high}}y - \widehat{\ell}\| \leq \|\Pi_{\text{high}}y - \Pi_{\text{high}}c^*\| + \tau. \quad (4.4)$$

Next, we estimate the location of  $\mu^*$  in the low variance directions. Let  $Y'_{\text{low}} = \Pi_{\text{low}}Y'$  denote the projection of  $Y'$  onto the low variance directions. By construction, the set  $Y'_{\text{low}}$  has variance at most  $\lambda_{(q+1)}(\Sigma_{Y'}) \leq \sigma^2$  in every direction. Since the set  $Y^*$  contains at least a  $\delta$ -fraction of the points in  $Y'$ , by [Claim 4.3.1](#), we have  $Y^*_{\text{low}} = \Pi_{\text{low}}Y^*$  has variance at most  $\sigma^2/\delta$  in every direction. By [Lemma 4.3.3](#), the mean  $\mu^*_{\text{low}}$  of  $Y^*_{\text{low}}$  is close to the mean  $\mu'_{\text{low}}$  of  $Y'_{\text{low}}$ , i.e.,

$$\|\mu^*_{\text{low}} - \mu'_{\text{low}}\|_2 \leq \sigma \cdot \sqrt{2(1-\delta)/\delta}. \quad (4.5)$$

Since the variance of  $Y^*_{\text{low}}$  is bounded by  $\sigma^2/\delta$ , and  $\mu'_{\text{low}}$  is close to  $\mu^*_{\text{low}}$ , [Lemma 4.2.1](#) tells us that, for a fixed choice of  $\gamma$ , there exists a subset  $\widehat{Y} \subseteq Y^*$  such that  $|\widehat{Y}| \geq (1-\gamma)|Y^*|$ , and for every point  $y \in \widehat{Y}$ ,

$$\|\Pi_{\text{low}}y - \mu'_{\text{low}}\|_2^2 \leq \|\Pi_{\text{low}}(y - c^*)\|_2^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1-\delta)}{\delta}\right).$$

For  $q = 0$ , this gives the final bound.

For  $q > 0$ , we take  $\widehat{c}$  such that  $\Pi_{\text{high}}\widehat{c} = \widehat{\ell}$  and  $\Pi_{\text{low}}\widehat{c} = \mu'_{\text{low}}$ . Putting the bounds together,

we have that there exists a set  $\widehat{Y} \subseteq Y^\star$ , with  $|\widehat{Y}| \geq (1 - \gamma)|Y^\star|$ , such that for every  $y \in \widehat{Y}$ ,

$$\begin{aligned} \|y - \widehat{c}\|_2^2 &= \|\Pi_{\text{low}}(y - \widehat{c})\|_2^2 + \|\Pi_{\text{high}}(y - \widehat{c})\|_2^2 \\ &\leq \|\Pi_{\text{low}}(y - c^\star)\|_2^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1 - \delta)}{\delta}\right) + (\|\Pi_{\text{high}}(y - c^\star)\|_2 + \tau)^2 \\ &\leq \left(\|y - c^\star\|_2 + O\left(\frac{\sigma}{\sqrt{\gamma\delta}}\right) + \tau\right)^2, \end{aligned}$$

where both the tighter bound (second line) and looser bound (third line) will be useful in applications of the lemma.  $\square$

**Lemma 4.3.6** (Most directions are low variance). *Let  $Y$  be a set of points contained in  $B(0, R)$  with mean  $\mu_Y$ . The covariance of  $Y$  can have at most  $q$  eigenvalues greater than  $\frac{R^2}{q}$ .*

*Proof.* Denote the covariance of  $Y$  by  $\Sigma_Y = \frac{1}{|Y|} \sum_{y \in Y} (y - \mu_Y)(y - \mu_Y)^\top$ , where  $\mu_Y$  is the mean of the points  $Y$ . We have that

$$\Sigma_Y \preceq \frac{1}{|Y|} \sum_{y \in Y} yy^\top,$$

since the mean minimizes the deviation. Since trace follows the PSD domination, we have that

$$\text{tr}(\Sigma_Y) \leq \text{tr}\left(\frac{1}{|Y|} \sum_{y \in Y} yy^\top\right).$$

Since trace is a linear operator and  $Y$  is contained in  $B(0, R)$ , we have that

$$\text{tr}\left(\frac{1}{|Y|} \sum_{y \in Y} yy^\top\right) = \frac{1}{|Y|} \sum_{y \in Y} \text{tr}(yy^\top) = \frac{1}{|Y|} \sum_{y \in Y} \|y\|_2^2 \leq R^2.$$

So we have that  $\text{tr}(\Sigma_Y) \leq R^2$ . Since the trace is equal to the sum of the eigenvalues, and  $\Sigma_Y$

---

**Algorithm 4.1** Algorithm DENSE-BALL for finding a small volume ball that contains at least  $\delta' = \delta(1 - \gamma)$  fraction of points

---

**Input:** a set of point  $Y \in \mathbb{R}^d$ , target fraction  $\delta \in (0, 1)$ , and slack  $\gamma \in (0, 1)$

**Output:** an ball  $\widehat{B} \subset \mathbb{R}^d$

1. Create a list of coarse balls  $\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\}$  and remove from  $\mathfrak{B}$  all balls that contain less than  $\delta n$  points in  $Y$ .
  2. Set  $R_{\min}$  to be the minimum radius among all balls in  $\mathfrak{B}$ . Then remove from  $\mathfrak{B}$  all balls with a radius greater than  $2R_{\min}$ .
  3. Create a list of refined balls  $\mathfrak{B}_2$  as follows:
    - (a) Create a list of candidate centers  $C_{B'}$  for each ball  $B'$  in  $\mathfrak{B}$  as follows.
    - (b) Compute the eigenvalues  $\lambda_1, \dots, \lambda_d$  and eigenvectors  $v_1, \dots, v_d$  of the covariance  $\Sigma_{Y'}$  of points  $Y' = Y \cap B'$ .
    - (c) Set  $\Pi_{\text{high}}$  be the projection onto the eigenspace of  $\Sigma_{Y'}$  with eigenvalues greater than  $(R')^2 \log \log d / \log d$  and  $\Pi_{\text{low}} = I - \Pi_{\text{high}}$ . Set  $B'_{\text{high}} = \Pi_{\text{high}} B'$  and  $Y'_{\text{low}} = \Pi_{\text{low}} Y'$ . Note that there are at most  $q = \log d / \log \log d$  eigenvalues of  $\Sigma_{Y'}$  is greater than  $(R')^2 \log \log d / \log d$ . When  $q > 0$ , we run a grid search over these high-variance directions; however, when  $q = 0$ , no grid search is needed.
    - (d) Set the list  $C_{B'} = \{c : \Pi_{\text{high}} c = \ell, \Pi_{\text{low}} c = \mu'_{\text{low}}, \ell \in L\}$ , where  $\mu'_{\text{low}}$  is the mean of  $Y'_{\text{low}}$  and  $L$  is the  $\tau$ -net of the ball  $B'_{\text{high}}$  for  $\tau = R_{\min} / \log d$ .
    - (e) Combine all candidate centers  $\mathfrak{C} = \cup_{B' \in \mathfrak{B}} C_{B'}$ .
    - (f) Set the list  $\mathfrak{B}_2 = \{B(c, \|y - c\|) \mid c \in \mathfrak{C}, y \in Y\}$ .
  4. Return the ball  $\widehat{B} \in \mathfrak{B}_2$  with the smallest volume that contains at least  $\delta(1 - \gamma)$  points in  $Y$ .
- 

is positive semidefinite and thus has all non-negative eigenvalues, we have by an averaging argument that  $\Sigma_Y$  can have at most  $q$  eigenvalues  $\geq \frac{R^2}{q}$ .  $\square$

### 4.3.1 Completing the Proof of Theorem 4.3.1

*Proof of Theorem 4.3.1.* We begin by finding a list of coarse guesses for  $B^*$ . We create a list of balls

$$\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\}.$$

Note that  $|\mathfrak{B}| \leq n^2$  since  $|Y| = n$ . Consider the subset  $Y^* \subseteq Y$  contained in ball  $B^* = B(c^*, R^*)$ . Choosing

$$y_1^*, y_2^* = \operatorname{argmax}_{y_1, y_2 \in Y^*} \|y_1 - y_2\|,$$

gives us that the ball

$$B' = B(y_1^*, \|y_1^* - y_2^*\|) \quad (4.6)$$

contains all points in  $Y^*$ , has radius  $R' \leq 2R^*$ , and  $B' \in \mathfrak{B}$ .

We now do a filtering step to get a coarse estimate of  $R^*$ , and remove any balls from  $\mathfrak{B}$  that are too large. In particular, let  $R_{\min}$  be the minimum radius among balls in  $\mathfrak{B}$  that contain at least  $\delta n$  points. Since  $B^*$  is the volume optimal ball that contains at least  $\delta$ -fraction of  $Y$ , we have  $R_{\min} \geq R^*$ . Thus, we know that

$$R^* \leq R_{\min} \leq 2R^*.$$

We filter  $\mathfrak{B}$  to only contain balls that contain at least  $\delta n$  points from  $Y$ , and have radius at most  $2R_{\min} \leq 4R^*$ . Finding  $R_{\min}$  and filtering the list take  $\tilde{O}(n^3)$  time.

Now we apply the coarse-to-fine refinement of [Lemma 4.3.5](#) to each ball in (the filtered)  $\mathfrak{B}$ , to generate a list of candidate centers for  $\hat{B}$ . In particular, we apply the procedure for  $q = \frac{\log d}{\log \log d}$  and  $\tau = \frac{R_{\min}}{\log d}$ . For each ball  $B$  in the filtered  $\mathfrak{B}$ , we have that the radius  $R$  of  $B$  is at most  $2R_{\min}$ . Thus, by [Lemma 4.3.5](#), the size of the list of potential centers  $C_B$  that we generate for  $B$  is at most

$$|C_B| \leq \left( \frac{3R'}{\tau} \right)^q \leq (6 \log d)^{\frac{\log d}{\log \log d}} \leq d^3.$$

Aggregating over the up to  $n^2$  balls in  $\mathfrak{B}$ , we collect a list  $\mathfrak{C} = \bigcup_{B \in \mathfrak{B}} C_B$  of most  $n^2 d^{\frac{7}{2}}$

potential centers.

Now, we do a more refined search for  $B^*$  using the candidate centers in  $\mathfrak{C}$ . That is, we generate the list

$$\mathfrak{B}_2 = \{B(c, \|y - c\|) \mid c \in \mathfrak{C}, y \in Y\}.$$

Note that  $|\mathfrak{B}_2| \leq |\mathfrak{C}| \cdot n \leq n^3 d^{\frac{7}{2}}$ . Then, we output the minimum volume ball  $B \in \mathfrak{B}_2$ , such that  $|B \cap Y| \geq \delta(1 - \gamma)n$ . Finding this ball can be done in time  $\tilde{O}(|\mathfrak{B}_2| \cdot n) = \tilde{O}(n^4 d^{\frac{7}{2}})$ . Thus the total runtime of our algorithm is bounded by  $\tilde{O}(n^4 d^{\frac{7}{2}})$ .

Now we argue that  $\mathfrak{B}_2$  contains a ball that is a good approximation to  $B^*$  in volume. Consider the ball  $B'$  from (4.6). We have that  $B'$  contains all points in  $Y^*$ , has radius  $R' \leq 2R^*$ , and is in the list  $\mathfrak{B}$ . Let  $Y' = Y \cap B'$  be the subset of points of  $Y$  that are captured by  $B'$ . An averaging argument (Lemma 4.3.6) tells us that  $\lambda_{(q+1)}(\Sigma_{Y'}) \leq \frac{R'^2}{q}$ , where  $\Sigma_{Y'}$  is the covariance of the points  $Y'$ , and  $\lambda_{(q+1)}(\Sigma_{Y'})$  is the  $(q+1)$ -th largest eigenvector of  $\Sigma_{Y'}$ . Thus, when we apply the procedure of Lemma 4.3.5 to  $B'$  with  $q = \frac{\log d}{\log \log d}$  and  $\tau = \frac{R_{\min}}{\log d}$ , we also have that the eigenvalue  $\lambda_{(q+1)}(\Sigma_{Y'})$  is upper bounded by  $\sigma^2 \leq \frac{R'^2}{q} \leq \frac{(2R^*)^2 \log \log d}{\log d}$ . This guarantees that, for the list of potential centers  $C_{B'}$  that we get for this ball  $B'$ , there is a center  $\hat{c} \in C_{B'}$  and a subset  $\hat{Y} \subseteq Y^*$ ,  $|\hat{Y}| \geq (1 - \gamma)|Y^*|$ , such that for every  $y \in \hat{Y}$ ,

$$\|y - c^*\|_2^2 \leq \|\Pi_{\text{low}}(y - c^*)\|_2^2 + O\left(\frac{\sigma^2}{\gamma\delta}\right) + (\|\Pi_{\text{high}}(y - c^*)\|_2 + \tau)^2,$$

where  $\Pi_{\text{high}}$  is the projection onto the top- $q$  eigenspace of  $\Sigma_{Y'}$  and  $\Pi_{\text{low}}$  is the projection onto the space orthogonal to the top- $q$  eigenspace of  $\Sigma_{Y'}$ .

To bound the second term, we use the fact that

$$(a + b)^2 \leq (1 + \varepsilon)a^2 + \left(1 + \frac{1}{\varepsilon}\right)b^2, \quad \forall \varepsilon > 0.$$

Using  $\tau = \frac{R_{\min}}{\log d}$ , and  $\varepsilon = \frac{1}{\log d}$ , we get

$$\begin{aligned}
\|y - c^*\|_2^2 &\leq \|\Pi_{\text{low}}(y - c^*)\|_2^2 + O\left(\frac{R'^2 \log \log d}{\gamma \delta \log d}\right) \\
&\quad + \left(1 + \frac{1}{\log d}\right) \|\Pi_{\text{high}}(y - c^*)\|_2^2 + (1 + \log d) \cdot \frac{R_{\min}^2}{\log^2 d} \\
&\leq \left(1 + \frac{1}{\log d}\right) \|y - c^*\|_2^2 + O\left(\frac{(R^*)^2 \log \log d}{\gamma \delta \log d}\right) \\
&\leq (R^*)^2 \left(1 + O\left(\frac{\log \log d}{\gamma \delta \log d}\right)\right).
\end{aligned}$$

Thus, it is guaranteed that the list of potential centers  $C_{B'}$  that we get for this ball  $B'$  contains a center  $\hat{c} \in C_{B'}$  such that  $\hat{B} = B(\hat{c}, \hat{R})$  contains at least  $(1 - \gamma)|Y^*|$  points for

$$\hat{R} \leq R^* \left(1 + O\left(\frac{\log \log d}{\gamma \delta \log d}\right)\right)^{1/2},$$

which corresponds to a volume bound of

$$\text{vol}(\hat{B}) \leq \text{vol}(B^*) \exp\left(O\left(\frac{d \log \log d}{\gamma \delta \log d}\right)\right).$$

Since  $\hat{c} \in \mathfrak{C}$ , we are guaranteed that the smallest ball  $B_2 = B(\hat{c}, \|y - c\|)$  for  $y \in Y$  that contains at least  $(1 - \gamma)|Y^*|$  points can have radius at most that of  $\hat{B}$ . Since  $B_2 \in \mathfrak{B}_2$ , we know that the minimum volume ball in  $\mathfrak{B}_2$  that contains at least  $\delta(1 - \gamma)n$  points from  $Y$  can only have radius at most that of  $B_2$ , which is at most  $\hat{R}$ .  $\square$



### 4.3.2 Better Volume Approximation for Almost Isotropic Distributions

Our strategy starts by finding a ball in  $B'$  that contains all of the points in  $B^*$  and has radius at most twice that of  $B^*$ . (More precisely, we find a list of candidate balls, which can be done simply.) Trivially, this means that in every direction the standard deviation of the points in  $B'$  is at most  $R^*$ . [Lemma 4.3.5](#) tells us that choosing the mean of the points in  $B'$  as the center of  $\hat{B}$  essentially allows us to capture most of the points in  $B^*$  with radius that is  $R^*$  plus the maximum standard deviation in any direction. This gives a 2-approximation to the radius. We note that even a slightly stronger bound on the standard deviation can improve the approximation factor to  $1+o(1)$ . In [Theorem 4.3.1](#), we do this by grid-searching the  $\log d$  many highest variance directions of the points in  $B'$ , and taking the mean in the low-variance directions. This allows us to argue about the standard deviation of the  $(\log d + 1)$ th highest variance direction, which can be at most  $\frac{R^*}{\sqrt{\log d}}$  ([Lemma 4.3.6](#)).

This bound on the variance that we use in [Theorem 4.3.1](#) is the best that our approach achieves, since it is possible that the variance of the points is concentrated in a few directions. However, if the points in  $B^*$  (the “inliers”) are not worst-case, and are instead approximately isotropic, we expect the standard deviation in any direction to be bounded by  $\frac{R^*}{\sqrt{d}}$ , which we can use to give a better bound on the radius of the ball that we output. In our earlier argument, we bounded the variance not only of the inliers— the points in  $B^*$ — but also all of  $Y'$ , which includes some of the outliers. To utilize this weaker assumption on the variance, we appeal to algorithms for list-decodable robust mean estimation, which work exactly in this setting where the inliers are well-behaved, but the outliers may be arbitrary.

This observation is also useful as a subroutine for our algorithm that finds a small volume ellipsoid in [Section 4.4](#). At a high level, that algorithm will first estimate a linear transforma-

---

**Algorithm 4.2** Algorithm DENSE\_BALL\_ISOTROPIC for finding a small volume ball that contains at least  $\delta' = \delta(1 - \gamma)$  fraction of points for isotropic distributions

---

**Input:** a set of point  $Y \in \mathbb{R}^d$ , target fraction  $\delta \in (0, 1)$ , slack  $\gamma \in (0, 1)$ , isotropic parameter  $\beta \in (0, 1)$

**Output:** a ball  $\hat{B} \subset \mathbb{R}^d$

1. Create a list of coarse balls  $\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\}$  and remove from  $\mathfrak{B}$  all balls that contain less than  $\delta n$  points in  $Y$ .
  2. Set  $R_{\min}$  to be the minimum radius among all balls in  $\mathfrak{B}$ .
  3. Create a list of candidate centers  $L$  by running the list-decodable mean estimation algorithm [Diakonikolas, Kane, Kongsgaard, Li, and Tian, 2021, Diakonikolas and Kane, 2023] on points  $Y$ , target fraction  $\delta$ , and variance  $\beta(R_{\min})^2$ .
  4. Create a list of refined balls  $\mathfrak{B}_2 = \{B(c, \|y - c\|) \mid c \in L, y \in Y\}$ .
  5. Return the ball  $\hat{B} \in \mathfrak{B}_2$  with the smallest volume that contains at least  $\delta(1 - \gamma)$  points in  $Y$ .
- 

tion that limits the variance of the points in  $B^*$ . Then, after applying this transformation, we essentially use Algorithm DENSE\_BALL\_ISOTROPIC (Algorithm 4.2) to search for a ball in the transformed space with a better volume approximation. In that application, we will apply the transformation to and control the variance of all of our points, inliers and outliers, and thus avoid the need for list-decodable robust mean estimation.

**Theorem 4.1.5** (Bounded variance implies better bounds). *Let  $Y \subseteq \mathbb{R}^d$  be a set of  $n$  points, and  $\delta, \gamma \in (0, 1)$ , such that there is an unknown ball  $B^* = B(c^*, R^*)$  with  $Y^* = B^* \cap Y$  satisfying  $|Y^*| \geq \delta|Y|$ , and  $Y^*$  is  $\beta(R^*)^2/d$ -isotropic for some  $\beta \geq 0$ . That is, if  $\mu^*$  is the mean of the points  $Y^*$ , then*

$$\Sigma_{Y^*} = \frac{1}{|Y^*|} \sum_{y \in Y^*} (y - \mu^*)(y - \mu^*)^\top \preceq \beta \frac{(R^*)^2}{d} I.$$

*Then we can find a ball  $\hat{B} = B(\hat{c}, \hat{R})$  such that  $|Y \cap \hat{B}| \geq (1 - \gamma)|Y^*|$ , and*

$$\text{vol}(\hat{B})^{1/d} \leq \text{vol}(B^*)^{1/d} \sqrt{1 + O\left(\frac{\beta}{\gamma \delta d}\right)},$$

in polynomial time.

*Proof.* We begin by finding a coarse estimate of  $R^*$ . We do this as in [Theorem 4.3.1](#): we can create a list of coarse guesses for  $B^*$ :

$$\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\}.$$

By taking  $y_1, y_2$  to be maximally distant points in  $B^*$ , we have that there is a  $\hat{B} \in \mathfrak{B}$  such that the radius  $\hat{R}$  of  $\hat{B}$  satisfies

$$R^* \leq \hat{R} \leq 2R^*.$$

Taking the minimum radius  $R_{\min}$  over all balls in  $\mathfrak{B}$  that covers at least  $\delta n$  points gives us an estimate of  $R^*$  such that  $R^* \leq R_{\min} \leq \hat{R} \leq 2R^*$ .

Let  $\mu^*$  be the mean of the points in  $Y^*$ . We know that  $Y^*$  has covariance p.s.d. dominated by  $\frac{\beta(R^*)^2}{d}I$ , which is in turn p.s.d. dominated by  $\frac{\beta(R_{\min})^2}{d}I$ . The polynomial-time algorithm for list-decodable mean estimation [[Diakonikolas, Kane, Kongsgaard, Li, and Tian, 2021](#), [Diakonikolas and Kane, 2023](#)] outputs a list  $L$  of length  $O(\log(1/p)/\delta)$  such that with probability  $1 - p$ , there exists a  $\hat{\mu} \in L$  such that

$$\|\hat{\mu} - \mu^*\| = O\left(\left(\frac{\beta}{\delta d}\right)^{1/2} R_{\min}\right) = O\left(\left(\frac{\beta}{\delta d}\right)^{1/2} R^*\right).$$

By [Lemma 4.2.1](#) we have that there exists a subset  $\hat{Y} \subseteq Y \cap B^*$ , such that  $|\hat{Y}| \geq (1 - \gamma)|Y \cap$

$B^*$ ], and for points  $y \in \widehat{Y}$ ,

$$\begin{aligned} \|y - \widehat{\mu}\|_2^2 &\leq \|y - c^*\|_2^2 + \gamma^{-1} \left( \frac{\beta}{d} (R_{\min})^2 + O(\delta^{-1}(\beta/d)(R^*)^2) \right) \\ &\leq (R^*)^2 + O(\gamma^{-1}\delta^{-1}(\beta/d)(R^*)^2) \\ &\leq (R^*)^2 \left( 1 + O\left(\frac{\beta}{\gamma\delta d}\right) \right) \end{aligned}$$

Now, we can do one more search step and create a refined list of candidate balls

$$\mathfrak{B}_2 = \{B(\widehat{\mu}, \|\widehat{\mu} - y\|) \mid \widehat{\mu} \in L, y \in Y\}.$$

Since  $|\mathfrak{B}_2| \leq |L| \cdot |Y|$ , we have that this list is polynomially sized. We output the minimum volume ball  $\widehat{B} \in \mathfrak{B}_2$  such that  $|\widehat{B} \cap Y| \geq (1 - \gamma)\delta|Y|$ . This guarantees that  $\widehat{B}$  has radius  $\widehat{R}$  with

$$\widehat{R} \leq R^* \sqrt{1 + O\left(\frac{\beta}{\gamma\delta d}\right)}.$$

□

## 4.4 Finding Confidence Sets $(\Gamma = 1 + d^{-1/2+o(1)})$ -Competitive with Euclidean Balls

In this section, we provide an algorithm to find a confidence set whose volume is within a  $\exp(O(d^{1/2+o(1)}))$ -factor of the volume of the smallest ball that contains a  $\delta$  fraction of  $Y$ .

**Theorem 4.1.1** (Learning confidence sets competitive with Euclidean balls). *There is a polynomial time algorithm that for any target coverage  $\delta \in (0, 1)$  and coverage slack  $\gamma \in (0, 1)$ , when given  $n = \Omega(d^2/\gamma^2)$  samples drawn i.i.d. from an arbitrary distribution  $\mathcal{D}$ , finds*

with high probability a set  $S \subset \mathbb{R}^d$  that is  $\Gamma = \exp(O_{\gamma,\delta}(d^{-1/2+o(1)}))$  competitive, i.e.,

$$\mathbf{P}_{y \sim \mathcal{D}}[y \in S] \geq \delta,$$

and

$$\text{vol}(S)^{1/d} \leq \text{vol}(B^*)^{1/d} (1 + O(d^{-1/2+o(1)}(\gamma\delta)^{-1})).$$

where  $B^*$  is the minimum volume ball that achieves  $\geq \delta + \gamma + O(\sqrt{d^2/n})$  coverage over  $\mathcal{D}$ .

The above is an immediate consequence of the following theorem which is for the empirical version of the problem, using standard concentration tools [Devroye and Lugosi, 2001] by incurring an additive  $O(\sqrt{d^2/n})$  term in the coverage probability (since the VC-dimension of  $d$ -dimensional ellipsoids is at most  $d^2 + d$ ).

**Theorem 4.2.2** (Finding small volume ellipsoid for  $n$  points). *Suppose we are given a set of points  $Y \subseteq \mathbb{R}^d$ ,  $|Y| = n$ , and  $0 \leq \delta \leq 1, 0 \leq \gamma \leq 1$ , such that there exists a subset  $Y^* \subseteq Y$ ,  $|Y^*| \geq \delta|Y|$ , that is contained in an unknown ball  $B^* = B(c^*, R^*)$ . Then we can find an ellipsoid  $\hat{E}$  such that  $|\hat{E} \cap Y| \geq \delta(1 - \gamma)|Y|$ , and*

$$\text{vol}(\hat{E})^{1/d} \leq \text{vol}(B^*)^{1/d} \cdot (1 + O(d^{-1/2+o(1)}/\gamma\delta)).$$

*Proof of Theorem 4.2.2.* We begin by finding a list of coarse guesses for  $B^*$ , as in Theorem 4.3.1. We create a list of balls

$$\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\},$$

(step 1 of Algorithm DENSE-ELLIPSOID). Note that  $|\mathfrak{B}| \leq n^2$  since  $|Y| = n$ . Consider the

---

**Algorithm 4.3** Algorithm DENSE\_ELLIPSOID for finding a small volume ellipsoid that contains at least  $\delta' = \delta(1 - \gamma)$  fraction of points

---

**Input:** a set of point  $Y \in \mathbb{R}^d$ , target fraction  $\delta \in (0, 1)$ , and slack  $\gamma \in (0, 1)$

**Output:** an ellipsoid  $\widehat{E} \subset \mathbb{R}^d$

1. Create a list of coarse balls  $\mathfrak{B} = \{B(y_1, \|y_1 - y_2\|) \mid y_1, y_2 \in Y\}$  and remove from  $\mathfrak{B}$  all balls that contains less than  $\delta n$  points in  $Y$ .
  2. Create an ellipsoid  $\widehat{E}_i$  for each coarse ball  $B'_i$  in the list  $\mathfrak{B}$  as follows:
    - (a) Set  $\widehat{\tau} = d^{1/4}$  and  $R'_i$  be the radius of  $B'_i$ .
    - (b) Compute the eigenvalues  $\lambda_1, \dots, \lambda_d$  and eigenvectors  $v_1, \dots, v_d$  of the covariance  $\Sigma_{Y'_i}$  of points  $Y'_i = Y \cap B'_i$ .
    - (c) Create an ellipsoidal shape  $\widehat{M}_i = \sum_{j=1}^d a_j^2 v_j v_j^\top$ , where  $a_j^2 = d$  if  $\lambda_j \geq \widehat{\tau}^2 (R'_i)^2 / d$  and  $a_i^2 = 1$  if  $\lambda_j < \widehat{\tau}^2 (R'_i)^2 / d$ .
    - (d) Compute the transformed points  $Z_i = \widehat{M}_i^{-1/2} Y$ . Find a ball  $\widehat{B}_i$  using subroutine from [Lemma 4.3.5](#) for  $q = 0$ ,  $\lambda_{(q+1)}(\Sigma_{Y'}) \leq \sigma^2$ ,  $\sigma^2 = \widehat{\tau}^2 (R')^2 / d$ , slack factor  $\gamma$ .
    - (e) Set  $\widehat{E}_i = \widehat{M}_i^{1/2} \widehat{B}_i$ .
  3. Return the ellipsoid  $\widehat{E}$  with the smallest volume in the list  $\{\widehat{E}_i\}$ .
- 

subset  $Y^\star \subseteq Y$  contained in ball  $B^\star = B(c^\star, R^\star)$ . Choosing

$$y_1^\star, y_2^\star = \operatorname{argmax}_{y_1, y_2 \in Y^\star} \|y_1 - y_2\|,$$

gives us that the ball

$$B' = B(y_1^\star, \|y_1^\star - y_2^\star\|)$$

contains all points in  $Y^\star$ . It has radius  $R' \leq 2R^\star$  and  $B' \in \mathfrak{B}$ . Let  $Y'$  be all points contained in this ball  $B'$ .

For a parameter  $\widehat{\tau}$  that we will set later, Algorithm DENSE\_ELLIPSOID chooses an ellipsoidal shape  $\widehat{M}$  to have the same eigenvectors as  $\Sigma_{Y'}$ , with eigenvalues  $a_i^2$  of  $\widehat{M}$  satisfying  $a_i^2 = d$  if  $\lambda_i \geq \widehat{\tau}^2 (R')^2 / d$  and  $a_i^2 = 1$  if  $\lambda_i < \widehat{\tau}^2 (R')^2 / d$ .  $\widehat{M}$  has the following nice properties.

- (i) The eigenvalues of  $\widehat{M}$  are in  $\{1, d\}$ . Thus  $\widehat{M} \succeq I$ , and in particular, for any set of points  $P$ ,

$$\text{vol}(\text{encball}(\widehat{M}^{-1/2}P)) \leq \text{vol}(\text{encball}(P)), \quad (4.7)$$

where  $\text{encball}(S)$  is the minimum volume ball enclosing  $S$ .

- (ii) Let  $\Sigma_{Y'}$  be the covariance matrix of  $Y'$ , and  $\lambda_1, \dots, \lambda_d$  be the eigenvalues of  $\Sigma_{Y'}$ . We show that

$$a_i^2 \geq \frac{d}{\widehat{\tau}^2(R')^2} \lambda_i \quad \text{for all } i. \quad (4.8)$$

When  $\lambda_i < \widehat{\tau}^2(R')^2/d$ , we set  $a_i^2 = 1$ , which satisfies (4.8). When  $\lambda_i \geq \widehat{\tau}^2(R')^2/d$ , we set  $a_i^2 = d$ . We know that since  $Y' \subseteq B'$ , every eigenvalue of  $\Sigma_{Y'}$  is upper bounded as  $\lambda_i \leq (R')^2$ . Thus  $(d/\widehat{\tau}^2(R')^2)\lambda_i \leq d/\widehat{\tau}^2 \leq d$ , satisfying (4.8). By definition, (4.8) implies that

$$\widehat{M} \succeq \frac{d}{\widehat{\tau}^2(R')^2} \Sigma_{Y'}.$$

In particular, this implies that the points in  $Y'$  have bounded variance in the transformed space  $\widehat{M}^{-1/2}(\mathbb{R}^d) = \{\widehat{M}^{-1/2}y : y \in \mathbb{R}^d\}$ ,

$$\widehat{M}^{-1/2} \Sigma_{Y'} \widehat{M}^{-1/2} \preceq O\left(\frac{\widehat{\tau}^2(R')^2}{d}\right) I_d. \quad (4.9)$$

- (iii) Since  $Y' \subseteq B'$ , the total variance of the points  $Y'$  is bounded as

$$\sum_i \lambda_i \leq (R')^2.$$

Since the  $\lambda_i$  are non-negative, this tell us that there can be at most  $d/\widehat{\tau}^2$  values of  $i$  for which  $\lambda_i \geq \widehat{\tau}^2(R')^2/d$ . Thus at most  $d/\widehat{\tau}^2$  eigenvalues of  $\widehat{M}$  are set to  $d$ .

In particular, this implies that for any shape  $S$ ,

$$\frac{\text{vol}(S)}{\text{vol}(\widehat{M}^{-1/2}S)} \leq d^{d/\widehat{\tau}^2} \leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right). \quad (4.10)$$

The algorithm transforms the points  $Y'$  by  $\widehat{M}^{-1/2}$ . We denote the points in the transformed space as  $T'$ ,

$$T' = \{\widehat{M}^{-1/2}y \mid y \in Y'\}.$$

Recall that by the choice of  $Y'$ ,  $Y^\star \subseteq Y'$ , and we denote

$$T^\star = \{\widehat{M}^{-1/2}y \mid y \in Y^\star\} \subseteq T'.$$

We apply the procedure from [Lemma 4.3.5](#) with  $q = 0$  to the minimum volume ball  $B'_T = B(c'_T, R'_T)$  containing  $T'$ . We have that  $\lambda_{\max}(\Sigma_{Y'}) \leq \sigma^2$ , where  $\sigma^2 = \widehat{\tau}^2(R')^2/d$  (due to (4.9)), and slack factor  $\gamma$ . Let  $B_T^\star = B(c_T^\star, R_T^\star)$  be the minimum volume ball containing  $T^\star$ . [Lemma 4.3.5](#) allows us to find a ball  $\widehat{B}_T = B(\widehat{c}_T, \widehat{R}_T)$  that contains a subset  $\widehat{T} \subseteq T'$ , where  $|\widehat{T}| \geq (1 - \gamma)|T^\star|$ , and

$$\widehat{R}_T \leq \sqrt{(R_T^\star)^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1 - \delta)}{\delta}\right)}. \quad (4.11)$$

([Lemma 4.3.5](#) has an additional parameter also named  $\tau$ , that is different from  $\widehat{\tau}$ , but this parameter does not apply here since we set  $q = 0$  and avoid the grid search element of the procedure entirely.)

Finally, we transform  $\widehat{B}_T$  back to the transformed space to get the candidate ellipsoid  $\widehat{E}$ ,

$$\widehat{E} = \widehat{M}^{1/2}\widehat{B}_T.$$



Note that  $|Y' \cap \widehat{E}| \geq (1 - \gamma)|Y^*|$  (since  $|\widehat{T}| \geq (1 - \gamma)|T^*|$ ). We can bound the volume of  $\widehat{E}$  as

$$\begin{aligned} \text{vol}(\widehat{E}) &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot \text{vol}(\widehat{B}_T) && \text{by (4.10)} \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot c_d \widehat{R}_T^d, \end{aligned}$$

where  $c_d$  is the volume of the unit ball in  $d$  dimensions. Thus,

$$\begin{aligned} \text{vol}(\widehat{E}) &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot c_d \left( (R_T^*)^2 + \sigma^2 \gamma \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right)^{d/2} && \text{by (4.11)} \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot c_d \left( (R^*)^2 + \frac{\sigma^2}{\gamma} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right)^{d/2} && R^* \geq R_T^* \text{ by (4.7)} \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot c_d (R^*)^d \left(1 + \frac{\sigma^2}{\gamma (R^*)^2} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right)^{d/2} \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot \exp\left(\frac{d}{2} \cdot \frac{\sigma^2}{\gamma (R^*)^2} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right) \cdot \text{vol}(B^*) \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2}\right) \cdot \exp\left(\frac{\widehat{\tau}^2 (R')^2}{2\gamma (R^*)^2} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right) \cdot \text{vol}(B^*) && \text{since } \sigma^2 = \frac{\widehat{\tau}^2 (R')^2}{d} \\ &\leq \exp\left(\frac{d \ln d}{\widehat{\tau}^2} + \widehat{\tau}^2 \cdot \frac{2}{\gamma} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right) \cdot \text{vol}(B^*) && \text{since } R' \leq 2R^* \\ &\leq \exp\left(d^{1/2} \left(\ln d + \frac{2}{\gamma} \left(1 + \frac{2(1 - \delta)}{\delta}\right) \right) \right) \cdot \text{vol}(B^*) && \text{setting } \widehat{\tau}^2 = d^{1/2} \\ &\leq \exp\left(d^{1/2+o(1)}(\gamma\delta)^{-1}\right) \cdot \text{vol}(B^*). \end{aligned}$$

Rephrasing the above bound in terms of  $\text{vol}(\widehat{E})^{1/d}$  gives the stated bound.  $\square$

## 4.5 Greedy Algorithm and Unions of Sets

In this section, we provide a greedy algorithm for constructing a union of sets from a given set system that achieves the desired coverage with a small volume compared to the minimum

---

**Algorithm 4.4** Algorithm GREEDY\_DENSITY for finding the small confidence set

---

**Input:** points  $Y \subseteq \mathbb{R}^d$  with  $|Y| = n$ , coverage level  $\delta \in [0, 1]$ , slack factor  $\gamma$ , number of sets  $k \in \mathbb{N}$ , black-box algorithm  $\mathcal{A}_\alpha$  (see description in [Theorem 4.5.1](#))

**Output:** confidence set  $\widehat{C} \subseteq \mathbb{R}^d$

1. Set  $t = 0$  and  $\widehat{C} = \emptyset$ ,  $Y_t = Y$ .
2. While  $|\widehat{C} \cap Y| \leq \delta n$ :
  - (a) For  $i \in \{0, \dots, \lceil \log_2 n \rceil\}$ , let  $S_i^{(t)} = \mathcal{A}_\alpha(Y_t, \frac{2^i}{n}, \gamma')$
  - (b) Let

$$S^{(t)} = \max_{S_i^{(t)}} \frac{|S_i^{(t)} \cap Y_t|}{\text{vol}(S_i^{(t)})}$$

such that  $|S_i^{(t)} \cap Y_t| \geq (1 - \gamma') \frac{\gamma}{4k} \cdot n$

- (c) Set  $\widehat{C} = \widehat{C} \cup S^{(t)}$ ,  $Y_{t+1} = Y_t \setminus S^{(t)}$ ,  $t = t + 1$

3. Output  $\widehat{C}$
- 

volume union of  $k$  sets.

**Theorem 4.5.1** (Greedy algorithm). *Let  $Y \subseteq \mathbb{R}^d$  be a set of  $n$  points, and  $\mathcal{C}$  be a set system over  $\mathbb{R}^d$  with bounded VC-dimension  $D$ ,  $k \in \mathbb{N}$ ,  $\delta \in (0, 1)$  be a coverage level, and  $\gamma \in (0, 1 - \delta)$  be a slack parameter in coverage. Let  $\mathcal{A}_\alpha(Y', \delta', \gamma')$  for approximation factor  $\alpha \geq 1$  be an algorithm, that given a set of points  $Y'$ , and a coverage level  $\delta'$  and a slack  $\gamma' \in (0, 1)$ , outputs a  $C \in \mathcal{C}$  such that  $|C \cap Y'| \geq (1 - \gamma')\delta|Y'|$ , and  $\text{vol}(C) \leq \alpha \cdot \text{vol}(C')$ , where  $C'$  is the minimum volume set in  $\mathcal{C}$  such that  $|C' \cap Y'| \geq \delta|Y'|$ .*

*Then, in  $\text{poly}(n, d, k)$  time and calls to  $\mathcal{A}_\alpha$ , Algorithm GREEDY\_DENSITY ([Algorithm 4.4](#)) outputs a set  $\widehat{C} \subseteq \mathbb{R}^d$  such that  $\widehat{C}$  is a union of  $O(\delta k / \gamma)$  sets from  $\mathcal{C}$ , and*

$$\text{vol}(\widehat{C}) \leq O\left(\frac{\alpha \log(k/\gamma)}{\gamma}\right) \cdot \text{vol}(C^*),$$

*where  $C^*$  is the minimum volume union of  $k$  sets from  $\mathcal{C}$  such that  $|C^* \cap Y| \geq (\delta + \gamma)n$ .*

*Proof.* Let  $C_1^*, \dots, C_k^* \in \mathcal{F}$  be the optimal  $k$  sets in  $\mathcal{F}$  that covers at least a  $\delta + \gamma$  fraction

of points in  $Y$ . Let  $C^* = C_1^* \cup \dots \cup C_k^*$ .

First we bound the number of iterations of the algorithm. For each iteration of the algorithm, we have that  $|\widehat{C} \cap Y| \leq \delta n$ . This means that  $|C^* \cap Y_t| \geq \gamma n$ , so there exists a  $C_\ell^* \in C^*$  such that  $|C_\ell^* \cap Y_t| \geq \frac{\gamma n}{k}$ . Thus, step (b) of our algorithm will successfully find a set in  $S^{(t)} \in \mathcal{C}$  such that  $|S^{(t)} \cap Y_t| \geq (1 - \gamma') \frac{\gamma}{4k} \cdot n$ . Setting  $\gamma' = \frac{1}{2}$  we have that the algorithm must terminate in

$$O\left(\frac{\delta}{\gamma} \cdot k\right)$$

iterations, and output a union of at most that many balls.

Now we bound the volume of the output set. We group the iterations into phases. In particular, phase  $j$  consists of iterations in which  $|\widehat{C} \cap Y| \in [(1 - \frac{1}{2^j})\delta n, (1 - \frac{1}{2^{j+1}})\delta n)$ .

[Claim 4.5.1](#) tells us that the volume of sets added in each phase is

$$O\left(\frac{\alpha}{\gamma}\right) \cdot \text{vol}(C^*).$$

Since each iteration, the new picked set  $S^{(t)}$  covers at least  $(1 - \gamma') \frac{\gamma}{4k} \cdot n$  points, there can be at most  $\log_2(\delta k / \gamma) \leq \log(k / \gamma)$  many phases. This means that the total volume of the sets output over all phases is bounded by

$$O\left(\frac{\alpha}{\gamma} \cdot \log(k / \gamma)\right) \cdot \text{vol}(C^*).$$

□

**Claim 4.5.1** (Marginal volume added in each phase is bounded). *In every phase  $j$ , the total*

marginal volume added is

$$\leq O(\alpha) \cdot (\text{vol}(C_1^*) + \cdots + \text{vol}(C_k^*)).$$

*Proof.* Let  $t_j$  be the first iteration in phase  $j$ , and  $t_{j+1} - 1$  be defined as the last iteration in phase  $j$ .

We first show that in every iteration  $t$ , one of the sets  $C_1^*, \dots, C_k^*$  is a feasible solution with good density. Note that at the beginning of each iteration, the confidence set covers at most  $\delta$  fraction of points in  $Y$ . Let  $C^* = C_1^* \cup \cdots \cup C_k^*$ . Since  $|C^* \cap Y| \geq (\delta + \gamma)|Y| = (\delta + \gamma)n$ , we have that  $|C^* \cap Y_t| \geq \gamma n$  for any iteration  $t$ .

[Claim 4.5.2](#) tells us that there is a set  $C_\ell^*$  that achieves individually high coverage and high density compared to  $C^*$  over  $Y_t$ . That is, there is an  $\ell \in [k]$  such that

$$|C_\ell^* \cap Y_t| \geq \frac{\gamma n}{2k} \quad \text{and} \quad \frac{|C_\ell^* \cap Y_t|}{\text{vol}(C_\ell^*)} \geq \frac{|C^* \cap Y_t|}{2 \cdot \text{vol}(C^*)}.$$

Since we are in phase  $j$ , we have that the number of points that we have covered thus far is  $|Y \setminus Y_t| \leq (1 - \frac{1}{2^{j+1}})\delta n$ , so  $|C^* \cap Y_t| \geq \frac{\delta n}{2^{j+1}}$ , so

$$\frac{|C_\ell^* \cap Y_t|}{\text{vol}(C_\ell^*)} \geq \frac{|C^* \cap Y_t|}{2 \cdot \text{vol}(C^*)} \geq \frac{\delta n}{2^{j+2} \cdot \text{vol}(C^*)}.$$

This means that, by [Claim 4.5.3](#), will choose a set  $C^{(t)} \in \mathcal{C}$  such that

$$|C^{(t)} \cap Y_t| \geq (1 - \gamma') \frac{\gamma n}{4k} \quad \text{and} \quad \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})} \geq \frac{|C_\ell^* \cap Y_t|}{\alpha \cdot \text{vol}(C_\ell^*)} \geq \frac{\delta n}{\alpha \cdot 2^{j+2} \cdot \text{vol}(C^*)}.$$

This implies that

$$\begin{aligned} \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})} &\geq \frac{\delta n}{\alpha \cdot 2^{j+2} \cdot \text{vol}(C^*)} \\ \frac{|C^{(t)} \cap Y_t| \cdot \alpha \cdot 2^{j+2} \cdot \text{vol}(C^*)}{\delta n} &\geq \text{vol}(C^{(t)}). \end{aligned}$$

Now we can aggregate over the iterations in phase  $j$ . We first consider all but the last iteration, that is  $t$  such that  $t_j \leq t \leq t_{j+1} - 2$ . (It can be the case that  $t_{j+1} - 2 < t_j$ , in which case there are no  $t$  in this category and the bound holds trivially.)

$$\left( \sum_{t=t_j}^{t_{j+1}-2} |C^{(t)} \cap Y_t| \right) \cdot \frac{\alpha \cdot 2^{j+2} \cdot \text{vol}(C^*)}{\delta n} \geq \sum_{t=t_j}^{t_{j+1}-2} \text{vol}(C^{(t)})$$

Now, since for all iterations  $t$  such that  $t_j \leq t \leq t_{j+1} - 2$ , we have that  $t + 1$  is still in phase  $j$ , we know that the chosen sets could not have covered more than  $\frac{1}{2^j}n$  points.

$$\begin{aligned} \frac{\delta n}{2^j} \cdot \frac{\alpha \cdot 2^{j+2} \cdot \text{vol}(C^*)}{\delta n} &\geq \sum_{t=t_j}^{t_{j+1}-2} \text{vol}(C^{(t)}) \\ O(\alpha) \cdot \text{vol}(C^*) &\geq \sum_{t=t_j}^{t_{j+1}-2} \text{vol}(C^{(t)}). \end{aligned}$$

Finally, for the iteration  $t = t_{j+1} - 1$ , we have that  $|C^{(t)} \cap Y_t| \leq n$ . So we get that

$$\begin{aligned} \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})} &\geq \frac{1}{2\alpha} \cdot \frac{|C_\ell^* \cap Y_t|}{\text{vol}(C_\ell^*)} && \text{by Claim 4.5.3} \\ \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})} &\geq \frac{1}{4\alpha} \cdot \frac{|C^* \cap Y_t|}{\text{vol}(C^*)} && \text{by Claim 4.5.2} \\ \frac{n}{\text{vol}(C^{(t)})} &\geq \frac{1}{4\alpha} \cdot \frac{\gamma n}{\text{vol}(C^*)} \\ \frac{4\alpha}{\gamma} \text{vol}(C^*) &\geq \text{vol}(C^{(t)}). \end{aligned}$$

Thus in total, over all days in phase  $j$ , we have that the total volume of sets chosen by our algorithm is at most

$$O\left(\frac{\alpha}{\gamma}\right) \cdot \text{vol}(C^*).$$

□

**Claim 4.5.2.** *Given  $a_1, a_2, \dots, a_k, b_1, b_2, \dots, b_k \geq 0$  such that  $\sum_{i=1}^k a_i \geq \beta$ , and  $\sum_{i=1}^k a_i / \sum_{i=1}^k b_i \geq \gamma$ . Then there exists  $i \in [k]$ , satisfying  $a_i \geq \beta/(2k)$  and  $a_i/b_i \geq \gamma/2$ .*

*Proof.* Let  $\beta' = \sum_{i=1}^k a_i$ . Suppose there is no  $a_i, b_i$  such that  $a_i \geq \beta'/(2k)$  and  $a_i/b_i \geq \gamma/2$ . Let  $B = \{i \in [k] : a_i \geq \beta'/(2k)\}$ . Then, we have

$$\sum_{i \in B} a_i = \sum_{i \in [k]} a_i - \sum_{i \in [k] \setminus B} a_i > \sum_{i \in [k]} a_i - k \cdot \frac{\beta'}{2k} = \frac{\beta'}{2}.$$

Moreover for all  $i \in B$  we have  $a_i/b_i < \gamma/2$  i.e.

$$\sum_{i \in B} b_i > \frac{2}{\gamma} \sum_{i \in B} a_i > \frac{2}{\gamma} \cdot \frac{\beta'}{2} = \frac{\beta'}{\gamma}.$$

But this contradicts  $\sum_{i \in [k]} b_i \leq \sum_{i=1}^k a_i / \gamma = \beta' / \gamma$ , which completes the proof. □

**Claim 4.5.3** (Each iteration finds approximate density maximizer). *In each iteration, our algorithm chooses a set  $S^{(t)}$  such that*

$$\frac{|S^{(t)} \cap Y_t|}{\text{vol}(S^{(t)})} \geq \frac{1}{2\alpha} \cdot \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})},$$

and

$$(1 - \gamma') \frac{\gamma}{4k} \cdot n \leq |S^{(t)} \cap Y_t|,$$

where  $C^{(t)}$  is the maximum density set such that  $\frac{\gamma}{2k}n \leq |C^{(t)} \cap Y_t|$ .

*Proof.* Fix an iteration  $t$ , and let  $C^{(t)}$  be the set in  $\mathcal{C}$  such that  $\frac{\gamma}{2k}n \leq |C^{(t)} \cap Y_t|$ , that maximizes

$$\frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})}.$$

Let  $i^* \in \{0, \dots, \lceil \log_2 n \rceil\}$  be the value that satisfies

$$2^{i^*} \leq |C^{(t)} \cap Y_t| \leq 2 \cdot 2^{i^*}.$$

Since  $C^{(t)}$  achieves coverage at least  $\frac{2^{i^*}}{n}$ , we have that the set  $S_{i^*}^{(t)}$  chosen in step (a) of the algorithm must have

$$\text{vol}(S_{i^*}^{(t)}) \leq \alpha \cdot \text{vol}(C^{(t)}) \quad \text{and} \quad |S_{i^*}^{(t)} \cap Y_t| \geq (1 - \gamma') \frac{2^{i^*}}{n}.$$

Thus, we have that

$$\frac{|S_{i^*}^{(t)} \cap Y_t|}{\text{vol}(S_{i^*}^{(t)})} \geq \frac{1}{2\alpha} \cdot \frac{|C^{(t)} \cap Y_t|}{\text{vol}(C^{(t)})},$$

and that

$$(1 - \gamma') \frac{\gamma}{4k} \cdot n \leq |S_{i^*}^{(t)} \cap Y_t|.$$

□

**Corollary 4.5.2** (Union of Balls). *We give an algorithm that, given a set of points  $Y \subseteq \mathbb{R}^d$ ,  $|Y| = n$ , a coverage fraction  $\delta \in (0, 1)$ , a slack parameter in coverage  $\gamma \in (0, 1 - \delta)$  and  $k \in \mathbb{N}$ , can find a set  $\widehat{C}$  such that*

$$\text{vol}(\widehat{C}) \leq \exp(O_\delta(d^{1/2+o(1)})) \cdot O\left(\frac{1}{\gamma} \cdot \log(k/\gamma)\right) \cdot \text{vol}(C^*),$$

and  $\widehat{C}$  is a union of  $O(\frac{\delta k}{\gamma})$  ellipsoids, where  $C^*$  is the minimum volume union of  $k$  balls that covers at least  $(\delta + \gamma)$  fraction of the points in  $Y$ .

**Corollary 4.5.3** (Proper Learning Union of balls). *Let  $\delta \in (0, 1), \gamma \in (0, 1), k \in \mathbb{N}$  be any constants. There is a polynomial time algorithm that for target coverage  $\delta \in (0, 1)$  and coverage slack  $\gamma \in (0, 1)$  when given  $n = \Omega(kd^2/\gamma^2)$  samples drawn i.i.d. from an arbitrary distribution  $\mathcal{D}$ , finds with high probability a set  $S \subset \mathbb{R}^d$  that is a union of balls, and is  $\Gamma = (1 + O_{\gamma, \delta}(\log \log d / \log d))$  competitive; more precisely, it satisfies  $\mathbf{P}_{y \sim \mathcal{D}}[y \in S] \geq \delta$ , and*

$$\text{vol}(S)^{1/d} \leq \text{vol}(C_k^*)^{1/d} \left(1 + O_{k, \delta}(\log \log d / \log d)\right) \cdot \left(\frac{O(\log(k/\gamma))}{\gamma}\right)^{1/d}$$

where  $C_k^*$  is the minimum volume union of  $k$ -balls that achieves at least  $\delta + \gamma + O(\sqrt{kd^2/n})$  coverage over  $\mathcal{D}$ .

## 4.6 Hardness of Proper Learning

### 4.6.1 NP-hardness of Proper Learning

In this section, we show the approximation hardness of finding the smallest volume ball that achieves the required coverage.

**Theorem 4.6.1** (NP-hardness of Proper Learning). *For any small constant  $\varepsilon > 0$  and there is some constant  $\delta \geq 1/4$ , unless  $P = NP$ , there is no algorithm that given a set of points  $Y \subseteq \mathbb{R}^d$  runs in polynomial time in  $|Y|, d$  that finds a ball  $\hat{B} = B(\hat{c}, \hat{R})$  that contains at least a  $\delta$  fraction of points in  $Y$  with radius  $\hat{R} \leq (1 + d^{-\varepsilon})R^*$ , where  $R^*$  is the radius of the smallest ball  $B^*$  that contains at least a  $\delta$  fraction points in  $Y$ .*

We remark that the above hardness result rules out polynomial time proper learning algorithms that work for arbitrary distributions— this is called “the distribution-free setting” in PAC learning (as opposed to the distribution-specific setting). More formally, one can consider the following reduction from the worst-case problem of finding the minimum volume



ball containing a  $\delta$  fraction of an adversarially chosen point set  $Y \subset \mathbb{R}^d$ , to our problem. We construct a distribution  $D_Y$  that is the uniform distribution over  $Y$ . By the construction of  $Y$ , finding a set that contains probability mass  $\delta$  over  $D_Y$  corresponds exactly to finding a set that contains at least a  $\delta$  fraction of  $Y$ . We can simulate running our algorithm on  $D_Y$  by sampling from  $D_Y$  and providing these samples to the algorithm. Thus our problem is only harder than the worst-case problem, because it only gets sample access to  $D_Y$ , rather than access to  $Y$  itself.

The above theorem involves a reduction from the maximum clique problem to the smallest  $k$ -enclosing ball due to [Shenmaier \[2015\]](#). We now proceed to the proof of [Theorem 4.6.1](#).

*Proof.* [Shenmaier \[2015\]](#) proves the strong NP-hardness of the smallest  $k$ -enclosing ball in Euclidean space through a reduction from the  $k$  clique problem on regular graphs. We utilize this construction in our reduction as follows. Let  $G$  be a  $\Delta$ -regular graph with  $n$  vertices and  $m$  edges. We construct a set of  $n$  points  $Y \subset \mathbb{R}^d$  with dimension  $d = m^{3/\varepsilon}$  for a small constant  $\varepsilon \in (0, 1)$ . Each point  $y \in Y$  corresponds to a vertex  $v$  in graph  $G$ . The first  $m$  coordinates of  $y$  are the  $m$ -dimensional row for vertex  $v$  in the incident matrix of  $G$ . Then, for  $i = 1, 2, \dots, m$ , we have  $y_i = 1$  if the edge  $i$  connects vertex  $v$ ; otherwise  $y_i = 0$ . The rest  $d - m$  coordinates of  $y$  have value 0. Note that this reduction takes time  $O(nd)$ , which is polynomial in  $m$  when  $\varepsilon$  is a small constant in  $(0, 1)$ .

We now show that this construction of  $Y$  provides the hardness of proper learning. Since the graph  $G$  is  $\Delta$  regular, we have the distance between two points in  $Y$  is  $\sqrt{2\Delta - 2}$  if the corresponding vertices are connected in  $G$ ; otherwise, the distance is  $\sqrt{2\Delta}$ . By Lemma 3 in [Shenmaier \[2015\]](#), we have if  $G$  has a  $k$  clique, then the smallest ball that contains at least  $k$  points in  $Y$  has a radius  $R^* \leq \sqrt{A_k}$  where  $A_k = (\Delta - 1)(1 - 1/k)$ ; otherwise, the smallest ball that contains  $k$  points in  $Y$  has a radius at least  $\sqrt{A_k + 2/k^2}$ . Note that

$\sqrt{A_k + 2/k^2} \geq \sqrt{A_k}(1 + 1/n^3) \geq \sqrt{A_k}(1 + 1/m^3)$ . The  $k$ -clique problem is NP-hard even for regular graphs [Fleischner, Sabidussi, and Sarvanov, 2010, Feige, 2003]. Fleischner, Sabidussi, and Sarvanov [2010] show that it is NP-hard to find the maximum independent set for planar three regular graphs. Note that for three regular graphs, the maximum independent set has a size of at least  $n/4$ . The complement of a regular graph is still a regular graph and the maximum independent set corresponds to the maximum clique in the complement graph. Thus, for some constant  $\delta \geq 1/4$ , given a regular graph  $G$  on  $n$  vertices that contains  $k = \delta n$  clique, it is NP-hard to find a  $k$  clique in  $G$ .

Since we pick  $d = m^{3/\varepsilon}$ , we have it is NP-hard to approximate the radius of the smallest ball that contains a  $\delta$  fraction of points in  $Y$  within a factor of

$$1 + \frac{1}{m^3} = 1 + \frac{1}{d^\varepsilon}.$$

Thus, it is NP-hard to find a ball that contains a  $\delta$  fraction of points in  $Y$  with a radius at most  $1 + \frac{1}{d^\varepsilon}$  times the radius  $R^*$  of the minimum ball that contains  $\delta$  fraction of  $Y$ .  $\square$

#### 4.6.2 Computational Intractability even with Slack in Coverage

We give different reduction to provide evidence of strong computational intractability even when we are allowed to violate the coverage by a constant factor. Our hardness result is assuming the Small Set Expansion (SSE) hypothesis of Raghavendra and Steurer [2010], which is closely related to the Unique Games Conjecture [Khot, 2002].

**Conjecture 4.6.2** (SSE hypothesis of Raghavendra and Steurer [2010], see e.g., Theorem IV.5 of Raghavendra et al. [2012]). *For any constant  $\eta \in (0, 1/2)$ , there is a constant  $\tau \in (0, 1)$  such that there is no polynomial time algorithm to distinguish between the following two cases given a graph  $G = (V, E)$  on  $n$  vertices with degree  $D$ :*

- *YES: Some subset  $S \subseteq V$  with  $|S| = \tau n$  satisfies that the induced subgraph on  $S$  is dense i.e., the number of edges going out of  $S$  is  $|E(S, V \setminus S)| \leq \eta D|S|$  edges.*
- *NO: Any set  $S \subseteq V$  with  $\eta\tau n \leq |S| \leq 2\tau n$  has most of the edges incident on it going outside i.e.,  $|E(S, V \setminus S)| \geq (1 - \eta)|S|D$ .*

We prove the following theorem.

**Theorem 4.6.3** (Computational Intractability with Slack in Coverage). *For any constant  $\gamma > 0$ , there exists a constant  $\delta \in (0, 1)$ , such that assuming the SSE hypothesis for any constant  $\varepsilon > 0$  there is no algorithm that given a set of points  $Y \subseteq \mathbb{R}^d$  runs in polynomial time and finds a ball  $\hat{B} = B(\hat{c}, \hat{R})$  that contains at least a  $\gamma\delta$  fraction of points in  $Y$  with radius  $\hat{R} \leq (1 + d^{-\varepsilon})R^*$ , where  $R^*$  is the radius of the smallest ball  $B^*$  that contains at least a  $\delta$  fraction points in  $Y$ .*

*Proof.* Set  $\eta := \min\{\frac{\gamma^2}{9}, \frac{1}{16}\}$ . Given a  $D$ -regular graph  $G = (V, E)$  of the SSE problem in [Conjecture 5.4.1](#) on  $n$  vertices with parameter  $\eta$ , we first construct an instance for ball coverage in  $\mathbb{R}^m$  where  $m = |E|$  as follows. Every edge corresponds to a coordinate, and every vertex corresponds to a point in  $\mathbb{R}^m$  that corresponds to its edge incidences i.e.,  $u_i(e) = 1$  if vertex  $i$  is an endpoint of edge  $e$ , and 0 otherwise. Note that  $\|u_i - u_j\|^2 = 2D - 2$  if  $(i, j) \in E$  and  $\|u_i - u_j\|^2 = 2D$  when  $(i, j) \notin E$ . Let  $k = \tau n$  where  $\tau \in (0, 1)$  is the parameter in [Conjecture 5.4.1](#). Let  $R^* = \sqrt{D - \frac{D}{k}}$ , and  $R' = \sqrt{D - \frac{D}{2k}}$ . We first prove that it is SSE-hard to distinguish between the following two cases:

- (YES) case when there exists a ball of radius  $R^*$  that contains at least  $\delta n$  points,
- (NO) case when every ball of radius at most  $R'$  has at most  $\gamma\delta n$  points inside it.

*Completeness argument (YES case):* In the YES case of SSE hypothesis, there exists a subset

$S \subset V$  of size  $k$  with  $|E(S, S)| \geq \frac{kD(1-\eta)}{2}$ . Consider a candidate center

$$c^* = \begin{cases} \frac{2}{k} & \text{if } (i, j) \in E(S, S) \\ 0 & \text{otherwise} \end{cases}.$$

Let  $S^* = \{i \in S : \deg_S(i) \geq (1 - \sqrt{\eta})D\}$ . Since  $|E(S, S^c)| \leq \eta kD$ , by Markov's inequality,  $|S^*| \geq (1 - \sqrt{\eta})|S| \geq 3/4 \cdot |S| = \delta n$  where  $\delta = 3\tau/4$ . We have  $S^* \subset \text{Ball}(c^*, R^*)$  since

$$\begin{aligned} \forall i \in S^*, \quad \|u_i - c^*\|_2^2 &= \sum_{e \in E} \left(u_i(e) - \frac{2}{k}\right)^2 = (D - \deg_S(i)) + \deg_S(i) \left(1 - \frac{2}{k}\right)^2 \\ &\quad + (|E(S, S)| - \deg_S(i)) \left(\frac{2}{k}\right)^2 \\ &= D - \deg_S(i) \left(\frac{4}{k} - \frac{4}{k^2}\right) + (|E(S, S)| - \deg_S(i)) \cdot \left(\frac{4}{k^2}\right) \\ &\leq D + \frac{kd}{2} \cdot \frac{4}{k^2} - \frac{4(1 - \sqrt{\eta})D}{k} \leq D - (1 - 2\sqrt{\eta}) \frac{2D}{k} \\ &\leq (R^*)^2, \text{ since } \eta \leq 1/16. \end{aligned}$$

*Soundness argument (NO case):* Suppose there exists a subset of points corresponding to vertices  $S' \subset V$  with  $|S'| = k'$  for  $3\gamma k/4 \leq k' \leq k$  that are contained in a ball of radius  $R'$  around a point  $c' \in \mathbb{R}^n$ . Note that  $|S'| \geq \eta\tau n$  and  $|S'| = k' \geq 3\gamma k/4 = \gamma\delta n$ . We have

$$\begin{aligned} (R')^2 &= \max_{i \in S'} \|u_i - c'\|^2 \geq \text{Avg}_{i \in S'} \|u_i - c'\|^2 \geq \text{Avg}_{i \in S'} \left\| u_i - \text{Avg}_{i \in S'} u_i \right\|^2 \\ &= \frac{1}{2} \text{Avg}_{i, j \in S'} \|u_i - u_j\|^2 = D - \frac{2|E(S', S')|}{|S'|(|S'| - 1)}. \end{aligned}$$

Since  $3\gamma k/4 \leq |S'| \leq k$ , we have

$$\begin{aligned} D - \frac{2|E(S', S')|}{k'(k' - 1)} &\leq (R')^2 \leq D - \frac{D}{2k} \\ |E(S', S')| &\geq \frac{D}{4k} \cdot k'(k' - 1) \geq \frac{3\gamma(3\gamma k/4 - 1)D}{16} \geq \frac{\gamma^2}{8}kd > \eta D|S'|, \end{aligned}$$

since  $\gamma^2 > 8\eta$ . This contradicts the NO case of the SSE conjecture. This finishes the soundness argument. Thus, we have established the hardness of approximating the minimum radius ball containing  $\delta n$  points, even when we are allowed an arbitrary constant slackness factor  $\gamma > 0$  in the coverage. It is hard to approximate the radius within a factor of

$$\frac{R'}{R^*} = \sqrt{\frac{D - D/(2k)}{D - D/k}} \geq \sqrt{1 + \frac{1}{2k}} \geq 1 + \frac{1}{4m}.$$

Now as in [Theorem 4.6.1](#) we can pad the instance with dummy coordinates to make it a  $d$  dimensional instance with  $d = (4m)^{1/\varepsilon}$  to get the desired  $1 + d^{-\varepsilon}$  inapproximability factor in radius.  $\square$

## 4.7 Application to Conformal Prediction

As an immediate application of our result, we obtain an algorithm and guarantee for conformal prediction with approximate volume optimality in the high-dimensional setting.

We get the following sample theorem when the class  $\mathcal{C}$  is the set of Euclidean balls. (Equivalent statements go through for the other settings we consider in this work: properly learning Euclidean balls, and properly/improperly competing with unions of balls. All of the cases are simple applications of our meta-theorem for volume-optimal conformal prediction, [Theorem 2.5.1](#).)

**Theorem 4.1.6** (Conformal Prediction with Approximate Volume Optimality). *We have an algorithm for conformal prediction over examples from  $\mathcal{Y} = \mathbb{R}^d$ , that achieves approximate restricted volume optimality (Problem 2.2.1) with respect to the set  $\mathcal{B}$  of Euclidean balls. That is, we have an algorithm that, given training examples  $Y_1, \dots, Y_n \in \mathcal{Y}$  that are exchangeable with an unseen test example  $Y_{n+1} \in \mathcal{Y}$ , miscoverage rate  $0 < \alpha < 1$ , and coverage slack factor  $0 \leq \gamma \leq 1$ , outputs a set  $\hat{C}$  (not necessarily in  $\mathcal{B}$ ), such that*

(a)  $\hat{C}$  achieves the target coverage on  $Y_{n+1}$ ,

$$\mathbb{P}[Y_{n+1} \in \hat{C}] \geq 1 - \alpha. \quad (4.1)$$

(b) if  $n = \Omega(d^2/\gamma^2)$ , then  $\hat{C}$  achieves approximate volume optimality with respect to Euclidean balls,

$$\text{vol}(\hat{C})^{1/d} \leq (1 + O_{\gamma,\delta}(d^{-1/2+o(1)})) \text{vol}(C^*)^{1/d},$$

where

$$C^* = \underset{C \in \mathcal{B}}{\text{argmin}} \text{vol}(C) \quad \text{s.t.} \quad \mathbb{P}[Y_{n+1} \in C] \geq 1 - \alpha + \gamma.$$

*Proof.* Application of our meta-theorem for volume-optimal conformal prediction, Theorem 2.5.1 (see also Remark 2.5.2 about improper learning), using our subroutine to find approximately volume optimal confidence balls.  $\square$

## 4.8 Chapter Acknowledgements

This chapter is based on joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan. We thank an anonymous reviewer for helpful pointers to previous work involving core-sets.

## Chapter 5

# Competing with High-dimensional Ellipsoids

This chapter is based on *Learning Confidence Ellipsoids and Applications to Robust Subspace Recovery*, which is joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan [Gao, Shan, Srinivas, and Vijayaraghavan, 2026].

In this chapter, we develop computationally efficient algorithms to find prediction sets that satisfy approximate  $\mathcal{C}$ -restricted volume optimality [Problem 2.2.1](#), for  $\mathcal{C}$  the family of ellipsoids in  $\mathbb{R}^d$ . Our results have a mild dependence on the *condition number* of the ellipsoids that we compete with, which we show is necessary. We show that this problem is connected to the problem of robust subspace recovery, and we use our methods to design the first algorithm, to the best of our knowledge, for this problem in the worst-case setting.

## 5.1 Introduction

Among natural classes of confidence sets, ellipsoids occupy a privileged position. Confidence ellipsoids are a highly expressive family of confidence sets, as they naturally adapt to correlations among high-dimensional outputs and highlight directions of high or low uncertainty. They have been successfully applied in practical settings with structured outputs — for instance, in time-series forecasting, where prediction errors across timesteps are highly correlated [Messoudi, Destercke, and Rousseau, 2022, Xu, Jiang, and Xie, 2024]. Moreover, confidence ellipsoids are universal approximators of *all convex confidence sets* due to the famous John’s theorem [John, 1948].

Recall that an ellipsoid  $E$  is parameterized by a center  $c \in \mathbb{R}^d$  and a positive definite matrix  $M$  and given by  $E = \{x : \|M^{-1/2}(x - c)\|_2 \leq 1\}$ . The family of ellipsoids in  $\mathbb{R}^d$  has VC dimension  $\binom{d+1}{2} + d$ . Given any arbitrary distribution  $\mathcal{D}$ , we can draw  $n = O(d^2)$  i.i.d. samples from  $\mathcal{D}$ , and solve the following empirical version of the problem. For  $n$  arbitrary points  $a_1, a_2, \dots, a_n \in \mathbb{R}^d$ , find

$$\operatorname{argmin}_{E \in \mathcal{E}} \operatorname{vol}(E) \text{ s.t. } |E \cap \{a_1, \dots, a_n\}| \geq (1 - \alpha)n, \quad (5.1)$$

where  $\mathcal{E}$  is the family of ellipsoids in  $d$  dimensions. Any ellipsoid  $E$  that contains at least  $(1 - \alpha)n$  of the points also achieves a coverage of  $1 - \alpha - O(\sqrt{\frac{d^2}{n}})$  on  $\mathcal{D}$ , via uniform convergence for set families of bounded VC dimension. While there is a distributional flavor to the conformal prediction problem (Problem 2.2.1), which can be thought of as a population version of (5.1) (with  $\mathcal{C} = \mathcal{E}$ ), the distributional and empirical problems are actually equivalent. Given any arbitrary set of  $n$  points  $a_1, \dots, a_n \in \mathbb{R}^d$ , one can just consider  $\mathcal{D}$  to be the uniform probability distribution on these  $n$  points. This emphasizes the worst-case nature



of the problem, as the distribution  $\mathcal{D}$  is arbitrary. This is also crucial in many applications where the method is used to estimate uncertainty of an (unknown) arbitrary distribution.

Problem (5.1) has been an important object of study in several different communities. In statistics, this is the famous *minimum volume estimator* (MVE) of Rousseeuw [1985], that is studied as a robust non-parametric estimator for location and scatter parameters of a distribution. It is preferred for several desirable properties including affine equivariance (they behave well under affine transformations of the data), and having the *maximum breakdown value*<sup>1</sup> among all robust estimators of scatter and location (see the excellent survey by Van Aelst and Rousseeuw [2009] for details).

Solving this problem exactly is known to be NP-hard [Bernholt, 2006, Ahmadi, Malioutov, and Luss, 2014]. This is also been studied as the *robust minimum volume ellipsoid*, *minimum volume ellipsoid with outliers*, and *minimum  $k$ -enclosing ellipsoid problem* in optimization, applied mathematics and computational statistics, where many heuristics have been developed. These methods have been used as subroutines for applications in control theory, dynamical systems, and uncertainty quantification [Candela, 1996, Ahmadi, Malioutov, and Luss, 2014, Ahipasaoglu, 2014, Calafiore and Ghaoui, 2004, Vlahakis, Lindemann, Sopasakis, and Dimarogonas, 2024, Henderson, Mazoyer, and Gamboa, 2025].

However, despite several decades of work, to the best of our knowledge, there is no known polynomial time algorithm for learning small confidence ellipsoids.

**Question.** *Can we design polynomial time algorithms that learn confidence ellipsoids with rigorous volume guarantees?*

When  $\alpha = 0$ , this is the classic minimum volume enclosing ellipsoid problem (MVEE) which

---

<sup>1</sup>The breakdown value is the fraction of outliers the estimator can tolerate while providing a meaningful estimate.

can be solved in polynomial time using semi-definite programming [Boyd and Vandenberghe, 2004, Todd, 2016]. This is also the famous Löwner-John ellipsoid for the convex hull of the given points [John, 1948]. For general  $\alpha \in (0, 1)$ , in Chapter 4 [Gao, Shan, Srinivas, and Vijayaraghavan, 2025a] we study the special case of (5.1) for Euclidean balls, which is already known to be NP-hard. We give polynomial time algorithms that achieve volume guarantees that are competitive with that of the optimal solution up to a multiplicative factor, while relaxing the confidence parameter  $\alpha$  by a small amount.

The algorithmic problem (5.1) is significantly more challenging compared to balls. A special case is *robust subspace recovery* which is the computationally hard problem of finding a subspace of dimension at most a prescribed  $(1 - \gamma)d$  that contains an  $1 - \alpha$  fraction of points [Hardt and Moitra, 2013]. The NP-hardness of the robust subspace recovery problem directly implies that it is NP-hard to get *any non-trivial volume approximation* for (5.1) i.e., distinguish when the minimum volume ellipsoid containing a  $1 - \alpha$  fraction of points has  $\text{vol} = 0$  versus  $\text{vol} > 0$  (see Appendix C.1). However, this hard instance is pathological, where the optimal ellipsoid is ill-conditioned, with one of the axes lengths being 0 (or close to 0). This motivates restricting our attention to ellipsoids with bounded *condition number*.

Given an ellipsoid  $E = \{x : \|M^{-1/2}(x - c)\|_2 \leq 1\}$ , the condition number  $\beta(E) \in [1, \infty)$  is the ratio of maximum to minimum axes lengths:

$$\beta(E) := \frac{\max_{i \in [d]} \lambda_i(M^{1/2})}{\min_{i \in [d]} \lambda_i(M^{1/2})} = \sqrt{\frac{\max_{i \in [d]} \lambda_i(M)}{\min_{i \in [d]} \lambda_i(M)}}, \quad (5.2)$$

where  $\lambda_1(M), \dots, \lambda_d(M)$  are the eigenvalues of  $M$ . The condition number  $\beta(E) = 1$  for a Euclidean ball, and  $\beta \rightarrow \infty$  as the ellipsoid becomes supported on a proper subspace of  $\mathbb{R}^d$ . It will be instructive to think of  $\beta$  as being a large constant or polynomial in the dimension  $d$ . The main question that we study in this chapter is the following.

**Question 5.1.1** (Minimum  $\beta$ -Conditioned Confidence Ellipsoids). *Suppose there exists a  $\beta$ -conditioned ellipsoid  $E^*$  that contains  $(1 - \alpha)$  fraction of the input points  $a_1, \dots, a_n \in \mathbb{R}^d$ . Can we, in polynomial time, find an ellipsoid  $E$  containing an  $1 - \alpha'$  fraction of the points whose volume is  $\Gamma$ -competitive with  $E^*$  i.e.,*

$$\text{vol}(E)^{1/d} \leq \Gamma \cdot \text{vol}(E^*)^{1/d}, \quad (5.3)$$

where the missed coverage  $\alpha' = f(\alpha)$  goes to 0 as  $\alpha \rightarrow 0$ ?

*Remark 5.1.2.* Here the approximation factor  $\Gamma(\beta, d, \alpha)$  is a function of  $\beta, d, \alpha$  and is measured with respect to  $\text{vol}^{1/d}$  following the convention in [Chapter 4](#) [[Gao, Shan, Srinivas, and Vijayaraghavan, 2025a](#)], and the corresponding approximation factor for volume becomes  $\Gamma^d$ .

### 5.1.1 Results in this chapter

Our main result is a polynomial time algorithm that achieves a rigorous volume guarantee for ellipsoids whose condition number  $\beta$  is bounded.

**Theorem 5.1.3** (Algorithm for  $\beta$ -conditioned ellipsoids). *For any parameters  $\alpha \in (0, 1)$ ,  $\gamma \in (0, 1)$ ,  $\beta \geq 1$ , there exists a universal constant  $c > 0$  such that the following holds. There is an algorithm that given a set of  $n$  points  $A = \{a_1, \dots, a_n\} \in \mathbb{R}^d$  for which there exists an ellipsoid  $E^* \subset \mathbb{R}^d$  that is  $\beta$ -conditioned and  $|E^* \cap A| \geq (1 - \alpha)n$ , finds in polynomial time an ellipsoid  $\hat{E} \subset \mathbb{R}^d$  with  $\Gamma = \beta^\gamma$  i.e.,*

$$\text{vol}^{1/d}(\hat{E}) \leq (4\beta)^{2\gamma(1+\frac{1}{d})} \cdot \text{vol}^{1/d}(E^*), \text{ and encloses at least } |\hat{E} \cap A| \geq \left(1 - \frac{c\alpha}{\gamma}\right)n \text{ points.} \quad (5.4)$$

(See [Theorem 5.2.7](#) in [Section 5.2](#) for more details.) This immediately implies a corresponding guarantee for the distributional version, using the VC dimension bound on ellipsoids.

**Corollary 5.1.4** (Distributional version of [Theorem 5.1.3](#)). *For any parameters  $\alpha \in (0, 1)$ ,*

and  $\gamma \in (0, 1), \beta \geq 1$ , there is an algorithm that given  $n$  i.i.d. samples from an arbitrary distribution  $\mathcal{D}$  over  $\mathbb{R}^n$  with an (unknown)  $\beta$ -conditioned ellipsoid  $E^* \subset \mathbb{R}^d$  satisfying  $\mathbb{P}_{\mathcal{D}}[E^*] \geq (1 - \alpha)$ , finds in polynomial time an ellipsoid  $\hat{E} \subset \mathbb{R}^d$  with

$$\text{vol}^{1/d}(\hat{E}) \leq (4\beta)^{2\gamma(1+\frac{1}{d})} \cdot \text{vol}^{1/d}(E^*), \text{ and } \mathbb{P}_{\mathcal{D}}[\hat{E}] \geq 1 - O\left(\frac{\alpha}{\gamma} + \sqrt{\frac{d^2}{n}}\right). \quad (5.5)$$

**Implication:**  $\Gamma = \beta^{o(1)}$  **approximation.** Consider the setting when  $\alpha = o(1)$  (or a very small constant). We can compare against an algorithm that uses as a black-box the guarantee given in [Chapter 4](#) [[Gao, Shan, Srinivas, and Vijayaraghavan, 2025a](#)] for Euclidean balls. Since the ellipsoid  $E^*$  is  $\beta$ -conditioned, there is also a Euclidean ball whose volume is within a  $\beta^d$  multiplicative factor of it. The result in [Chapter 4](#) computes an ellipsoid whose volume is within a  $2^{o(d)}$  factor to the volume of the best ball (while also incurring a small loss in coverage). This leads to a volume approximation factor of  $2^{o(d)}\beta^d$ . In other words, this achieves an approximation factor  $\Gamma \approx \beta$ .

In comparison, our algorithm achieves a guarantee of  $\Gamma = \beta^\gamma$  i.e., a volume approximation of  $\beta^{\gamma d}$ , with miscoverage  $\alpha/\gamma$ . This achieves a  $\Gamma = \beta^{o(1)}$  factor for any choice of  $\gamma = o(1)$ . For example, by setting  $\gamma = \sqrt{\alpha}$ , the ellipsoid encloses at least  $(1 - O(\sqrt{\alpha}))$  fraction of the points, and achieves  $\Gamma = \beta^{\sqrt{\alpha}d}$ . Alternatively, by setting  $\gamma = \omega(1/\log \beta)$ , we can achieve an approximation factor  $\Gamma = \exp(o(d))$ , which is independent of  $\beta$ .

As we will see later, the  $\Gamma = \beta^{o(1)}$  guarantee is key to getting an algorithm for robust subspace recovery. This gives an example of how this guarantee is qualitatively different and more challenging than the weaker  $\Gamma = \beta$  guarantee.

Moreover, our algorithm deviates significantly from the approach in [Chapter 4](#) which leverages a new connection to robust high-dimensional estimation. In contrast, we design and

analyze a natural iterative algorithm based on the rich primal-dual structure of the *minimum volume enclosing ellipsoid* problem, and the *D-optimal design* problem,<sup>2</sup> and serves as a potential subroutine for other robust estimation tasks.

Our next result shows that the dependence on  $\beta$  in the factor  $\Gamma$  (and the volume approximation) is unavoidable (up to a constant factor in the exponent), even when we allow for some slack in the missed coverage  $\alpha$ . This is assuming the well-studied *small set expansion* (SSE) conjecture of Raghavendra and Steurer [2010], which is closely related to the unique games conjecture [Khot, 2002]. In fact the hardness holds for  $\Gamma = \beta^{\Omega(1)}$ ; hence achieving a factor  $\Gamma = \beta^{o(1)}$  requires some loss in coverage.

**Informal Theorem 5.1.5** (SSE-hardness). *Assuming the Small-Set Expansion conjecture, there exist small constants  $\alpha, \delta > 0$  and  $\beta_0 \geq 1$  such that for any (sufficiently) large  $\beta \geq \beta_0$ , no polynomial-time algorithm when given as input  $n$  points with a  $\beta$ -conditioned ellipsoid containing a  $1 - \alpha$  fraction of them can achieve a factor  $\Gamma < \beta^{\delta - o(1)}$  with coverage  $1 - (1 + \delta)\alpha$ .*

See Theorem 5.4.2 in Section 5.4 for a formal statement. This hardness result is based on the hardness for robust subspace recovery due to Hardt and Moitra [2013], and already suggests a strong connection between the two problems.

## Robust Subspace Recovery and Other Applications

Our algorithm from Theorem 5.1.3 gives a new algorithmic tool for robust estimation problems in high-dimensions. We show that we can use our algorithm to obtain polynomial time algorithmic guarantees for the challenging *robust subspace recovery* problem. Recall that we are given as input  $n$  points  $a_1, \dots, a_n \in \mathbb{R}^d$  (which we can assume are unit vectors), an out-

---

<sup>2</sup>The dual to the minimum volume enclosing ellipsoid SDP is the well-studied *D-optimal design* problem or equivalently the *G-optimal design* problem in statistics and experimental design, that tries to weight the covariates of a linear regression problem to minimize the variances of the predictions [Todd, 2016]. See also [Nikolov, 2015, Nikolov and Singh, 2016] for the “integral” version of the problem and other interesting variants.

lier fraction  $\alpha \in (0, 1)$ , and a closeness parameter  $\varepsilon \in (0, 1)$ . The goal is to find a subspace  $S \subset \mathbb{R}^d$  (passing through the origin) of smallest possible dimension (or largest co-dimension) that is  $\varepsilon$ -close in  $\ell_2$  distance to at least  $(1 - \alpha)n$  points. This is a well-studied problem in statistics, with motivation as a variant of PCA that finds a low-dimensional subspace even in the presence of outliers [Lerman and Maunu, 2018, Hardt and Moitra, 2013]. Due to the NP-hardness of the problem [Khachiyan, 1995], previous works that provide polynomial time guarantees need to make structural assumptions on either the outliers, or the inliers (points on the subspace).

**Prior algorithmic results on robust subspace recovery.** The work of Hardt and Moitra [2013] assumes genericity (similar to a smoothed analysis assumption) on the outliers and gives elegant polynomial time algorithms when the dimension of the subspace  $k$  containing  $(1 - \alpha)n$  points satisfies  $k \leq d(1 - \alpha)$ . They complement their result with a hardness result in the *worst-case setting* when  $k \geq (1 - \alpha)d$  assuming the SSE conjecture, even in the noiseless setting where  $\varepsilon = 0$  [Hardt and Moitra, 2013]. The work of Bhaskara, Chen, Perreault, and Vijayaraghavan [2019] improved the condition on  $\alpha$  in the smoothed analysis setting, and also provided some noise tolerance  $\varepsilon \approx 1/\text{poly}(n)$ . More recent works based on sum-of-squares (SoS) relaxations assume distributional assumptions (e.g., Gaussianity or rotational invariance) on the points in the subspace [Bakshi and Kothari, 2021] with stronger noise tolerance. In contrast we study the worst-case version of the problem, and make no assumptions on either the inliers or outliers. We are unaware of any non-trivial algorithmic guarantee in the worst-case setting of the problem prior to our work. We show that the algorithm for learning small volume confidence ellipsoids can be used as a subroutine to provide algorithmic guarantees for robust subspace recovery in the worst-case.

**Informal Theorem 5.1.6.** *For any  $\gamma \in (0, 1)$ , there is a polynomial time algorithm, that given as input points  $A = \{a_1, \dots, a_n\} \in \mathbb{R}^d$  with a subspace  $S^*$  of dimension  $(1 - 8\gamma)d$  that*

is  $\varepsilon_\star \leq \varepsilon^{4/\gamma}$  close to at least  $(1 - \alpha)n$  of the given points, finds w.h.p. a subspace  $S \subset \mathbb{R}^d$  of dimension at most  $(1 - \gamma)d$  such that at least  $(1 - O(\alpha/\gamma))n$  points are  $\varepsilon$  close to  $S$ .

See [Theorem 5.3.1](#) for the formal statement. When we know that there are many points that lie exactly on the subspace  $S$ , we have  $\varepsilon_\star = 0$ . We can pick  $\varepsilon$  to be whatever desired accuracy (that can be arbitrarily close to 0) and apply the above algorithm to find a subspace that is  $\varepsilon$ -close to the desired number of points. On the other hand, we also get guarantees when the desired accuracy  $\varepsilon$  is a small constant (for e.g.,  $\gamma$  is a constant and  $\varepsilon_\star$  is a sufficiently small constant) or  $\varepsilon$  is inverse polynomial in  $d^\gamma$  (when  $\varepsilon_\star \leq 1/\text{poly}(d)$ ).

To use this approach, it is crucial that we use an algorithm that can approximate a confidence ellipsoid up to a factor better than  $\Gamma = \beta$ . Our algorithm uses the fact that the points that lie close to the true underlying subspace must lie in a thin slab. Thus, they also lie in an ellipsoid that is thin in many directions, corresponding to the co-dimension of the subspace. Our algorithm applies a small random perturbation to all of the points, ensuring that we can control the width of this ellipsoid in all of the thin directions. Then, we argue that a guarantee of  $\Gamma = \beta^\gamma$  (i.e.  $\beta^{\gamma d}$  approximation in volume) to the ellipsoid can only blow up  $\approx \gamma d$  of these directions. To recover subspaces of nontrivial dimension  $< d$ , we use that  $\gamma d \ll d$  so we do not blow up too many directions. This gives an example why the  $\Gamma = \beta^{o(1)}$  guarantee is qualitatively different and more challenging than the trivial  $\Gamma = \beta$  guarantee. To the best of our knowledge, this provides the first rigorous polynomial time guarantee for robust subspace recovery in the worst-case setting.

**Statistical Applications.** Our algorithm for learning confidence ellipsoids has some direct applications to other related problems. One implication of our algorithm is that it gives an algorithm for learning *convex confidence sets* with bounded condition number that are of approximately minimum volume ([Theorem 5.5.2](#)). This is because any convex set can be

well-approximated by an ellipsoid, and our algorithm can provide a good approximation to the ellipsoid. An interesting aspect of this guarantee is that the family of convex sets has infinite VC dimension. We know that bounded VC dimension is a sufficient condition for a family of confidence sets to be statistically efficiently learnable. However, our result suggests that bounded VC dimension is not a necessary condition for learning an approximately volume optimal confidence set in the family.

Another implication of our result is for high-dimensional conformal prediction ([Theorem 5.5.4](#)). In conformal prediction, the goal is to use training examples to construct a prediction set that contains an unseen test example with probability at least  $1 - \alpha$ . An algorithm for finding small confidence sets on arbitrary samples directly implies an efficient conformal prediction algorithm. Please see [Section 5.5](#) for details.

### 5.1.2 Technical Overview

#### Approximate Minimum Volume Confidence Ellipsoid

Given a set of points  $A = \{a_1, \dots, a_n\} \subseteq \mathbb{R}^d$ , our goal is to find a small ellipsoid that contains  $(1 - \alpha)$ -fraction of the points in  $A$ . Let  $E^*$  be the optimal ellipsoid that encloses  $(1 - \alpha)n$  points of  $A$ , and has condition number at most  $\beta$ . We will refer to the points enclosed by  $E^*$  as the *inliers*, and the  $\alpha n$  points of  $A$  that are not enclosed by  $E^*$  as the *outliers*.

**Origin-centered setting and SDP formulation.** For the purpose of the technical overview, we will focus on the *origin-centered setting*, when the center of the ellipsoid is fixed to be the origin. This special case is already challenging and well-motivated, as it captures the application to the robust subspace recovery problem. Moreover it will inspire the main ideas in the algorithm and analysis — the general setting where the center is not the origin is handled by solving an appropriate origin-centered problem in  $d + 1$  dimensions.



In [Section 5.2](#), we present the full algorithm for the general setting where the center of the ellipsoid can be arbitrary. In the rest of this overview, we will present the main algorithmic ideas, and give a proof sketch of the entire analysis.

For the classic problem of finding the *minimum volume enclosing ellipsoid* (origin-centered) containing *all* of the points in  $A$ , then there is a very well-studied formulation of this problem as a semidefinite program (SDP):

$$\begin{aligned} & \max_Q \log \det Q \\ & \text{s.t. } \forall i, \ a_i^\top Q a_i \leq 1 \\ & \quad Q \succeq 0. \end{aligned} \tag{MVEE-OC}$$

Note that the volume of the resulting ellipsoid  $E = \{x : x^\top Q x \leq 1\}$  is  $\kappa_d \det(Q^{-1/2})$ , where  $\kappa_d > 0$  is the volume of the unit radius ball in  $d$ -dimensions. For the purposes of this chapter, we can ignore this fixed constant  $\kappa_d$ , as all the  $d$ -dimensional volumes will be scaled by the same amount. This SDP can be solved in polynomial time up to desired accuracy  $\eta$  in polynomial time in  $d, n, \log(1/\eta)$ . The goal is now to choose  $\alpha n$  points of  $A$  to leave out, such that it brings down the objective value of the above SDP (MVEE-OC) as much as possible. The above SDP seems challenging to work with directly when we only need to enclose  $1 - \alpha$  of the distribution.

Which points contribute to increasing the volume of the enclosing ellipsoid the most? Intuitively, this is a problem that has to be addressed somehow by the SDP itself, as these points are the ones that the solution has to work hardest to cover. This motivates us to look at the dual formulation, which has a variable that captures how hard it is to satisfy the constraint from each point.

**Dual SDP and D-Optimal Design.** The dual of the minimum volume enclosing ellipsoid problem is the classic *D-optimal design* problem, which is (a scaled version of) the problem of finding a distribution supported on the points  $a_i$ , that maximizes the determinant of the covariance or second moment matrix (centered at the origin).

$$\begin{aligned}
& \max_{w_i, \forall i \in [n]} \log \det \left( \sum_{i=1}^n w_i a_i a_i^\top \right) \\
& \text{s.t. } \forall i, w_i \geq 0 \qquad \qquad \qquad (\text{dual-MVEE}) \\
& \qquad \sum_{i=1}^n w_i \leq d.
\end{aligned}$$

The weights  $w_i$  corresponding to the optimal solution of the dual can be shown to satisfy  $w_i \leq 1$ .

Intuitively, far-away outliers will contribute the most to increasing the volume of the enclosing ellipsoid. The dual SDP will likely place a significant fraction of weight on outliers that lie far outside the desired small ellipsoid. Another interpretation of this is via complementary slackness. The points that are assigned non-zero weight in the dual solution are exactly the points that lie on boundary of the optimal ellipsoid in the primal SDP solution. Deleting a point with  $w_i = 0$  i.e., one that is contained in the interior of the ellipsoid cannot, by itself, bring down the volume of the enclosing ellipsoid. Hence the weights  $(w_i : i \in [n])$  can be interpreted as a measure of how important the point is to the volume of the enclosing ellipsoid.

**Iterative algorithm based on dual and progress in outlier removal.** The above intuition suggests the following natural iterative algorithm. Our algorithm proceeds by iteratively solving the dual program (dual-MVEE) on the set of points  $A = (a_i : i \in [n])$ . This gives us weights  $(w_i : i \in [n])$ , which are essentially a distribution (the weights sum

to  $d$  instead of 1). We remove each point  $a_i$  from the dataset with probability proportional to  $w_i$ . If significant fraction of the weight ( $w_i : i \in [n]$ ) is supported on the outlier points, then this strategy makes progress in removing many outliers. We think of this in terms of a tunable parameter  $\gamma > 0$ . If in some iteration, the dual solution places at least  $\gamma d$  (out of the total  $d$ ) weight on outlier points, then in expectation, out of the  $d$  discarded points at least  $\gamma \gg \alpha$  fraction of it will be outliers. Hence in every iteration, as long as the dual program (dual-MVEE) puts  $\geq \gamma d$  mass on the outliers, we will continue making progress by decreasing the fraction of outliers in the remaining points, while throwing out approximately a  $\gamma/\alpha$  factor more points than the target  $\alpha n$ .

**Bounding the volume approximation.** Eventually, once we have thrown out many outliers, on some iteration the dual will place less than  $\gamma d$  weight on outlier points. In this case, the dual solution ( $w_i : i \in [n]$ ) is mostly supported on the inlier points i.e., on the boundary of the optimal solution, most of the points, by weight, are inliers. The main challenge is to prove that the volume of the enclosing ellipsoid in this iteration is a good approximation of the volume of  $E^*$ .

This is particularly tricky, because the SDP that we solve at this iteration is *not a relaxation* of enclosing only the inliers. We must instead tie the cost of the SDP on this instance to the cost on a different instance that contains only the inliers. This is also challenging because we do not make any assumptions on the distribution of the inliers — recall that the inliers are simply defined to be the points inside the optimal solution  $E^*$ . This is different from the standard setting in robust estimation, where the algorithm exploits strong properties of the inliers such as isotropicity or even higher-order moment conditions, which can be algorithmically certified.

A simple procedure allows us to assume that all of the points  $a_i$  are within a bounding ball

of radius  $O(\beta)$ , and that the minimum axis length of  $E^*$  is at least 1.

We remark that this rescaling is done once at the start of the algorithm before performing the iterative outlier removal procedure. Essentially, we can throw away any points that are very far from the majority of points, since  $E^*$  has condition number  $\leq \beta$ , so it cannot stretch too far from the majority of inliers.

The following lemma is the crucial claim in the analysis and shows that the minimum enclosing ellipsoid in any iteration where the dual places at most  $o(d)$  weight on outliers has small volume competitive with the optimal solution  $E^*$  (that encloses only the inliers).

**Lemma 5.1.7** (Good volume approximation when total outlier weight is  $o(d)$ ). *Let  $A$  be a set of points  $a_1, \dots, a_n \in \mathbb{R}^d$  with  $\|a_i\| \leq \beta$ , and  $E^* = \{x : x^\top S^{-1}x \leq 1\}$  be an ellipsoid with  $I \preceq S \preceq O(\beta) \cdot I$ . If the optimal dual solution (dual-MVEE)  $(w_i : i \in [n])$  has total weight at most  $\gamma d$  on the outliers  $A \setminus E^*$ , then the volume of the ellipsoid  $E$  corresponding to this dual solution is upper bounded by*

$$\begin{aligned} \log \left( \frac{1}{\kappa_d} \text{vol}(E) \right) &= \log \det \left( \sum_{i=1}^n w_i a_i a_i^\top \right) \leq \log \det(S) + O(\gamma d \log \beta) \\ \implies \text{vol}(E) &\leq \text{vol}(E^*) \cdot \beta^{O(\gamma d)}. \end{aligned}$$

Here  $\kappa_d > 0$  is the volume of the unit ball of radius 1 in  $d$  dimensions.

(See [Lemma 5.2.6](#) for the formal version of this lemma in the general non-origin centered setting.)

To develop some intuition for this lemma, imagine that the weights  $w_i$  are supported on exactly  $d$  points i.e.,  $w_i = 1$  for exactly  $d$  out of the  $n$  points, and  $w_i = 0$  for all others. Then, if the total weight on the outliers is at most  $\gamma d$  as in the assumption of the lemma,

then there are at most  $\gamma d$  outlier points on the boundary of the enclosing ellipsoid. Any ellipsoid that encloses all of these  $d$  points in the support of  $w$  also encloses all the points. Hence we can upper bound the volume of the minimum enclosing ellipsoid of all of the  $n$  points is by constructing another ellipsoid that contains these  $d$  points. First, we contain the  $(1 - \gamma)d$  inlier points by using  $E^*$ . Then, there are at most  $\gamma d$  remaining outlier points that need to be included. For each outlier point, we will stretch  $E^*$  in the direction of the outlier point to include it. Since  $E^*$  has width at least 1 in every direction, and each outlier point is at distance  $O(\beta)$  away, we expand the volume of  $E^*$  by a factor at most  $O(\beta)$  for each outlier point that we stretch to include. Thus, in this ideal situation when the weights  $w$  from the optimal dual solution are supported on exactly  $d$  points, with total weight of  $\gamma d$  on the outliers, the volume of the enclosing ellipsoid is at most

$$\text{vol}(E) \leq \text{vol}(E^*) \cdot O(\beta^{\gamma d}).$$

**Challenge: spread outliers.** Of course, in general, the dual weights could be supported on many more than  $d$  points. This presents a challenge for the type of analysis sketched above, as that uses not only that there is at most  $\gamma d$  weight on outlier points, but also *crucially that it is concentrated in at most  $\gamma d$  directions*. When the outlier weight could be spread out in many directions, it can affect the value of the dual solution significantly more.

Consider an example where  $E^*$  has width  $\beta$  in the first  $d/2$  coordinate directions, and width 1 in the remaining  $d/2$  coordinate directions. Then, the volume of  $E^*$  is proportional to  $\beta^{d/2}$ . Suppose the outliers are spread out at distance  $\beta$  in the “thin” directions of  $E^*$ . Then, even

if the outliers have total weight at most  $\gamma d$ ,  $M = \sum_i w_i a_i a_i^\top$  could look like

$$M = \left[ \begin{array}{c|c} 2(1-\gamma)\beta^2 I_{d/2} & 0 \\ \hline 0 & 2\gamma\beta^2 I_{d/2} \end{array} \right],$$

$$\text{with } \det(M) = \beta^d 2^{d/2} (1-\gamma)^{d/4} \gamma^{d/4} = \beta^{d/2} \cdot (4\beta^2(1-\gamma)\gamma)^{d/4}.$$

Hence, its volume of the corresponding ellipsoid could be larger than  $\text{vol}(E^\star) = \beta^{d/2}$  by a factor of  $(\beta^2\gamma)^{d/4} \approx \beta^{\Omega(d)}$ . Thinking of  $\beta$  as a large constant, or even polynomially large in  $d$ , we would like the exponent in our approximation factor to be brought down by  $\gamma$  as it was in the previous example with only  $d$  points.

**Using optimality via complementary slackness.** The key claim that allows us to prove [Lemma 5.1.7](#) is that this bad situation when the  $\gamma d$  total weight on outliers is spread in  $\omega(\gamma d)$  directions, will actually never happen in an *optimal* dual solution! Specifically, in the above example, the dual would actually be able to achieve a much larger determinant by putting  $\approx d/2$  weight on the inliers and  $\approx d/2$  weight on the outliers. This is good, because then we will make progress in removing outliers.

More generally, we will use the *complementary slackness* or KKT conditions for the optimality of the solutions to (MVEE-OC) and (dual-MVEE). Moreover, we will use the following important geometric inequality that uses the log-concavity of the determinant function. This is a geometric form of the famous *Brascamp-Lieb* inequality due to Keith Ball that was used to prove certain reverse isoperimetric inequalities [[Ball, 1989, 1991](#)].

**Claim 5.1.1** (Geometric Brascamp-Lieb inequality). *For any PSD matrix  $P \in \mathbb{R}^{d \times d}$ , weights*

$w_i > 0$ , and unit vectors  $\|b_i\| = 1$  such that  $\sum_i w_i b_i b_i^\top = I_d$ , we have

$$\log \det(P) \leq \sum_i w_i \log(b_i^\top P b_i).$$

The geometric interpretation implication of the claim is the following: the volume of any ellipsoid can be upper bounded by an appropriate geometric mean of the projected lengths of the ellipsoid onto a set of unit vectors that are isotropic (or more generally form a “tight frame”). We now present a short proof of [Lemma 5.1.7](#) assuming [Claim 5.1.1](#). In [Section 5.2](#) we show how it yields the general version in [Lemma 5.2.6](#) that is needed for the analysis of the general algorithm.

**Proof of [Lemma 5.1.7](#) for the Origin-centered Setting.** Let  $w$  be the optimal solution to the dual program. Let  $M = \sum_i w_i a_i a_i^\top$ . Let  $E^* = \{x : x^\top S^{-1} x \leq 1\}$  be the optimal ellipsoid that encloses the inliers. Then, the volume of  $E^*$  is proportional to  $\det(S)$ .

To compare the volumes of  $E^*$  and  $E$  consider the PSD matrix  $P = M^{1/2} S^{-1} M^{1/2}$ . Note that  $S$  is full rank by assumption. We have

$$\log \det(P) = \log \det(M) + \log \det(S^{-1}).$$

Consider all points  $a_i$  with weight  $w_i > 0$  in the optimal dual solution. By the KKT condition (complementary slackness), we have for any such point  $a_i$  with  $w_i > 0$ ,  $a_i^\top M^{-1} a_i = 1$ . Let  $b_i := M^{-1/2} a_i$ . Then this implies that for all  $i$  such that  $w_i > 0$ , the transformed points have length  $\|b_i\|_2 = 1$  (i.e., they are on the boundary) and

$$\sum_{i:w_i>0} w_i b_i b_i^\top = M^{-1/2} \left( \sum_{i:w_i>0} w_i a_i a_i^\top \right) M^{-1/2} = I.$$

Hence the subset of points  $b_1, \dots, b_n$  with weights  $w_i > 0$  are isotropic i.e., they form a tight frame. By [Claim 5.1.1](#), we have

$$\log \det(P) \leq \sum_{i:w_i>0} w_i \log(b_i^\top P b_i) = \sum_{i:w_i>0} w_i \log(a_i^\top M^{-1/2} P M^{-1/2} a_i) = \sum_{i:w_i>0} w_i \log(a_i^\top S^{-1} a_i),$$

since  $S^{-1} = M^{-1/2} P M^{-1/2}$ .

We now split the right-hand side into inliers and outliers. For the inliers  $a_i \in E^\star$ , we have  $a_i^\top S^{-1} a_i \leq 1$ . For the outliers, we have  $\|a_i\| \leq \beta$  (from the assumptions of the lemma), and  $E^\star$  has condition number  $O(\beta)$ . Hence  $a_i^\top S^{-1} a_i \leq O(\beta)$  for all outliers. We are given that the total weight  $w$  on the outliers is at most  $\gamma d$ . Then, we have

$$\log \det(P) \leq \sum_{i:w_i>0} w_i \log(a_i^\top S^{-1} a_i) \leq \gamma d \cdot O(\log \beta).$$

Therefore, we bound the optimal dual solution value by

$$\log \det(M) = \log \det(S) + \log \det(P) \leq \log \det(S) + \gamma d \cdot O(\log \beta).$$

This completes the proof of [Lemma 5.1.7](#).

We remark that the above proof does not use optimality of the ellipsoid  $E^\star$  for the inliers. In particular, the above proof uses complementary slackness conditions for  $M$ , and not for  $S$ . Hence  $E^\star$  can be any ellipsoid of condition number  $\leq \beta$  containing  $(1 - \alpha)n$  points, and our algorithm is guaranteed to find an ellipsoid whose volume is competitive with it. In fact, the algorithm does not need the knowledge of  $\beta$ , and outputs an ellipsoid for which the guarantee holds for any choice of  $\beta$ .

We defer the proof for the general (non-origin-centered) setting, and analysis of the entire



algorithm to [Section 5.2](#).

## Robust Subspace Recovery

We present an algorithm for robust subspace recovery by leveraging our algorithm for finding the minimum volume confidence ellipsoid. The approach builds on the geometric intuition that if many points lie near a low-dimensional subspace, the smallest-volume ellipsoid covering them must be elongated along that subspace. We then apply a PCA-style procedure that selects the eigenspace corresponding to the large axes lengths of the ellipsoid as an approximation to the underlying subspace. This is related to the example that we discussed above where the outliers are spread out. Due to log-concavity of volume, the volume of the ellipsoid is smaller if we stretch the ellipsoid a lot but only in a few directions, as opposed to stretching it a little bit in many directions.

The main challenge is that the ellipsoid produced by our algorithm may not be well-conditioned. In some directions, it can have very small eigenvalues, which makes it difficult to bound the dimension of the recovered subspace. To address this, we introduce a random perturbation step. The algorithm adds small, independent Gaussian noise to each point, which enforces an anti-concentration property. This property ensures that in every direction, only a small fraction of points have very small projections. As a result, any ellipsoid that covers most points must be sufficiently “fat” in all directions (see [Lemma 5.3.2](#) for the technical claim). This ensures a lower bound on the smallest eigenvalues, allowing us to recover a low-dimensional subspace from the ellipsoid’s principal directions.

## Hardness

We establish the Small Set Expansion (SSE) hardness for our problem, using the robust subspace recovery problem as an intermediate step. Hardt and Moitra proved the SSE-

hardness of robust subspace recovery for low-dimensional subspaces ( $\delta d$ -dimensional), which shows that distinguishing whether there exists a low-dimensional subspace containing a certain fraction of points is SSE-hard [Hardt and Moitra, 2013]. We then connect this to our algorithmic result for subspace recovery in Section 5.3, which shows that if there exists a small-volume ellipsoid covering most of the points, then those points must lie close to some subspace.

There are two main challenges.

- (1) Our PCA-based argument introduces a small gap in the subspace dimension, while the hardness by Hardt and Moitra [2013] assumes the same dimension in both YES and NO cases. To bridge this, we modify their reduction to tolerate such a dimensional slack, yielding a strengthened hardness for robust subspace recovery with a small dimension gap.
- (2) After establishing that many points lie close to a low-dimensional subspace, we still need to argue that there exists a subspace that exactly contains a large fraction of these points.

To address this, we replace the randomized construction used by Hardt and Moitra [2013] with a deterministic construction of the robust subspace recovery instance. This discrete structure ensures that when many points are close to a subspace, we can identify a subspace of comparable dimension that exactly contains a large subset of them (Lemma 5.4.4), thereby completing the reduction and establishing the SSE-hardness of the minimum volume confidence ellipsoid problem.

### 5.1.3 Related Work

**Finding Small Volume Confidence Sets** The problem of learning minimum volume confidence sets has been widely studied in statistics and machine learning, due to applications in uncertainty quantification, learning density level sets, anomaly detection, or distribution support estimation [Einmahl and Mason, 1992, Polonik, 1997, Rousseeuw, 1985, Garcia, Kotalik, Cho, and Wolkenhauer, 2003, Scott and Nowak, 2005]. The works of Polonik [1999], Scott and Nowak [2005] design statistically efficient methods when the VC dimension of the class of confidence sets  $\mathcal{C}$  is bounded. This problem has also received recent attention due to efficiency or volume optimality considerations in conformal prediction and uncertainty quantification, see [Lei, Robins, and Wasserman, 2013, Sadinle, Lei, and Wasserman, 2016, Braun, Aolaritei, Jordan, and Bach, 2025] and also Chapter 3, Chapter 4, and Chapter 6 [Gao, Shan, Srinivas, and Vijayaraghavan, 2025b,a, Srinivas, 2026].

The most relevant prior work is the work on small volume learning confidence balls in high dimensions is our work in Chapter 4 [Gao, Shan, Srinivas, and Vijayaraghavan, 2025a]. The class  $\mathcal{C}$  of Euclidean balls in  $d$  dimensions is already computationally challenging for large  $d$ . In Chapter 4 we study both improper learning (where the confidence set that is output does not need to be from  $\mathcal{C}$ ) and proper learning algorithms, and designs polynomial time algorithms with rigorous guarantees of approximate volume optimality for  $\mathcal{C}$  being the set of Euclidean balls (or unions of  $O(1)$  balls). The best known polynomial time proper learning algorithm for Euclidean balls achieves  $\Gamma = 1 + \tilde{O}(1/\log d)$  (volume approximation of  $\exp(\tilde{O}(d/\log d))$ ) [Badoiu, Har-Peled, and Indyk, 2002], while the best improper learning algorithms for learning confidence balls outputs ellipsoids that achieve  $\Gamma = 1 + \tilde{O}(1/\sqrt{d})$  i.e., a volume approximation of  $\exp(\tilde{O}(\sqrt{d}))$  (Chapter 4). This is complemented by a computational intractability result for proper learning better than  $\Gamma = 1 + d^{-\varepsilon}$  for some constant  $\varepsilon > 0$ . The main technical idea is to design a robust proxy for computing the center, inspired by

algorithms for robust mean estimation.

These results do not give any guarantees when competing against the class  $\mathcal{C}$  of ellipsoids in  $d$  dimensions. For the learning confidence ellipsoids, the challenge is to learn both the center (location), and more importantly, the shape of a good ellipsoid for the distribution. It is an interesting open question to generalize our results to unions of ellipsoids, or use our new algorithm to get an improved improper algorithm for learning Euclidean balls.

**Comparison to Prior Work on Robust Estimation.** A related direction that has received much attention in both statistics and computer science is robust estimation, where the goal is to develop methods for parameter estimation that are robust to outliers. Here, the uncorrupted data points i.e., *the inliers* are drawn i.i.d from a nice distribution  $\mathcal{D}_\theta$  over  $\mathbb{R}^d$  with a parameter  $\theta$  that we would like to estimate, e.g., a Gaussian with an unknown mean, or unknown covariance. In the input dataset,  $1 - \alpha$  fraction of the samples are drawn from the inlier distribution, while the remaining  $\alpha$  fraction are *outliers*, which are arbitrary points in  $\mathbb{R}^d$  (stronger contamination models allow for replacing an arbitrary  $\alpha$  fraction of the inliers with arbitrary outliers). There is a rich body of work over the past six decades developing robust methods for several basic parameter estimation tasks under different contamination models [see e.g., extensive books on this topic [Huber, 2004](#), [Diakonikolas and Kane, 2023](#)]. In the previous chapter ([Chapter 4](#)), we even used ideas from robust estimation to find small confidence balls.

Most relevant to us are the works on robust covariance estimation, and robust estimators for the scatter of the distribution in high dimensions. The important works of [Diakonikolas, Kamath, Kane, Li, Moitra, and Stewart \[2016\]](#), [Lai, Rao, and Vempala \[2016\]](#) gave polynomial time algorithms for robust mean and covariance estimation of Gaussians in high dimensions, with near optimal dependencies. Over the past decade there have been several

polynomial time algorithms proposed based on filtering, sophisticated convex relaxations including sum-of-squares and many other techniques [please see [Diakonikolas and Kane, 2023](#), for a recent book on the topic]. In particular, for robust covariance estimation, these works assume Gaussianity of the inlier distribution, or various fourth moment or higher-order moment assumptions on the inlier distribution. Such assumptions are necessary to define a certifiable property (e.g., hypercontractivity, anticoncentration) of the inlier distribution in parameter estimation tasks. In contrast, in our setting we do not make any assumptions on the data points — in particular, the points in the optimal solution do not need to satisfy any Gaussianity or hypercontractivity or anticoncentration condition. Handling arbitrary distributions is important for applications in uncertainty quantification. Moreover, our goal is not to estimate or recover any ground-truth parameter, but instead to find some small volume confidence ellipsoid that contains most points.

The classic work of [Rousseeuw \[1985\]](#) studies the same problem as the famous *minimum volume estimator* (MVE) for location and scatter parameters of a distribution. See the survey by [Van Aelst and Rousseeuw \[2009\]](#) for several desirable properties including its high breakdown point and other statistical properties. As described earlier, this problem and heuristics for it have also found applications in control theory, dynamical systems, and uncertainty quantification. However, known algorithms for this problem are either computationally inefficient, or do not come with rigorous guarantees of (approximate) volume optimality.

**Robust Subspace Recovery.** Finally, the problem of robust subspace recovery in high dimensions has received significant attention on the algorithms side in the past decade starting with the influential work of [Hardt and Moitra \[2013\]](#). Polynomial time algorithms that recover a ground-truth subspace in the presence of outliers are only known under additional distributional assumptions about the inliers (points on the subspace) [[Bakshi and Kothari,](#)

2021], or non-degeneracy assumptions/ the smoothed analysis setting on the outliers [Hardt and Moitra, 2013, Bhaskara, Chen, Perreault, and Vijayaraghavan, 2019]. In particular, the work of Hardt and Moitra gives polynomial time algorithms that recover the ground truth subspace of dimension  $k$  when an  $\alpha < 1 - k/d$  fraction of outliers are in general position in the ambient space, and the  $1 - \alpha$  fraction of inliers are in general position in the subspace. One of their algorithms uses a beautiful connection to radial isotropy [Hardt and Moitra, 2013]. This uses a different convex program due to Barthe [1997], which is also related to a reverse version of the Brascamp-Lieb inequality see also [Hopkins, Kane, Lovett, and Mahajan, 2020, Diakonikolas, Tzamos, and Kane, 2022] for related work on the Forster transform and the radial isotropy position. Please see Section 5.3 for a more detailed comparison to prior work on robust subspace recovery.

## 5.2 Algorithm and Analysis

Let  $E^*$  be the smallest ellipsoid that contains at least  $(1 - \alpha)$ -fraction of the points in  $A = \{a_1, \dots, a_n\}$ . We begin by finding a bounding ball  $\widehat{B}$  that contains all points in  $E^*$ , and has radius at most twice that of the maximum axis length of  $E^*$ . This can be done by a simple search over all pairs of points in  $A$  (Lemma C.3.2). Restricted to  $\widehat{B}$ , it suffices to find an ellipsoid that captures close to  $(1 - \alpha)n$  of the embedded points, with volume approximately that of the smallest volume ellipsoid of condition number at most  $\beta$  that captures  $(1 - \alpha)n$  points.

We will refer to the  $(1 - \alpha)n$  points in  $A$  that are captured by the optimal  $E^*$  as *inliers*. We refer to the remaining  $\alpha n$  points in  $A$  as *outliers*.

Our algorithm proceeds by iteratively solving the dual program of the minimum volume ellipsoid (Proposition 5.2.4). This gives us weights, which are essentially a distribution

supported on the  $a_i$  (the weights sum to  $d$  instead of 1). We remove each point from the dataset with probability that is proportional to the weight it is assigned in the dual. For a tunable parameter  $0 < \gamma < 1$ , we show the following. In a given iteration, consider the following cases.

- (a) If more than  $\gamma d$  weight is placed on outliers, then when we remove points, more than a  $\gamma$  fraction of the points that are removed will be outliers, in expectation.
- (b) If less than  $\gamma d$  weight is placed on outliers, then the minimum volume ellipsoid enclosing all of the remaining points is already a good approximation in volume to the minimum volume ellipsoid enclosing only the inliers ([Lemma 5.2.6](#)).

Since we are only removing points in each iteration, the volume of the minimum volume enclosing ellipsoid of the remaining points can only decrease in each iteration. Thus if we iterate  $R \approx \frac{an}{\gamma d}$  times, we will have by the pigeonhole principle that on one of these iterations, less than  $\gamma d$  mass was placed on outliers, and therefore the volume approximation held on that and all following iterations. The fact that we make progress removing outliers be shown with Chebyshev's inequality, after observing that the point removal process forms a Martingale ([Lemma C.3.4](#)). We can also ensure that we do not remove too many points, inliers and outliers, over the  $R$  rounds via Chebyshev's inequality ([Lemma C.3.5](#)).

Finding the bounding ball, and part (a) of the outline above, which argues about the number of points removed, have proofs relegated to [Section C.3](#). In this section, we prove (b), which is the main technical contribution.

### 5.2.1 SDP and KKT condition

The main technical statement that we need to prove is about the form of the optimum of SDP for the minimum volume enclosing ellipsoid. We are interested in finding the minimum volume enclosing ellipsoid (MVEE) of *arbitrary center*. This can be found by the following convex program.

For a given set of points  $A$ , the minimum volume ellipsoid that contains *all* of the points  $A$  can be found via the following convex program.

**Proposition 5.2.1** (Minimum Volume Enclosing Ellipsoid, see e.g. [Sun and Freund \[2004\]](#), [Todd and Yildirim \[2007\]](#) ). *For a set of points  $a_1, \dots, a_n$ , the minimum volume ellipsoid enclosing the points is given by*

$$\{x : (x - c_\star)^\top S_\star (x - c_\star) \leq 1\}$$

for  $X = S_\star^{1/2}$  and  $z = S_\star^{1/2} c_\star$  be the optimum of the convex program

$$\begin{aligned} \min_{X \in \mathbb{R}^{d \times d}, z \in \mathbb{R}^d} & -2 \log \det X \\ \text{s.t. } & \|X a_i - z\|_2^2 \leq 1, \quad \forall i \in [n], \\ & X \succeq 0. \end{aligned}$$

The dual of this program is the following.

**Proposition 5.2.2** (Dual program of MVEE). *The dual of the program in [Proposition 5.2.1](#)*



is given by the convex program

$$\begin{aligned} \max_{w_i} \quad & \log \det \left( \sum_i w_i (a_i - c)(a_i - c)^\top \right) \\ \text{s.t.} \quad & w_i \geq 0, \quad \forall i \in [n] \\ & \sum_i w_i = d \\ & c = \frac{1}{d} \sum_i w_i a_i \end{aligned}$$

We refer to the value  $w_i$  as the weight assigned to  $a_i$ . While it is not given as an explicit constraint, it is without loss of generality that in the optimal solution, every  $w_i \leq 1$ . Furthermore, strong duality holds, and the optimal value of the primal program is equal to the optimal value of the dual program.

Another way to find the MVEE (minimum volume enclosing ellipsoid of arbitrary center) is to reduce to the problem of finding the minimum volume origin-centric ellipsoid. That is, for every point  $a_i \in \mathbb{R}^d$ , we identify it with the “lifted” point

$$\tilde{a}_i \in \mathbb{R}^{d+1} = [a_i, 1].$$

Let  $E^\star$  be the MVEE of the original  $a_i$ s. Let  $\tilde{E}$  be the minimum volume origin centric ellipsoid containing the  $\tilde{a}_i$ s. These have the nice property that  $E^\star$  is exactly the *slice* of  $\tilde{E}$  corresponding to the last index being 1,

$$E^\star = \tilde{E} \cap \{x \in \mathbb{R}^{d+1} : x_{d+1} = 1\}.$$

The minimum volume origin centric ellipsoid and its dual problem are given by the following SDPs. We note that the optimal dual solution for the MVEE problem on the  $a_i$ s ([Propo-](#)

sition 5.2.2), and the optimal dual solution for the origin centric minimum volume origin centric ellipsoid containing the  $\tilde{a}_i$ s are exactly the same. We provide a proof in Proposition C.3.6. To prove the volume approximation guarantee, we find it convenient to adopt the viewpoint of the origin centric dual.

**Proposition 5.2.3** (Minimum Volume Origin Centric Ellipsoid, see e.g. Boyd and Vandenberghe [2004]). *For a set of points  $a_1, \dots, a_n$ , the minimum volume centered ellipsoid enclosing the points is given by*

$$\{x : x^\top (M^\star)^{-1} x \leq 1\}$$

for  $(M^\star)^{-1} = Q$  the optimum of the SDP

$$\begin{aligned} & \max_{Q \in \mathbb{R}^{n \times n}} \log \det Q \\ & \text{s.t. } \forall i \in [n], \quad a_i^\top Q a_i \leq 1 \\ & \quad Q \succeq 0. \end{aligned}$$

**Proposition 5.2.4** (Dual program of origin centered MVE, see e.g. Boyd and Vandenberghe [2004]). *The dual of the program in Proposition 5.2.3 is given by the SDP*

$$\begin{aligned} & \max_{w_i} \log \det \left( \sum_i w_i a_i a_i^\top \right) \\ & \text{s.t. } \forall i \in [n], \quad w_i \geq 0 \\ & \quad \sum_i w_i \leq d. \end{aligned}$$

We refer to the value  $w_i$  as the weight assigned to  $a_i$ . While it is not given as an explicit constraint, it is without loss of generality that in the optimal solution, every  $w_i \leq 1$ . Furthermore, strong duality holds, and the optimal value of the primal program is equal to the

optimal value of the dual program.

*Proof of Proposition 5.2.2.* Let  $w_i \geq 0$  be Lagrange multipliers for the constraints  $\|Xa_i - z\|_2^2 \leq 1$ . The Lagrangian relaxation of MVEE is

$$\mathcal{L}(X, z, w) = -2 \log \det X + \sum_{i=1}^n w_i (\|Xa_i - z\|_2^2 - 1).$$

*Stationarity in  $z$ .* Taking the gradient in  $z$  and setting it to zero gives

$$0 = \nabla_z \mathcal{L} = 2 \sum_i w_i (z - Xa_i) \implies z = Xc, \quad c := \frac{\sum_i w_i a_i}{\sum_i w_i}.$$

Define the matrix

$$M := \sum_{i=1}^n w_i (a_i - c)(a_i - c)^\top \succeq 0.$$

With  $z = Xc$ , the quadratic term in the Lagrangian is

$$\sum_{i=1}^n w_i \|Xa_i - z\|_2^2 = \sum_{i=1}^n w_i \|X(a_i - c)\|_2^2 = \text{tr}(X^\top XM).$$

Let  $S = X^2$ . Using  $-2 \log \det X = -\log \det S$ , the Lagrangian becomes

$$\mathcal{L}(S, w) = -\log \det S + \text{tr}(SM) - \sum_i w_i.$$

*Stationarity in  $S$ .* Minimizing over  $S \succ 0$  yields the unique optimizer  $S = M^{-1}$ , and the minimum value

$$\min_{S \succ 0} \mathcal{L}(S, w) = \log \det M + d - \sum_i w_i.$$

Hence, the dual program of MVEE is

$$\begin{aligned} \max_{w \geq 0} \quad & \log \det M + d - \sum_i w_i \\ \text{s.t.} \quad & c = \frac{\sum_i w_i a_i}{\sum_i w_i}, \\ & M = \sum_{i=1}^n w_i (a_i - c)(a_i - c)^\top. \end{aligned}$$

Scaling all weights by a positive factor  $t$  leaves  $c$  unchanged and scales  $M$  by  $t$ . Thus, we have  $\log \det M = \log \det M + d \log t$  while  $-\sum_i t w_i = -t \sum_i w_i$ . Optimizing over  $t$  makes the total weight  $\sum_i w_i = d$  at optimum. Thus, we get the dual program.

*Strong duality.* The primal program MVEE has a strictly feasible solution. Therefore, strong duality holds and the optimal values of the primal and dual are equal.

*Bounds  $w_i \leq 1$  at optimum.* Let  $(X^*, z^*)$  be any optimal primal solution. Let  $w_i^*$  be the optimal dual solution and  $M^*$  be the corresponding matrix of  $w_i^*$ . Then, we have  $(M^*)^{-1} = (X^*)^2$ , which implies

$$I_d = X^* M^* X^* = \sum_i w_i (X^* (a_i - c))(X^* (a_i - c))^\top.$$

Let  $y_i = X^* (a_i - c)$ . By complementary slackness ([Proposition 5.2.5](#)), if  $w_i^* > 0$  then  $\|y_i\| = 1$ .

Thus, we have

$$1 = y_i^\top I_d y_i = \sum_j w_j (y_i^\top y_j)^2 \geq w_i (y_i^\top y_i)^2 = w_i,$$

so  $w_i \leq 1$  for all  $i$ . □

**Proposition 5.2.5** (Complementary Slackness). *Let  $(X^*, z^*)$  be any optimal primal solution to the MVEE problem in [Proposition 5.2.1](#), and let  $w^* = (w_1^*, \dots, w_n^*)$  be any optimal dual*

solution to [Proposition 5.2.2](#). Then, for every  $i \in [n]$ ,

$$w_i^* (\|X^* a_i - z^*\|_2^2 - 1) = 0.$$

In particular, if  $w_i^* > 0$  then  $\|X^* a_i - z^*\|_2^2 = 1$ , and if  $\|X^* a_i - z^*\|_2^2 < 1$  then  $w_i^* = 0$ .

*Proof.* Let  $(X^*, z^*)$  be any optimal primal solution and  $w^*$  be any optimal dual solution. The KKT complementary slackness conditions are

$$w_i^* (\|X^* a_i - z^*\|_2^2 - 1) = 0, \quad i = 1, \dots, n.$$

Hence, for any index with  $w_i^* > 0$ , we have  $\|X^* a_i - z^*\|_2^2 = 1$ . □

### 5.2.2 Volume Approximation

The main component of our proof is a statement that if the optimal dual solution to the origin centric minimum volume enclosing ellipsoid problem does not put much weight on outlier points, then the volume of the enclosing ellipsoid of all of the points is comparable to the smallest ellipsoid of bounded condition number, that contains all of the inliers. Thus, the outliers that are still included do not blow up the volume of the enclosing ellipsoid by too much.

The crux of the argument is in the following claim. This inequality is related to the general *Brascamp-Lieb inequality* [[Lieb, 1990](#)], and was used by [Ball \[1989\]](#) to prove certain reverse isoperimetric statements. We give a simple self-contained proof here for completeness.

**Claim 5.1.1** (Geometric Brascamp-Lieb inequality). *For any PSD matrix  $P \in \mathbb{R}^{d \times d}$ , weights*

$w_i > 0$ , and unit vectors  $\|b_i\| = 1$  such that  $\sum_i w_i b_i b_i^\top = I_d$ , we have

$$\log \det(P) \leq \sum_i w_i \log(b_i^\top P b_i).$$

*Proof.* Let  $P = U\Lambda U^\top$  be the SVD decomposition of  $P$ , where  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_{d+1})$ . Let  $c_i = U^\top b_i$ . Then, we have  $\|c_i\| = \|b_i\| = 1$  and  $\sum_i w_i c_i c_i^\top = \sum_i w_i b_i b_i^\top = I_{d+1}$ . For each  $i$ , we have

$$\log(b_i^\top P b_i) = \log(c_i^\top \Lambda c_i) = \log\left(\sum_{j=1}^{d+1} \lambda_j c_{ij}^2\right) \geq \sum_{j=1}^{d+1} c_{ij}^2 \log(\lambda_j),$$

where the last inequality is due to the concavity of the log function and Jensen's inequality.

Since  $\sum_i w_i c_i c_i^\top = I_{d+1}$ , for every  $j$ , we have  $\sum_i w_i c_{ij}^2 = 1$ . Thus, we have

$$\sum_i w_i \log(b_i^\top P b_i) \geq \sum_i w_i \sum_{j=1}^{d+1} c_{ij}^2 \log(\lambda_j) = \sum_{j=1}^{d+1} \log(\lambda_j) = \log \det(P).$$

□

We now bound the volume of the ellipsoid provided by our algorithm. We embed each point  $a_i \in \mathbb{R}^d$  to point  $\tilde{a}_i := (a_i, 1) \in \mathbb{R}^{d+1}$  by adding the  $(d+1)$ th coordinate with value 1. We then solve the origin-centered minimum volume ellipsoid for  $\tilde{a}_1, \dots, \tilde{a}_n \in \mathbb{R}^{d+1}$ . Let  $w$  be the optimal solution to the corresponding dual program in [Proposition 5.2.4](#). Then, we consider the ellipsoid  $E$  in  $\mathbb{R}^d$  as follows. Let center  $c = \sum_i w_i a_i / (d+1)$  and  $M = \sum_i w_i (a_i - c)(a_i - c)^\top$ . The ellipsoid  $E$  is  $\{x : (x - c)^\top M^{-1} (x - c) \leq \frac{d}{d+1}\}$ .

**Lemma 5.2.6** (Volume approximation). *Let  $a_1, \dots, a_n \in \mathbb{R}^d$  be points contained in a ball of radius  $2\beta$ , and  $E^* = \{x : (x - c_\star)^\top (S^*)^{-1} (x - c_\star) \leq 1\}$  be an ellipsoid with width at least 1 in every direction that encloses the inliers. If the optimal dual solution ([Proposition 5.2.4](#)) places at most  $\gamma(d+1)$  weight on outliers, then we have the shape matrix  $\overline{M} = \frac{d}{d+1} M$  of the*

ellipsoid  $E = \{x : (x - c)^\top (\overline{M})^{-1} (x - c) \leq 1\}$  satisfies

$$\log \det(\overline{M}) \leq \log \det(S^\star) + \gamma(d+1) \cdot \log(16\beta^2).$$

*Proof.* It is convenient to rescale the matrix  $(S^\star)^{-1}$  to a matrix  $S^{-1}$  such that

$$E^\star = \left\{ x : (x - c_\star)^\top S^{-1} (x - c_\star) \leq \frac{d}{d+1} \right\}.$$

We embed each point  $a_i \in \mathbb{R}^d$  to point  $\tilde{a}_i := (a_i ; 1) \in \mathbb{R}^{d+1}$ . Let  $w$  be the optimal solution to the origin centric dual program ([Proposition 5.2.4](#)) for the lifted instance in  $\mathbb{R}^{d+1}$ . Then, we have  $\sum_i w_i = d+1$ . and the lifted matrix  $\widehat{M} \in \mathbb{R}^{(d+1) \times (d+1)}$

$$\widehat{M} := \sum_{i=1}^n w_i \tilde{a}_i \tilde{a}_i^\top = \begin{pmatrix} \sum_i w_i a_i a_i^\top & \sum_i w_i a_i \\ \sum_i w_i a_i^\top & \sum_i w_i \end{pmatrix}.$$

Let  $c = \sum_{i=1}^n w_i a_i / (d+1)$  and  $M = \sum_{i=1}^n w_i (a_i - c)(a_i - c)^\top$ . We first show that for each lifted point  $\tilde{a}_i \in \mathbb{R}^{d+1}$  with  $\tilde{a}_i^\top \widehat{M}^{-1} \tilde{a}_i \leq 1$ , its corresponding point  $a_i \in \mathbb{R}^d$  is contained in the ellipsoid  $E = \{x : (x - c)^\top M^{-1} (x - c) \leq \frac{d}{d+1}\}$ . Note that  $M^{-1}$  is scaled so that the right-hand-side of the inequality is  $\frac{d}{d+1}$ , making it directly comparable to  $S^{-1}$ .

To show this, consider the Schur complement  $\widehat{M} / \widehat{M}_{d+1, d+1}$  ([Fact C.3.1](#)) of the  $(d+1, d+1)$  entry of  $\widehat{M}$ , which is the  $d \times d$  matrix

$$\widehat{M} / \widehat{M}_{d+1, d+1} = \sum_{i=1}^n w_i a_i a_i^\top - (d+1)c \cdot \frac{1}{d+1} \cdot (d+1)c^\top = \sum_{i=1}^n w_i (a_i - c)(a_i - c)^\top = M.$$

Then, the inverse of  $\widehat{M}$  is

$$\widehat{M}^{-1} = \begin{pmatrix} M^{-1} & -M^{-1}c \\ -c^\top M^{-1} & c^\top M^{-1}c + \frac{1}{d+1} \end{pmatrix}.$$

Hence, we have for every point  $\tilde{a}_i$

$$\tilde{a}_i \widehat{M}^{-1} \tilde{a}_i = (a_i - c)^\top M^{-1} (a_i - c) + \frac{1}{d+1},$$

which implies that  $a_i \in E$  for every  $\tilde{a}_i$  with  $\tilde{a}_i \widehat{M}^{-1} \tilde{a}_i \leq 1$ .

We now show that the log determinant of  $\widehat{M}$  is close to that of  $M$ . By the Schur complement, we have

$$\begin{aligned} \log \det \widehat{M} &= \log(d+1) + \log \det \left( \sum_{i=1}^n w_i a_i a_i^\top - \frac{1}{d+1} \cdot \left( \sum_{i=1}^n w_i a_i \right) \left( \sum_{i=1}^n w_i a_i^\top \right) \right) \\ &= \log(d+1) + \log \det \left( \sum_{i=1}^n w_i (a_i - c)(a_i - c)^\top \right) = \log(d+1) + \log \det M. \end{aligned}$$

Recall  $E^* = \{x : (x - c_\star)^\top S^{-1} (x - c_\star) \leq \frac{d}{d+1}\}$  is the ellipsoid with width at least 1 in every direction that encloses the inliers. Then, the volume of  $E^*$  is proportional to  $\det(S)$ . We define the following lifted matrix  $\widehat{S}^{-1} \in \mathbb{R}^{(d+1) \times (d+1)}$

$$\widehat{S}^{-1} := \begin{pmatrix} S^{-1} & -S^{-1}c_\star \\ -c_\star^\top S^{-1} & c_\star^\top S^{-1}c_\star + \frac{1}{d+1} \end{pmatrix}.$$

This lifted matrix corresponds to the shape of an origin-centered ellipsoid in  $\mathbb{R}^{d+1}$ . For each



embedded point  $\tilde{a}_i \in \mathbb{R}^{d+1}$ , we have

$$\tilde{a}_i^\top \hat{S}^{-1} \tilde{a}_i = (a_i - c_\star)^\top S^{-1} (a_i - c_\star) + \frac{1}{d+1}.$$

We then show that the log determinant of  $\hat{S}$  is close to that of  $S$ . By the Schur complement, we have

$$\log \det(\hat{S}^{-1}) = -\log(d+1) + \log \det(S^{-1}) = -\log(d+1) - \log \det S.$$

Let  $P = \widehat{M}^{1/2} \hat{S}^{-1} \widehat{M}^{1/2}$ . We have

$$\log \det P = \log \det \widehat{M} + \log \det(\hat{S}^{-1}) = \log \det M - \log \det S.$$

We then use the geometric Brascamp-Lieb inequality in [Claim 5.1.1](#) to bound  $\log \det P$ . Consider all points  $\tilde{a}_i$  with weight  $w_i > 0$  in the optimal dual solution. By the complementary slackness, we have  $\tilde{a}_i^\top \widehat{M}^{-1} \tilde{a}_i = 1$ . For any such point  $\tilde{a}_i \in \mathbb{R}^{d+1}$ , let  $b_i = \widehat{M}^{-1/2} \tilde{a}_i \in \mathbb{R}^{d+1}$ . Then, we have  $\|b_i\|_2 = 1$  and

$$\sum_{i:w_i>0} w_i b_i b_i^\top = \widehat{M}^{-1/2} \left( \sum_{i:w_i>0} w_i \tilde{a}_i \tilde{a}_i^\top \right) \widehat{M}^{-1/2} = I_{d+1}.$$

Thus, by [Claim 5.1.1](#), we have

$$\log \det P \leq \sum_i w_i \log(b_i^\top P b_i) = \sum_i w_i \log \left( \tilde{a}_i^\top \hat{S}^{-1} \tilde{a}_i \right),$$

since  $P = \widehat{M}^{1/2} S^{-1} \widehat{M}^{1/2}$  and  $b_i = \widehat{M}^{-1/2} \tilde{a}_i$ .

We now split the right-hand side into inliers and outliers. For the inliers  $a_i \in E^*$ , we have

$$\tilde{a}_i^\top \hat{S}^{-1} \tilde{a}_i = (a_i - c_\star)^\top S^{-1} (a_i - c_\star) + \frac{1}{d+1} \leq \frac{d}{d+1} + \frac{1}{d+1} = 1.$$

The outliers  $a_i \notin E^*$  are contained in a ball of radius  $2\beta$ , so  $\|a_i - c_\star\| \leq 4\beta$ . We also have that the width of  $E^*$  is at least 1 in every direction, so  $S^{-1} \preceq I$ , so

$$\begin{aligned} \tilde{a}_i^\top \hat{S}^{-1} \tilde{a}_i &= (a_i - c_\star)^\top S^{-1} (a_i - c_\star) \cdot \frac{d}{d+1} + \frac{1}{d+1} \\ &\leq 16\beta^2 \cdot \frac{d}{d+1} + \frac{1}{d+1} \leq 16\beta^2. \end{aligned}$$

Suppose the weight  $w$  puts  $\gamma(d+1)$  mass on outliers. Then, we have

$$\log \det(P) \leq \sum_i w_i \log \left( \tilde{a}_i^\top \hat{S}^{-1} \tilde{a}_i \right) \leq \gamma(d+1) \cdot \log(16\beta^2).$$

Therefore, we bound the optimal dual solution value by

$$\log \det(M) = \log \det(S) + \log \det(P) \leq \log \det(S) + \gamma(d+1) \cdot \log(16\beta^2).$$

Finally recall that, for convenience, we worked with  $M^{-1}$  being a scaling of  $\overline{M}^{-1}$  such that

$$E = \left\{ x : (x - c)^\top M^{-1} (x - c) \leq \frac{d}{d+1} \right\} = \left\{ x : (x - c)^\top \overline{M}^{-1} (x - c) \leq 1 \right\},$$

and similarly  $S^{-1}$  is a scaling of  $(S^*)^{-1}$  such that

$$E^* = \left\{ x : (x - c_\star)^\top S^{-1} (x - c_\star) \leq \frac{d}{d+1} \right\} = \left\{ x : (x - c_\star)^\top (S^*)^{-1} (x - c_\star) \leq 1 \right\}.$$

---

**Algorithm 5.1** Approximation algorithm for the minimum volume ellipsoid covering at least  $1 - \alpha$  fraction of points.

---

**Input** sample points  $A = \{a_1, \dots, a_n\} \subseteq \mathbb{R}^d$ , target miscoverage  $0 < \alpha < 1$ , parameter  $0 < \gamma < 1$

**Output** an ellipsoid  $\hat{E} \subseteq \mathbb{R}^d$

1. Initialize the candidate set  $\mathfrak{E} \leftarrow \emptyset$ .
  2. For each ordered pair  $(k_1, k_2) \in [n] \times [n]$ :
    - (a) Set a coarse bounding ball  $\hat{B} = B(\hat{c}, \hat{r})$  with  $\hat{c} = a_{k_1}$  and  $\hat{r} = \|a_{k_1} - a_{k_2}\|$ .
    - (b) Form the lifted set
 
$$\tilde{A} \leftarrow \left\{ \tilde{a}_i \in \mathbb{R}^{d+1} : a_i \in A \cap \hat{B} \right\}, \quad \tilde{a}_i := \begin{bmatrix} a_i, 1 \end{bmatrix}.$$
    - (c) Let  $J = \lceil c \cdot (\alpha n / \gamma d) \rceil$  for a constant  $c$ . Repeat for  $j = 1, \dots, J$ :
      - i. Solve the **dual** SDP on  $\tilde{A}$  (Proposition 5.2.4) to obtain weights  $\{w_i \in [0, 1]\}$  for  $\tilde{a}_i \in \tilde{A}$ .
      - ii. Independently remove each  $\tilde{a}_i \in \tilde{A}$  with probability  $w_i$ .
    - (d) Let  $\hat{E}$  be the minimum volume enclosing ellipsoid of  $\hat{A} \leftarrow \{a_i \in \mathbb{R}^d : \tilde{a}_i \in \tilde{A}\}$ .
    - (e) If  $|\hat{E} \cap A| \geq (1 - O(\alpha/\gamma))n$ , add  $\hat{E}$  to  $\mathfrak{E}$ .
  3. Repeat step 2  $O(\log n)$  times.
  4. Return the smallest-volume ellipsoid in  $\mathfrak{E}$ .
- 

Thus we have

$$\log \det(\overline{M}) \leq \log \det(S^*) + \gamma(d+1) \cdot \log(16\beta^2).$$

□

### 5.2.3 Algorithmic guarantee

**Theorem 5.2.7** (Approximately Optimal Confidence Ellipsoid). *Fix a set of points  $A \subseteq \mathbb{R}^d$ , a target miscoverage rate  $0 < \alpha < 1$ , a condition number  $\beta \geq 1$ , a parameter  $0 < \gamma < 1$ , and a success rate  $0 \leq c_1 < 1$ . Then, in  $\text{poly}(n, d) \cdot T_{\text{SDP}}$  time, with probability  $1 - o(1)$ ,*

*Algorithm 5.1* outputs an ellipsoid  $\hat{E}$ , such that  $\hat{E}$  approximately satisfies coverage,

$$|\hat{E} \cap A| \geq \left(1 - O\left(\frac{\alpha}{\gamma}\right)\right)n,$$

and is approximately volume optimal

$$\text{vol}(\hat{E}) \leq \text{vol}(E^*) \cdot (4\beta)^{2\gamma(d+1)},$$

where  $E^*$  is the minimum volume ellipsoid of condition number at most  $\beta$  that encloses at least  $(1 - \alpha)|A|$  points of  $A$ . Here  $T_{\text{SDP}}$  is the running time for solving the  $\text{poly}(n, d)$  sized semidefinite programs in [Proposition 5.2.3](#) and [Proposition 5.2.4](#).

The running time  $T_{\text{SDP}}$  of standard semidefinite programming solvers like the ellipsoid method for  $\text{poly}(n, d)$ -sized SDPs including [Proposition 5.2.3](#) and [Proposition 5.2.4](#) is polynomial in  $n, d$  and  $\log(1/\eta)$  where  $\eta > 0$  is the desired precision for solving the SDP [[Todd, 2016](#)]. Algorithms for solving the specific semidefinite program [Proposition 5.2.3](#) for the minimum volume enclosing ellipsoid has received particular attention, with faster algorithms based on interior-point methods and first-order methods; see [[Todd, 2016](#), [Cohen, Cousins, Lee, and Yang, 2019](#)] for details and more references. We will ignore numerical precision issues in the analysis, and assume that the SDP can be solved to optimality.

*Proof.* We can begin by assuming the following parameter settings. If  $\alpha < \frac{1}{n}$ , we can simply output the minimum volume enclosing ellipsoid containing all of  $A$ , by solving the relevant SDP ([Proposition 5.2.3](#), with lifted points). If  $\gamma \leq \alpha$ , then  $\alpha/\gamma \geq 1$ , so a trivial ellipsoid with 0 volume and 0 coverage satisfies the guarantee. Thus, we can assume that  $\gamma > \alpha \geq \frac{1}{n}$ . This allows us to assume that

$$\alpha n / \gamma d \geq 1. \tag{5.6}$$

If this is not the case, we can simply include  $\ell = \lceil \frac{\gamma d}{\alpha n} \rceil$  copies of each point in  $A$ . Since the ellipsoid either covers all copies of a point or none of them, the coverage and volume guarantees will carry over. The new number of points

$$n' = \ell \cdot n \leq \left( \frac{\gamma d}{\alpha n} + 1 \right) n \leq \gamma d n + n = \text{poly}(n),$$

is still polynomial in the original  $n$ , and thus the runtime guarantees will still hold.

We begin by using the procedure in [Lemma C.3.2](#) to construct a set  $\mathfrak{B}$  of at most  $n^2$  potential bounding balls for the points in  $A$ . Let  $E^\star = \{x \in \mathbb{R}^d : (x - c^\star)^\top S^{-1}(x - c^\star) \leq 1\}$  be the minimum volume ellipsoid (of arbitrary center), with condition number at most  $\beta$ , that encloses the points  $A$ . We are guaranteed that there exists a ball  $\widehat{B} \in \mathfrak{B}$ , such that  $\widehat{B}$  contains all points in  $E^\star$ ,

$$(E^\star \cap A) \subseteq \widehat{B},$$

and  $\widehat{B}$  has radius at most twice the largest axis length of  $E^\star$ .

For the sake of analysis, for a candidate ball  $B \in \mathfrak{B}$ , rescale the ball and the points in the ball so that  $\text{radius}(B) = 2\beta$ . For  $\widehat{B}$  this ensures that, in the rescaled space, the maximum axis length of  $E^\star$  is at least  $\beta$ , and therefore the minimum axis length of  $E^\star$  is at least 1. That is,  $S^{-1} \preceq I$ .

For all points  $a_i \in A \cap B$ , identify  $a_i \in \mathbb{R}^d$  with the lifted point  $\tilde{a}_i \in \mathbb{R}^{d+1}$ ,

$$\tilde{a}_i = (a_i ; 1).$$

Starting with  $\tilde{A} = \{\tilde{a}_i : a_i \in A \cap B\}$  as the included points, the algorithm will do

$$J = \lceil c \cdot (\alpha n / \gamma d) \rceil \leq 2c \cdot (\alpha n / \gamma d)$$

iterations of the following, where  $c$  is the constant from [Lemma C.3.4](#), and the inequality uses [\(5.6\)](#).

- (a) Solve the dual SDP on the included  $\tilde{a}_i$  to get weights  $0 \leq w_i \leq 1$ ,  $\sum_i w_i = d + 1$  ([Proposition 5.2.4](#)).
- (b) For each included  $i$  independently, discard  $\tilde{a}_i$  from the included points with probability  $w_i$ .

By [Lemma C.3.4](#) we have that with probability  $\geq 0.99$ , there exists some iteration  $j^*$  of this procedure on which less than  $\gamma(d + 1)$  mass was placed on outliers in the dual solution. On such an iteration  $j^*$ , let  $E_{(j^*)} = \{x \in \mathbb{R}^d : (x - c)^\top M_{(j^*)}^{-1}(x - c) \leq 1\}$  be the minimum volume ellipsoid that encloses all of the  $a_i$  that are included on iteration  $j^*$ . By [Lemma 5.2.6](#), we have that

$$\log \det(M_{(j^*)}) \leq \log \det(S) + \gamma(d + 1) \log(16\beta^2).$$

On iterations subsequent to  $j^*$ , we only remove points  $\tilde{a}_i$  from being included. Thus, after the last iteration the dual SDP value is only lower. For an ellipsoid  $E = \{x \in \mathbb{R}^d : (x - c)^\top M^{-1}(x - c) \leq 1\}$ , the volume of  $E$  is  $\text{vol}(E) = \kappa_d \det(M)^{1/2}$ , where  $\kappa_d$  is the volume of a unit ball in  $d$  dimensions. Thus

$$\log \text{vol}(\hat{E}) \leq \log \text{vol}(E^*) + \gamma(d + 1) \log(16\beta^2).$$

By [Lemma C.3.5](#), we have that with probability  $\geq 0.99$ , we remove at most  $c_2 \cdot \frac{\alpha n}{\gamma}$  points  $\tilde{a}_i$  over all iterations, for a constant  $c_2$ .  $\hat{B}$  started with at least  $(1 - \alpha)n$  points  $a_i$ . Thus, with

probability  $\geq 0.98$ ,  $\hat{E}$  simultaneously achieves approximate coverage

$$|\hat{E} \cap A| \geq \left(1 - O\left(\frac{\alpha}{\gamma}\right)\right)n,$$

and  $\hat{E}$  approximates  $E^\star$  in volume:

$$\text{vol}(\hat{E}) \leq \text{vol}(E^\star) \cdot (4\beta)^{2\gamma(d+1)}.$$

Finally, when we aggregate the candidate ellipsoids from each of the candidate  $B \in \mathfrak{B}$ , conditioned on  $\hat{E}$  achieving the desired coverage, we will have that the output ellipsoid will achieve coverage and have volume at most that of  $\hat{E}$ . Thus the output of the algorithm will achieve coverage and volume approximation if this procedure succeeded on at least one of the  $O(\log n)$  trials, which occurs with probability at least

$$1 - (0.02)^{O(\log n)} = 1 - o(1).$$

We analyze the runtime of the algorithm. There are at most  $n^2$  bounding balls in  $\mathfrak{B}$ . For each ball, checking which points lie within it, and doing the embedding procedure can be done in  $\text{poly}(n, d)$  time. Then, we solve the dual SDP  $J = O(\frac{\alpha n}{\gamma d})$  times. Since  $\gamma \geq 1/n$  (see opening paragraph of proof), the number of iterations is  $\text{poly}(n)$ . We solve the relevant SDPs in  $T_{\text{SDP}}$  time. The outlier removal procedure takes  $O(n)$  time per iteration. Finally, we solve one last SDP per iteration, and compare  $O(n^2)$  candidate solutions. Thus, in total, the runtime of the algorithm is  $\text{poly}(n, d) \cdot T_{\text{SDP}}$ .  $\square$

### 5.3 Algorithm for Subspace Recovery

In this section, we will design an algorithm for the *robust subspace recovery* problem using our algorithm for finding the minimum volume ellipsoid with outliers.

**Robust subspace recovery.** Formally, we are given  $n$  points  $a_1, \dots, a_n \in \mathbb{R}^d$ ; without loss of generality we assume they have unit length i.e.,  $\|a_i\|_2 = 1$  for all  $i \in [n]$ . We are also given as input a closeness parameter  $\varepsilon \in (0, 1)$ , and the outlier fraction  $\alpha \in (0, 1)$ . The goal is to find a subspace  $S \subset \mathbb{R}^d$  (passing through the origin) of smallest possible dimension that is  $\varepsilon$ -close in Euclidean distance to at least  $1 - \alpha$  fraction of the given points.<sup>3</sup>

Unlike previous works, we address the worst-case version of the problem. In particular, we make no genericity, smoothed analysis, or structural assumptions on either the outliers or the inliers, as is done in [Hardt and Moitra \[2013\]](#), [Bhaskara, Chen, Perreault, and Vijayaraghavan \[2019\]](#). More recent works based on sum-of-squares (SoS) relaxations assume distributional assumptions (e.g., Gaussianity) on the points in the subspace [\[Bakshi and Kothari, 2021\]](#). As mentioned earlier, the worst-case problem is computationally intractable [\[Hardt and Moitra, 2013\]](#); see also [Section 5.4](#). Hence we will aim for approximation guarantees i.e., polynomial time algorithms that is (approximately) competitive in terms of all the parameters: the dimension, fraction of outliers  $\alpha$ , and the closeness parameter  $\varepsilon$ . Note that in the absence of any structural assumptions on the outliers or inliers, our goal is to find a subspace of small dimension that is close to many points; it may not be possible to recover any ground truth subspace.

The above algorithm runs our robust minimum volume enclosing ellipsoid algorithm on a small random perturbation applied to our given points. The random perturbation is

---

<sup>3</sup>Given a subspace  $S \subset \mathbb{R}^d$  of dimension  $s$  given by an (orthogonal) projection matrix  $\Pi_S$  of rank  $s$ , we will say that a point  $a \in \mathbb{R}^d$  is  $\varepsilon$ -close to subspace  $S$  iff  $\|\Pi_S^\perp a\|_2 = \|(I - \Pi_S)a\|_2 \leq \varepsilon\|a\|_2$ .



---

**Algorithm 5.2** Algorithm for finding a subspace of dimension at most  $(1-\gamma)d$  that contains at least  $1 - \alpha'$  fraction of points

---

**Input:** a set of points (unit vectors)  $a_1, \dots, a_n \in \mathbb{R}^d$ , parameter  $\gamma > 0$ , closeness parameter  $\varepsilon \in (0, 1)$

**Output:** a subspace  $S \subset \mathbb{R}^d$  passing through the origin

1. Set  $\varepsilon_\star := \varepsilon^{4/\gamma}$ . We randomly perturb the points to get  $a'_1, \dots, a'_n$  with  $\tilde{a}_i = a_i + \zeta_i$  where  $\zeta \sim N(0, \frac{\varepsilon_\star^2}{d} I_d)$ . Normalize the points to get points  $a'_i = \tilde{a}_i / \|\tilde{a}_i\|_2$  for each  $i \in [n]$ .
  2. Run [Algorithm 5.1](#) (from [Theorem 5.1.3](#)) on the points  $a'_1, \dots, a'_n$  to find an ellipsoid  $\hat{E} = \{x : x^\top (\hat{M})^{-1} x \leq 1\}$  passing through the origin satisfying the guarantees of [Theorem 5.1.3](#). (Note that to find an origin-centered ellipsoid, we can apply [Algorithm 5.1](#) to the  $2n$  points  $\pm a'_1, \pm a'_2, \dots, \pm a'_n$ .)
  3. Let  $\hat{S}$  be the top singular space of  $\hat{M}$  spanned by all the singular vectors with corresponding singular value  $\geq \varepsilon^2$ . Return  $\hat{S}$ .
- 

performed by our algorithm, and it is applied to all the points. We remark that this is different from the smoothed analysis setting where the algorithm is given randomly perturbed points as input, where the points in the subspace are assumed to be randomly perturbed in a way that keeps them inside the (unknown) ground-truth subspace. In contrast, our algorithm applies to an arbitrary set of points given as input.

**Theorem 5.3.1.** *Let  $\alpha \in (0, 1)$ , and  $\gamma \in (0, 1)$ ,  $\varepsilon \in (0, 1)$  be input parameters. There exists universal constants  $c_1, c_2 > 0, C > 1$  such that the following holds for  $\varepsilon_\star \leq \varepsilon^{4/\gamma}$ . Suppose we are given as input  $n \geq Cd \log(d/\varepsilon_\star) + \gamma d/\alpha$  unit vectors  $A = \{a_1, \dots, a_n\} \in \mathbb{R}^d$ , with a subspace  $S^\star$  of dimension  $(1 - 8\gamma)d$  that is  $\varepsilon_\star$  close to at least  $(1 - \alpha)n$  of the given points. Then [Algorithm 5.2](#) runs in polynomial time in  $n, d$  and finds w.h.p. a subspace  $S \subset \mathbb{R}^d$  of dimension at most  $(1 - \gamma)d$  such that at least  $(1 - \alpha')n$  points are  $\varepsilon$  close to  $S$ , for  $\alpha' = c_2\alpha/\gamma$ .*

We remark that the lower bound condition on the number of samples  $n$  is essentially without loss of generality since we can just resample (or take multiple copies of the dataset) and run the same algorithm to get a similar guarantee; this dependence will feature in the running time. When we know that there are many points that lie exactly on the subspace  $S$ , we have

$\varepsilon_\star = 0$ . So we can pick  $\varepsilon$  to be whatever desired accuracy (that can be arbitrarily close to 0) and since  $\varepsilon_\star = 2^{-\log(1/\varepsilon)/\gamma}$  (note that the number of bits of precision is only  $\log(1/\varepsilon)/\gamma$ ), and apply the above algorithm to find a subspace that is  $\varepsilon$ -close to the desired number of points.

Before we proceed to the proof, we first need the following claim that uses the random perturbations to the points to prove that any ellipsoid that encloses a significant fraction of the points needs to be sufficiently fat.

**Lemma 5.3.2.** *In the notation of [Algorithm 5.2](#), there exists universal constants  $C, c, c' > 0$  such that when  $n \geq Cd \log(d/\varepsilon_\star)$  we have with probability at least  $1 - \exp(-c'n) - n \exp(-2d)$  (over the random perturbation of the points) that any origin-centered ellipsoid  $E = \{x : x^\top M^{-1}x \leq 1\}$  that encloses at least  $4n/5$  of the points  $a'_1, \dots, a'_n$  has  $\lambda_{\min}(M) \geq c(\varepsilon_\star)^2/d$ .*

This lemma follows directly from the following anti-concentration claim that shows that in every direction most points are far from the origin.

**Claim 5.3.1.** *In the notation of [Lemma 5.3.2](#) for some universal constant  $c, c' > 0$ , we have with probability  $1 - \exp(-c'n) - n \exp(-2d)$ ,*

$$\forall u \in \mathbb{R}^d \text{ s.t. } \|u\| = 1, \left| \left\{ i \in [n] : \langle a'_i, u \rangle^2 \leq \frac{\varepsilon_\star^2}{256d} \right\} \right| \leq 0.2n. \quad (5.7)$$

We defer the proof of [Claim 5.3.1](#) to later. The choice of the constant 0.2 is arbitrary, and can be chosen as desired.

*Proof of [Lemma 5.3.2](#).* We condition on the event (5.7) in [Claim 5.3.1](#). Consider any ellipsoid  $E = \{x : x^\top M^{-1}x \leq 1\}$  containing  $T \subseteq \{a'_1, \dots, a'_n\}$  with  $|T| \geq 4n/5$ . Let  $\lambda$  be the smallest eigenvalue of  $M$ , and let  $u$  be a corresponding eigenvector of unit length. Then

$$\forall i \in T, 1 \geq (a'_i)^\top M^{-1}a'_i \geq \frac{\langle a'_i, u \rangle^2}{\lambda} \implies \forall i \in T, \langle a'_i, u \rangle^2 \leq \lambda.$$

From [Claim 5.3.1](#), we conclude that  $\lambda \geq \frac{\varepsilon_\star^2}{256d}$ .  $\square$

We now proceed to the proof of the main theorem of the section.

*Proof of [Theorem 5.3.1](#).* Set  $\gamma_\star = 8\gamma$ ,  $\varepsilon_\star = \varepsilon^{4/\gamma}$ . Suppose there exists a subspace  $S^\star$  (passing through the origin) of dimension  $(1 - \gamma_\star)d$  that is  $\varepsilon_\star$ -close to  $(1 - \alpha)n$  points in  $A$ . Let  $\Pi \in \mathbb{R}^{d \times d}$  be the projection matrix onto the subspace  $S^\star$  and  $\Pi^\perp = I - \Pi$  be the projection onto the subspace orthogonal to  $S^\star$ .

We first show that there is a well-conditioned ellipsoid  $E^\star$  of small volume containing the points that are  $\varepsilon_\star$ -close to  $S^\star$ . Consider the ellipsoid  $E^\star$  given by

$$E^\star = \left\{ x \in \mathbb{R}^d : \frac{x^\top \Pi x}{2} + \frac{x^\top \Pi^\perp x}{2\varepsilon_\star^2} \leq 1 \right\}. \quad (5.8)$$

We have  $\|a_i\|_2 = 1$  for all points  $i \in [n]$ . If  $a_i$  is  $\varepsilon_\star$ -close to  $S^\star$ , then

$$\begin{aligned} \|\Pi^\perp a_i\|_2 \leq \varepsilon_\star &\implies \frac{a_i^\top \Pi a_i}{2} + \frac{a_i^\top \Pi^\perp a_i}{2\varepsilon_\star^2} \leq 1. \\ &\implies a_i \in E^\star. \end{aligned}$$

Moreover, the ellipsoid  $E^\star$  has condition number  $\beta = 1/\varepsilon_\star$ , and its volume is

$$\text{vol}(E^\star) = \kappa_d \cdot 2^{(1-\gamma_\star)d/2} (2\varepsilon_\star^2)^{\gamma_\star d/2} = \kappa_d \cdot 2^{d/2} \varepsilon_\star^{\gamma_\star d},$$

where  $\kappa_d$  is the volume of a unit radius ball in  $d$  dimensions.

By running the algorithm from [Theorem 5.1.3](#) on these points, we get an origin-centered

ellipsoid  $\widehat{E} = \{x : x^\top \widehat{M}^{-1}x \leq 1\}$ . Let  $T = \{i \in [n] : a'_i \in \widehat{E}\}$ ; then  $|T| \geq (1 - c'\alpha/\gamma)n$ , and

$$\text{vol}(\widehat{E}) \leq \beta^{\gamma d} \text{vol}(E^*) \leq \kappa_d \cdot 2^{d/2} \varepsilon_\star^{(\gamma_\star - \gamma)d} \leq \kappa_d \cdot 2^{d/2} \varepsilon_\star^{\frac{7}{8}\gamma_\star d},$$

since  $\gamma_\star = 8\gamma$ .

Let  $\widehat{S}$  be the subspace spanned by the eigenvalues of  $\widehat{M}$  larger than  $\varepsilon$ , and let  $\widehat{\Pi}$  be the projection matrix onto the subspace  $\widehat{S}$ , and  $\widehat{\Pi}^\perp$  be the projection onto the subspace orthogonal to  $\widehat{S}$ . Then

$$\forall i \in T, \quad 1 \geq \|\widehat{M}^{-1/2} a'_i\|_2^2 \geq \frac{\|\widehat{\Pi}^\perp a'_i\|_2^2}{\varepsilon^2} \implies \|\widehat{\Pi}^\perp a'_i\|_2^2 \leq \varepsilon^2.$$

Hence every point  $a'_i$  for each  $i \in T$  is  $\varepsilon$ -close to  $\widehat{S}$ , as required. By triangle inequality, along with standard length concentration of Gaussians, we have that  $a_i$  is  $\varepsilon + O(\varepsilon_\star) \leq 2\varepsilon$ -close to  $\widehat{S}$ .

It remains to lower bound the co-dimension of  $\widehat{S}$ . Let the codimension  $\hat{k} := d - \dim(\widehat{S})$ . Let  $\widehat{M}$  have eigenvalues  $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \dots \geq \hat{\lambda}_d \geq 0$ ; then  $\hat{\lambda}_{d-\hat{k}} \geq \varepsilon^2$ . From [Lemma 5.3.2](#), we have  $\hat{\lambda}_d \geq c_2 \varepsilon_\star^2 / d$  w.h.p.. Conditioning on this, we have

$$\begin{aligned} \kappa_d \cdot 2^{d/2} \varepsilon_\star^{\frac{7}{8}\gamma_\star d} &\geq \text{vol}(\widehat{E}) = \kappa_d \prod_{i=1}^d \hat{\lambda}_i \\ 2^{d/2} \varepsilon_\star^{\frac{7}{8}\gamma_\star d} &\geq \prod_{i \leq d-\hat{k}} \hat{\lambda}_i \cdot \prod_{i: i \geq d-\hat{k}+1} \hat{\lambda}_i \geq (\varepsilon^2)^{d-\hat{k}} (\hat{\lambda}_d)^{\hat{k}} \\ &\geq \varepsilon^{2d} \cdot \left(\frac{c_2 \varepsilon_\star^2}{d}\right)^{\hat{k}} \geq \varepsilon_\star^{\gamma d} \cdot \left(\frac{c_2 \varepsilon_\star^2}{d}\right)^{\hat{k}}, \end{aligned} \tag{5.9}$$

where the last inequality is from  $\varepsilon_\star \leq \varepsilon^{4/\gamma}$ .

Since  $\varepsilon_\star \leq \varepsilon^{4/\gamma}$  and  $\gamma_\star = 8\gamma$ , we have  $\varepsilon_\star^{\gamma_\star/4} = \varepsilon_\star^{2\gamma} \leq \varepsilon^8$ . Thus, by picking closeness parameter

$\varepsilon \leq 2^{-1/8}$ , we have

$$2^{d/2} \varepsilon_\star^{\frac{7}{8}\gamma_\star d} \leq (2\varepsilon_\star^{\gamma_\star/4})^{d/2} \varepsilon_\star^{\frac{3}{4}\gamma_\star d} \leq (2\varepsilon^8)^{d/2} \varepsilon_\star^{\frac{3}{4}\gamma_\star d} \leq \varepsilon_\star^{\frac{3}{4}\gamma_\star d}.$$

Putting this together with (5.9), we get

$$\begin{aligned} \varepsilon_\star^{\frac{3}{4}\gamma_\star d} &\geq \varepsilon_\star^{\gamma d} \cdot \left(\frac{c_2 \varepsilon_\star^2}{d}\right)^{\hat{k}} \geq \varepsilon_\star^{\frac{1}{8}\gamma_\star d} \cdot \left(\frac{c_2 \varepsilon_\star^2}{d}\right)^{\hat{k}} \\ \left(\frac{c_2 \varepsilon_\star^2}{d}\right)^{\hat{k}} &\leq \varepsilon_\star^{\frac{5}{8}\gamma_\star d} \implies \hat{k} \geq \frac{5}{8}\gamma_\star d \cdot \left(\frac{\log(1/\varepsilon_\star)}{2\log(1/\varepsilon_\star) + \log(d/c_2)}\right) \geq \gamma d, \end{aligned}$$

where the last inequality is due to  $\log(1/\varepsilon_\star) \geq \log(1/\varepsilon_\star^{4/\gamma}) \geq \log(d/c_2)$  by picking  $\varepsilon$  sufficiently small. This lower bounds the codimension of  $\widehat{S}$ . This completes the proof.  $\square$

We now complete the proof of the helper claim.

*Proof of Claim 5.3.1.* This follows from a relatively standard net-based argument. Set  $\eta := \frac{\varepsilon_\star}{32\sqrt{d}}$ , and consider an  $\eta$ -net  $\mathcal{N}_\eta$  of unit vectors in  $\mathbb{R}^d$ ;  $|\mathcal{N}_\eta| \leq \left(\frac{2+\eta}{\eta}\right)^d$ . For a fixed unit vector  $u \in \mathcal{N}_\eta$  and a fixed  $i \in [n]$ , the small ball probability (or anticoncentration) of the Gaussian  $\zeta \sim N(0, \frac{\varepsilon_1^2}{d}I)$ ,

$$\begin{aligned} \mathbb{P}\left[|\langle a'_i, u \rangle| \leq \frac{\varepsilon_1}{8\sqrt{d}}\right] &= \mathbb{P}_{g \sim N(0,1)}\left[\left|\langle a_i, u \rangle + \frac{\varepsilon_1}{\sqrt{d}}g\right| \leq \frac{\varepsilon_1}{8\sqrt{d}}\right] \\ &\leq \sup_{t \in \mathbb{R}} \mathbb{P}_{g \sim N(0,1)}\left[|g + t| \leq \frac{1}{8}\right] \leq 0.1. \end{aligned}$$

For a fixed  $u \in \mathcal{N}_\eta$ ,  $\mathbb{P}\left[\left|\left\{i \in [n] : |\langle a'_i, u \rangle| \leq \frac{\varepsilon_1}{8\sqrt{d}}\right\}\right| \geq 0.2n\right] \leq \exp(-c_1 n)$ .

By doing a union bound over all  $u \in \mathcal{N}_\eta$ , we have that since  $n \geq \frac{2}{c_1} \log(\mathcal{N}_\eta)$ , we have

$$\mathbb{P} \left[ \sup_{u \in \mathcal{N}_\eta} \left| \left\{ i \in [n] : |\langle a'_i, u \rangle| \leq \frac{\varepsilon_1}{8\sqrt{d}} \right\} \right| \geq 0.2n \right] \leq |\mathcal{N}_\eta| \cdot \exp(-c_1 n) \leq \exp(-\tfrac{1}{2}c_1 n). \quad (5.10)$$

Let us condition on the above event in (5.10). For any unit vector  $v \in \mathbb{R}^d$ , there exists  $u^* \in \mathcal{N}_\eta$  with  $\|v - u^*\|_2 \leq \eta$ . Moreover  $\|a'_i\| \leq \|a_i\| + \|\zeta_i\|$  where  $\zeta_i \sim N(0, \frac{\varepsilon_*^2}{d}I)$ ; so with probability at least  $1 - n \exp(-2d)$ , we have  $\forall i \in [n]$ ,  $\|a'_i\| \leq 1 + 2\varepsilon_1 \leq 2$ . Hence  $\forall i \in [n]$ ,  $|\langle a'_i, v \rangle - \langle a'_i, u^* \rangle| \leq 2\eta \leq \frac{\varepsilon_1}{16\sqrt{d}}$ . Thus we conclude that with high probability

$$\mathbb{P} \left[ \sup_{v \in \mathbb{R}^d: \|v\|=1} \left| \left\{ i \in [n] : |\langle a'_i, v \rangle| \leq \frac{\varepsilon_1}{16\sqrt{d}} \right\} \right| \geq 0.2n \right] \leq \exp(-\tfrac{1}{2}c_1 n).$$

□

## 5.4 Computational Intractability

In this section, we provide evidence of the computational intractability of the minimum volume confidence ellipsoid problem. Specifically, under the Small Set Expansion (SSE) hypothesis of [Raghavendra and Steurer \[2010\]](#), we show that this problem is hard to approximate. The SSE hypothesis is closely related to the Unique Games Conjecture [[Khot, 2002](#)].

**Conjecture 5.4.1** (SSE hypothesis of [Raghavendra and Steurer \[2010\]](#)). *For any  $\varepsilon > 0$ , there is a constant  $\delta \in (0, 1)$  such that there is no polynomial time algorithm to distinguish between the following two cases given a graph  $G = (V, E)$  on  $n$  vertices with degree  $\Delta$ :*

- *YES: Some subset  $S \subseteq V$  with  $|S| = \delta n$  satisfies that the induced subgraph on  $S$  is dense i.e., the number of edges going out of  $S$  is  $|E(S, V \setminus S)| \leq \varepsilon \Delta |S|$  edges.*

- *NO: Any set  $S \subseteq V$  with  $\frac{1}{2}\varepsilon\delta n \leq |S| \leq 2\delta n$  has most of the edges incident on it going outside i.e.,  $|E(S, V \setminus S)| \geq (1 - \varepsilon)\Delta|S|$ .*

Define the edge expansion of a  $\Delta$ -regular graph  $G$  at scale  $\delta$  by

$$\phi_G(\delta) := \min_{S: |S| \leq \delta n} \frac{|E(S, V \setminus S)|}{\Delta|S|}.$$

Then, the SSE hypothesis asserts that it is computationally hard to distinguish between: (1) YES case, where  $\phi_G(\delta) \leq \varepsilon$ , and (2) NO case, where  $\phi_G(\delta') \geq 1 - \varepsilon$  for all  $\delta' \in [\delta\varepsilon/2, 2\delta]$ .

We now show that the minimum volume confidence ellipsoid problem is SSE-hard to approximate. In particular, for ellipsoids covering at least a  $(1 - \alpha)$  fraction of the points, it is SSE-hard to obtain a volume approximation factor better than  $\beta^{\gamma d/2}$  for any fixed  $\gamma < \delta$ . This hardness remains true even if one allows a small slack in the coverage requirement, replacing  $1 - \alpha$  by  $1 - (1 + \delta/2)\alpha$ .

**Theorem 5.4.2** (SSE-hardness). *Fix any  $\varepsilon, \delta \in (0, 0.1)$  and  $\gamma \in (0, \delta)$  and  $\alpha \in (0, 1 - (1 - \varepsilon)\delta)$ . For any  $C \geq 5/\delta$ , given a graph  $G$ , we can construct in polynomial time a set  $A = \{a_1, \dots, a_n\} \subseteq \mathbb{R}^d$  with the property:*

- *YES: If  $\phi_G(\delta) \leq \varepsilon$ , then there exists an  $\beta$ -conditioned ellipsoid covering  $(1 - \alpha)n$  points in  $A$  with volume at most  $K$ ; and*
- *NO: If  $\phi_G(\delta') \geq 1 - \varepsilon$  for all  $\delta' \in [\delta\varepsilon/2, 2\delta]$ , then every ellipsoid covering  $(1 - (1 + \delta/2)\alpha)n$  points has volume at least  $\beta^{\gamma d/2} K$ ,*

where  $\beta = 1/\tilde{\varepsilon}$ ,  $K = \kappa_d 2^{d/2} \tilde{\varepsilon}^{(1-\delta)d}$ ,  $\tilde{\varepsilon} = d^{-C}$ , and  $\kappa_d$  is the volume of the Euclidean unit ball in  $\mathbb{R}^d$ .

Consequently, unless the Small-Set Expansion conjecture fails, no polynomial-time algorithm

can approximate the volume of the optimal  $\beta$ -conditioned ellipsoid with coverage  $1 - \alpha$  within a factor of  $\beta^{\gamma d/2}$  for any  $\gamma \in (0, \delta)$ .

To prove the above SSE-hardness result, we first use the reduction from the Small Set Expansion to the robust subspace recovery in Theorem 14 by [Hardt and Moitra \[2013\]](#). Their construction is randomized and yields hardness in the regime where one seeks a subspace containing only a  $(1 - \varepsilon)\delta$  fraction of points. We make two minor modifications. First, we make the construction deterministic. Second, by padding the instance with additional points at the origin, we extend the reduction to the higher-coverage regime  $1 - \alpha$  for any  $\alpha \in (0, 1 - (1 - \varepsilon)\delta)$ . The resulting statement is as follows.

**Theorem 5.4.3.** *Let  $\varepsilon, \delta \in (0, 0.1)$  and  $\alpha \in (0, 1 - (1 - \varepsilon)\delta)$ . There is a polynomial-time reduction mapping a  $\Delta$ -regular graph  $G$  to an instance  $U = \{u_1, \dots, u_n\} \subset \mathbb{R}^d$  of unit vectors such that:*

- *YES: If  $\phi(\delta) \leq \varepsilon$ , there exists a subspace  $S^*$  of dimension  $\delta d$  containing at least  $1 - \alpha$  fraction of the points.*
- *NO: If  $\phi(\delta') \geq 1 - \varepsilon$  for every  $\delta' \in [\delta\varepsilon/2, 2\delta]$ , then for every  $\delta' \in [\delta\varepsilon/2, 2\delta]$ , every subspace of dimension  $\delta'd$  contains at most  $1 - (1 + (1 - 9\varepsilon)\delta)\alpha$  fraction of the points.*

*Proof of Theorem 5.4.3.* We construct the instance  $U$  similar to the reduction by [Hardt and Moitra \[2013\]](#) as follows. Let  $G(V, E)$  be a  $\Delta$ -regular graph on  $d$  vertices. For each edge  $e = (i, j) \in E$ , let  $u'_e = (e_i + e_j)/\sqrt{2}$  be the unit length vector corresponding to  $e$ . Let  $U' = \{u'_1, \dots, u'_m\} \subset \mathbb{R}^d$  be the instance constructed from  $G$  (with  $d = |V|$  and  $m = |E| = \Delta d/2$ ). Fix any parameter  $\eta \geq 0$  which is specified later and set  $T = \eta m/\delta$ . Let  $U$  be the set of points in  $U'$  and  $T$  points located at the origin. Then,  $U$  contains  $|U| = m + T$  points.



**Completeness.** Assume  $\phi_G(\delta) \leq \varepsilon$ . Then, by Theorem 14 in [Hardt and Moitra \[2013\]](#), there exists a  $\delta d$ -dimensional subspace  $S^*$  containing at least  $(1 - \varepsilon)\delta m$  points of  $U'$ . Moreover,  $S^*$  is a coordinate subspace spanned by a subset of the standard basis vectors. Thus, all  $T$  appended points at the origin lie in  $S^*$  as well. Therefore, we have for our new instance  $U$

$$\frac{|U \cap S^*|}{|U|} \geq \frac{T + (1 - \varepsilon)\delta m}{m + T} = 1 - \frac{m - (1 - \varepsilon)\delta m}{m + T} \geq 1 - \frac{\delta - (1 - \varepsilon)\delta^2}{\delta + \eta},$$

where the last inequality is from  $T = \eta m / \delta$ . For any  $\alpha \in (0, 1 - (1 - \varepsilon)\delta)$ , we choose the number of appended points  $T = \eta m / \delta$  such that  $\eta$  satisfies  $\alpha = \frac{\delta - (1 - \varepsilon)\delta^2}{\delta + \eta}$ . Then, this subspace  $S^*$  contains at least  $1 - \alpha$  fraction of the points in  $U$ .

**Soundness.** Assume  $\phi_G(\delta') \geq 1 - \varepsilon$  for all  $\delta' \in [\delta\varepsilon/2, 2\delta]$ . Fix any  $\delta' \in [\delta\varepsilon/2, 2\delta]$  and any subspace  $S$  of dimension  $r = \delta'd$ . We use the following claim to bound the number of points in  $U$  covered by this subspace  $S$ .

**Claim 5.4.1.** *For every integer  $r$  with  $\frac{\delta\varepsilon}{2}d \leq r \leq 2\delta d$ , every set  $P \subseteq U'$  that lies in some subspace of dimension at most  $r$  satisfies*

$$\frac{|P|}{m} \leq 4\varepsilon \frac{r}{d}.$$

*Equivalently, any  $r$ -dimensional subspace contains at most a  $4\varepsilon \cdot (r/d)$  fraction of the points in  $U'$ .*

By [Claim 5.4.1](#),

$$|U' \cap S| \leq 4\varepsilon \frac{r}{d} m = 4\varepsilon \delta' m \leq 8\varepsilon \delta m.$$

Hence

$$\frac{|U \cap S|}{|U|} \leq \frac{T + 8\varepsilon \delta m}{m + T} = 1 - \frac{m - 8\varepsilon \delta m}{m + T} \leq 1 - \frac{\delta - 8\varepsilon \delta^2}{\delta + \eta}.$$

Since  $\varepsilon \in (0, 0.1)$ , we have  $1 - \varepsilon \geq 8\varepsilon$ . Then, we have

$$\frac{|U \cap S|}{|U|} \leq 1 - \frac{\delta - 8\varepsilon\delta^2}{\delta + \eta} = 1 - \alpha - \frac{(1 - 9\varepsilon)\delta^2}{\delta + \eta} \leq 1 - \alpha - (1 - 9\varepsilon)\delta\alpha,$$

where the last inequality is from  $\alpha \leq \delta/(\delta + \eta)$ . Thus, the subspace  $S$  contains at most  $1 - (1 + (1 - 9\varepsilon)\delta)\alpha$  fraction of points in  $U$ .  $\square$

*Proof of Claim 5.4.1.* Let  $P \subseteq U$  be contained in a subspace of dimension at most  $r$ , let  $F \subseteq E$  be the corresponding edge set, and let  $S := N_B(F) \subseteq V$  be the neighborhood of  $F$  in the bipartite incidence graph  $B = (E, V)$ .

We require the following claim, corresponding to Claim 3.4 in [Hardt and Moitra \[2013\]](#). As their argument relies on a randomized construction, we include a proof using our deterministic construction in Appendix C.2.

**Claim 5.4.2.** *For every set of points  $P \subseteq U$  corresponding to a set of edges  $F \subseteq E$ , we have*

$$\frac{|N_B(F)|}{2} \leq \dim(\text{span } P) \leq |N_B(F)|.$$

By Claim 5.4.2, we have

$$\frac{|S|}{2} \leq \dim(\text{span } P) \leq |S|. \tag{5.11}$$

Then, we have  $r \leq |S| \leq 2r$ .

We consider two cases.

*Case 1:*  $|S| \leq 2\delta d$ . Since  $|S| \geq r \geq \delta\varepsilon d$ , let  $\delta' = |S|/d \in [\delta\varepsilon/2, 2\delta]$ . By the small-set expansion hypothesis, the number of edges leaving  $S$  is at least  $|E(S, V \setminus S)| \geq (1 - \varepsilon)\Delta|S|$ .

Since  $\Delta|S| = 2|E(S, S)| + |E(S, V \setminus S)|$ , we obtain

$$|E(S, S)| \leq \frac{\varepsilon\Delta}{2} |S|.$$

As  $F \subseteq E(S, S)$  and  $|S| \leq 2r$ , we get

$$|P| = |F| \leq \frac{\varepsilon\Delta}{2} |S| \leq \varepsilon\Delta r.$$

Dividing by  $m = \Delta d/2$  yields  $|P|/m \leq 2\varepsilon (r/d)$ .

*Case 2:*  $|S| > 2\delta d$ . Set  $s := 2\delta d$  and let  $S' \subseteq S$  be a size- $s$  subset maximizing  $|E(S', S')|$ .

By averaging (equivalently, taking expectation over a uniform size- $s$  subset of  $S$ ),

$$|E(S', S')| \geq \frac{\binom{s}{2}}{\binom{|S|}{2}} |E(S, S)| \geq \left(\frac{s}{|S|}\right)^2 |E(S, S)| \geq \left(\frac{s}{|S|}\right)^2 |F|. \quad (5.12)$$

Applying the expansion hypothesis to  $S'$  (note  $|S'| = 2\delta d$ ) gives

$$|E(S', S')| \leq \frac{\varepsilon\Delta}{2} |S'| = \varepsilon\Delta \delta d.$$

Combining with (5.12) and using  $|S| \leq 2r$  from (5.11),

$$|P| = |F| \leq \left(\frac{|S|}{s}\right)^2 |E(S', S')| \leq \left(\frac{2r}{2\delta n}\right)^2 \cdot \varepsilon\Delta \delta d = \varepsilon\Delta \frac{r^2}{\delta n}.$$

Hence

$$\frac{|P|}{m} \leq \frac{\varepsilon\Delta r^2/(\delta d)}{\Delta d/2} = 2\varepsilon \frac{r^2}{\delta d^2} \leq 4\varepsilon \frac{r}{d},$$

where the last step uses  $r/d \leq 2\delta$  since  $r \leq 2\delta d$ .

Combining both cases yields the claimed bound  $|P|/m \leq 4\varepsilon (r/d)$ . □

*Proof of Theorem 5.4.2.* We construct a set of points  $A$  as follows. Given a Small Set Expansion instance  $G(V, E)$ , let  $U = \{u_i\}_{i=1}^n$  be the set of points constructed by Theorem 5.4.3 with parameters  $(\varepsilon, \delta)$ . We then add random perturbations to these points and normalize them to unit vectors. Independently sample  $z_i \sim \mathcal{N}(0, \tilde{\varepsilon}^2 I_d/d)$  with  $\tilde{\varepsilon} = d^{-C}$  for a fixed constant  $C \geq 10$ , and set  $a_i = (u_i + z_i)/\|u_i + z_i\|$ . We then consider the set of points  $A = \{a_i\}$  and target coverage  $1 - \alpha$  for  $\alpha \in (0, 1 - (1 - \varepsilon)\delta)$ .

**Completeness.** If  $\phi_G(\delta) \leq \varepsilon$ , then by Theorem 5.4.3, there exists a subspace  $S^* \subset \mathbb{R}^d$  of dimension  $\delta d$  containing at least  $(1 - \alpha)n$  points of  $U$ .

Let  $\Pi$  be the projection to this subspace  $S^*$  of dimension  $\delta d$ . Then, for any  $\tilde{\varepsilon} \in (0, 1)$  define the ellipsoid

$$E^* := \left\{ x \in \mathbb{R}^d : \frac{x^\top \Pi x}{2} + \frac{x^\top \Pi^\perp x}{2(\tilde{\varepsilon})^2} \leq 1 \right\}. \quad (5.13)$$

If a unit vector  $a$  is  $\tilde{\varepsilon}$ -close to  $S$  then  $a \in E^*$ ; and

$$\text{vol}(E^*) = \kappa_d 2^{\delta d/2} (2\tilde{\varepsilon}^2)^{(1-\delta)d/2} = \kappa_d 2^{d/2} \tilde{\varepsilon}^{(1-\delta)d}, \quad (5.14)$$

where  $\kappa_d$  is the volume of the Euclidean unit ball in  $\mathbb{R}^d$ .

For any points  $u_i \in S$ , and by a Gaussian tail,  $\|z_i\|_2 \leq \tilde{\varepsilon}$  w.h.p., hence  $\text{dist}(a_i, S) \leq \tilde{\varepsilon}$ . Thus, we have all such  $a_i$  lie in the  $\beta$ -conditioned ellipsoid  $E^*$  with volume  $K$ .

**Soundness.** We now consider the NO case. Suppose, toward contradiction, that there exists an ellipsoid  $E = \{x : x^\top M^{-1}x \leq 1\}$  such that for  $\alpha' = (1 + \delta/2)\alpha \geq (1 + (1 - 9\varepsilon)\delta)\alpha - 40\delta d/n$

$$|E \cap A| \geq (1 - \alpha')n \quad \text{and} \quad \text{vol}(E) \leq \beta^{\gamma d/2} K. \quad (5.15)$$

Let  $E^*$  be a minimum-volume ellipsoid in  $\mathbb{R}^d$  covering at least  $(1 - \alpha')$  fraction of the points in  $A$ , and let  $M^*$  be its shape matrix. Then

$$\text{vol}(E^*) \leq \text{vol}(E) \implies \det(M^*) \leq \left( \frac{\text{vol}(E)}{\kappa_d} \right)^2.$$

Since  $\text{vol}(E) \leq \beta^{\gamma d/2} K$ , we have

$$\det(M^*) \leq \beta^{\gamma d} 2^d \tilde{\varepsilon}^{2(1-\delta)d}. \quad (5.16)$$

We now bound the small eigenvalue directions in  $M^*$ . Let the eigenvalues of  $M^*$  be  $\lambda_1 \geq \dots \geq \lambda_d > 0$ . Fix a constant threshold  $\tau := \varepsilon_0^2$  with  $\varepsilon_0 \in (0, 0.01)$ , and let  $\hat{T} \subset \mathbb{R}^d$  be the span of eigenvectors with eigenvalues  $\lambda_i \geq \tau$ . Let  $\hat{k}_d := d - \dim(\hat{T})$  be the number of small eigenvalues and  $\Pi_{\hat{T}}^\perp$  be the projection to orthogonal directions of  $\hat{T}$ . For any point  $a_i$  contained in  $E^*$ , we have  $1 \geq a_i^\top (M^*)^{-1} a_i \geq \|\Pi_{\hat{T}}^\perp a_i\|_2^2 / \tau$ . Thus, every covered point is  $\varepsilon_0$ -close to the subspace  $\hat{T}$ ; hence at least  $(1 - \alpha')n$  of the points in  $A$  satisfy  $\text{dist}(\tilde{a}_i, \hat{T}) \leq \varepsilon_0$ . By [Lemma 5.3.2](#), with high probability  $\lambda_{\min}(M^*) \geq c\tilde{\varepsilon}^2/d$  for some constant  $c$ , so

$$\tau^{d-\hat{k}_d} \left( \frac{c\tilde{\varepsilon}^2}{d} \right)^{\hat{k}_d} \leq \det(M^*) \leq \beta^{\gamma d} 2^d \tilde{\varepsilon}^{2(1-\delta)d}. \quad (5.17)$$

Let  $Y := \log d - (\log(c\tilde{\varepsilon}^2) - \log \tau) > 0$ . Taking logs in the first inequality of Equation (5.17) and rearranging gives

$$\hat{k}_d \geq \frac{-\log \det(M^*) + d \log \tau}{Y}. \quad (5.18)$$

Taking logs in the second inequality of Equation (5.17) and  $\tilde{\varepsilon} = d^{-C}$ , we have

$$\begin{aligned} \log \det(M^*) &\leq \log(2^d \tilde{\varepsilon}^{(2-2\delta-\gamma)d}) \leq d \log 2 + (2-2\delta-\gamma) d \log \tilde{\varepsilon} \\ &= -(2-2\delta-\gamma) d C \log d + O(d). \end{aligned}$$

Thus, we have  $-\log \det(M^*) \geq (2-2\delta-\gamma) d \cdot C \log d - O(d)$ . We have  $Y = \log d - (\log(c\tilde{\varepsilon}^2) - \log \tau) = (2C+1) \log d + O(1)$ . Therefore, by Equation (5.18), we have for sufficiently large  $d$ ,

$$\hat{k}_d \geq \frac{(2-2\delta-\gamma) d \cdot C \log d - O(d)}{(2C+1) \log d + O(1)} \geq \left(1 - \delta - \frac{\gamma}{2}\right) \frac{C}{C+1} d - o(d). \quad (5.19)$$

Then, we have

$$\dim(\hat{T}) \leq d - \hat{k}_d \leq \left(\delta + \frac{\gamma}{2} + \frac{1}{C}\right) d + o(d). \quad (5.20)$$

Since  $\gamma \in (0, \delta)$ , for sufficiently large  $C > 5/\delta$ , we have  $\dim(\hat{T}) \leq 7/4 \cdot \delta d$ . Thus, we have at least  $(1 - \alpha')$  fraction of the points in  $A$  are  $\varepsilon_0$  close to this subspace  $\hat{T}$  with dimension less than  $7/4 \cdot \delta d$ .

We now show that for these points  $a_i$  that are close to this subspace  $\hat{T}$ , a large fraction of the points  $u_i$  before random perturbation lie exactly in a subspace with dimension at most  $2\delta d$ . Since  $a_i = (u_i + z_i)/\|u_i + z_i\|$ , we have

$$\|a_i - u_i\| \leq \|u_i - (u_i + z_i)\| + \|(u_i + z_i) - a_i\| \leq \|z_i\| + |1 - \|u_i + z_i\|| \leq 2\|z_i\|.$$

Thus, we have these  $u_i$  are  $\varepsilon_0 + 2\tilde{\varepsilon}$  close to the subspace  $\hat{T}$ . In Lemma 5.4.4 below, we show that when many points are close to a subspace in this instance, then there exists a subspace that exactly contains a large portion of these points. By Lemma 5.4.4 and choosing  $\theta^2 = 7/8$ , we get a subspace with dimension at most  $2\delta d$  that contains at least  $(1 - \alpha')n - 40\delta d$  points

in  $U$ . Let  $\alpha'' = \alpha' + 40\delta d/n = (1 + (1 - 9\varepsilon)\delta)\alpha$ .

Thus, we have there is a subspace  $\bar{S} \subset \mathbb{R}^d$  with  $\dim(\bar{S}) \leq 2\delta d$  that contains at least  $(1 - \alpha'')$  of the points in  $A$ . This contradicts the NO case in [Theorem 5.4.3](#). Therefore, every ellipsoid covering  $(1 - \alpha')n$  points has volume at least  $\beta^{\gamma d}K$ .  $\square$

**Lemma 5.4.4** (Exact containment from many near points). *Let  $S \subset \mathbb{R}^d$  be an  $r$ -dimensional subspace and set  $P := P_S$  to be the projection matrix onto the subspace  $S$ . Fix  $\varepsilon \in (0, 1/4)$  and define the set of  $\varepsilon$ -close edges*

$$F := \{\{i, j\} \in E : \|(I - P)u_{ij}\|_2 \leq \varepsilon\},$$

where  $u_{ij} = (e_i + e_j)/\sqrt{2}$ . Let  $\theta \in (0, 1)$  and define the heavy index set  $J := \{i : P_{ii} \geq \theta^2\}$ .

Then:

(a)  $|J| \leq r/\theta^2$  (since  $\sum_i P_{ii} = \text{tr}(P) = r$ ).

(b) Let  $t := \frac{1-\theta^2}{2} - \varepsilon^2 \geq 0$ . Then all but at most  $(r/t^2)$  edges of  $F$  have both endpoints in  $J$ . Consequently,

$$|\{\{i, j\} \in F : i, j \in J\}| \geq |F| - \frac{r}{t^2}.$$

In particular, the coordinate subspace  $S_J := \text{span}\{e_k : k \in J\}$  has  $\dim(S_J) = |J| \leq r/\theta^2$  and contains exactly all edges in  $F \cap E(J, J)$ .

*Proof.* Part (a) is immediate from  $\sum_i P_{ii} = r$ .

For (b), take any edge  $\{i, j\} \in F$  with  $i \notin J$  (so  $P_{ii} < \theta^2$ ). Since

$$\|(I - P)v_{ij}\|_2^2 = 1 - \|Pv_{ij}\|_2^2 = 1 - \frac{1}{2}(P_{ii} + P_{jj} + 2P_{ij}) \leq \varepsilon^2,$$

we obtain

$$P_{ii} + P_{jj} + 2P_{ij} \geq 2(1 - \varepsilon^2).$$

Using  $P_{ii} < \theta^2$  and  $P_{jj} \leq 1$ ,

$$2P_{ij} \geq 2(1 - \varepsilon^2) - P_{ii} - P_{jj} \geq 2(1 - \varepsilon^2) - (1 + \theta^2),$$

hence  $P_{ij} \geq t := \frac{1-\theta^2}{2} - \varepsilon^2$ . For this fixed  $i \notin J$ , the number of neighbors  $j$  with  $P_{ij} \geq t$  is at most

$$\frac{1}{t^2} \sum_{j=1}^d P_{ij}^2 = \frac{P_{ii}}{t^2},$$

where the equality is due to  $P^2 = P$  for any projection matrix. Summing over all  $i \notin J$  bounds the number of edges in  $F$  with at least one light endpoint:

$$|F \setminus E(J, J)| \leq \sum_{i \notin J} \frac{P_{ii}}{t^2} \leq \frac{1}{t^2} \sum_{i=1}^d P_{ii} = \frac{r}{t^2}.$$

The remaining edges in  $F$  lie in  $E(J, J)$ , and each such vector  $v_{ij}$  is supported in  $J$ , hence lies exactly in  $S_J$ . □

## 5.5 Application to Convex Confidence Sets and Conformal Prediction

### 5.5.1 Convex confidence sets

Our routine for approximating the optimal confidence ellipsoid of a given aspect ratio  $\beta$ , naturally gives a guarantee for approximating the optimal *convex confidence set* of a distribution. This is because any convex set is well-approximated by an ellipsoid (via John's theorem), and



indeed any well-conditioned convex set is well-approximated by a well-conditioned ellipsoid [John, 1948].

This highlights an interesting aspect of the problem of learning confidence sets. We know that for a family  $\mathcal{F}$  of confidence sets,  $\mathcal{F}$  having bounded VC dimension is sufficient to ensure that we can statistically efficiently learn an approximate best confidence set in  $\mathcal{F}$ . However, bounded VC dimension is not a necessary condition! Indeed the family of convex sets (and even the family of well-conditioned convex sets) has infinite VC-dimension, even in 2 dimensions.

Bounded VC-dimension is not necessary to achieve coverage, because it is a qualitatively different goal than achieving low binary classification error. To achieve low binary classification error, it is important to include the positive examples in the set, and exclude the negative examples from your set. However, to achieve coverage, it is only important that enough points are included in the set, and there is no penalty for including extra points. Another way of seeing this is that the convex set that we are trying to approximate may have significantly different coverage on the population than on the empirical sample. However, it is well-approximated by an ellipsoid, and that ellipsoid must have similar coverage on the population and the sample via uniform convergence.

**Definition 5.5.1** (Aspect ratio of a convex body). *Let  $X \subset \mathbb{R}^d$  be a convex body with nonempty interior. Denote by*

$$r(X) := \max_{x \in \mathbb{R}^d} \sup\{r > 0 : B(x, r) \subseteq X\}, \quad R(X) := \min_{x \in \mathbb{R}^d} \inf\{R > 0 : X \subseteq B(x, R)\},$$

*the inradius and circumradius of  $X$ , respectively, where  $B(x, r)$  denotes the Euclidean ball*

of radius  $r$  centered at  $x$ . The aspect ratio of  $X$  is

$$\kappa(X) := \frac{R(X)}{r(X)} \geq 1.$$

**Theorem 5.5.2** (Competing with convex confidence sets). *Let  $A = \{a_1, \dots, a_n\} \subset \mathbb{R}^d$ . Fix parameters  $\alpha \in (0, 1)$  and  $\gamma \in (0, 1)$ . There exists a universal constant  $c > 0$  and a polynomial-time algorithm that outputs an ellipsoid  $\hat{E} \subset \mathbb{R}^d$  with the following property:*

*For the minimum volume convex set  $K^* \subset \mathbb{R}^d$  with  $|K^* \cap A| \geq (1 - \alpha)n$ , let  $\kappa(K^*)$  be the aspect ratio of  $K^*$ . Then the output  $\hat{E}$  satisfies*

$$\text{vol}(\hat{E})^{1/d} \leq t_d^{1+\gamma} \kappa(K^*)^\gamma \text{vol}(K^*)^{1/d}, \quad \text{and} \quad |\hat{E} \cap A| \geq \left(1 - \frac{c\alpha}{\gamma}\right)n, \quad (5.21)$$

where  $t_d = \sqrt{d}$  if  $K^*$  is centrally symmetric; otherwise  $t_d = d$ .

*Proof.* Let  $E_J$  be the John ellipsoid of  $K^*$ . By John's theorem,  $E_J \subseteq K^* \subseteq t_d E_J$  with  $t_d = d$  in general and  $t_d = \sqrt{d}$  if  $K^*$  is centrally symmetric. Set  $E' = t_d E_J$ . Then  $E'$  has the same condition number as  $E_J$  (scaling does not change condition number), and  $K^* \subseteq E'$ , hence  $|E' \cap A| \geq |K^* \cap A| \geq (1 - \alpha)n$ .

Apply [Theorem 5.1.3](#) to the inlier witness ellipsoid  $E'$ . Let  $\beta_J$  be the condition number of the ellipsoid  $E_J$ . Since  $E'$  is  $\beta_J$ -conditioned and covers at least  $(1 - \alpha)n$  points, [Theorem 5.1.3](#) yields an ellipsoid  $\hat{E}$  with

$$\text{vol}(\hat{E})^{1/d} \leq \beta_J^\gamma \text{vol}(E')^{1/d} = \beta_J^\gamma t_d \text{vol}(E_J)^{1/d} \leq \beta_J^\gamma t_d \text{vol}(K^*)^{1/d},$$

using  $\text{vol}(E_J) \leq \text{vol}(K^*)$  because  $E_J \subseteq K^*$ . The inlier bound  $|\hat{E} \cap A| \geq \left(1 - \frac{c\alpha}{\gamma}\right)n$  is inherited directly from [Theorem 5.1.3](#).

Let  $\lambda_{\min}$  and  $\lambda_{\max}$  be the minimum and maximum eigenvalues of the shape matrix of  $E_J$ , respectively. Then,  $\sqrt{\lambda_{\min}}$  and  $\sqrt{\lambda_{\max}}$  are the minimum and maximum axes lengths of  $E_J$ . Since  $E_J \subseteq K^* \subseteq t_d E_J$ , we have  $R(K^*) \geq \sqrt{\lambda_{\max}}$  and  $t_d \sqrt{\lambda_{\min}} \geq r(K^*)$ . Thus, we have

$$\beta_J = \sqrt{\frac{\lambda_{\max}}{\lambda_{\min}}} \leq t_d \frac{R(K^*)}{r(K^*)} = t_d \kappa(K^*).$$

□

*Remark 5.5.3.* We will see in [Chapter 7](#) that the  $d^d$  volume approximation factor that arises from John's theorem is actually avoidable, and convex sets are volumetrically learnable (defined formally in [Chapter 7](#)) for volume approximation factor  $(1 + \varepsilon)$  for any  $\varepsilon > 0$ .

### 5.5.2 Conformal Prediction

A natural application of learning confidence sets is to conformal prediction. In conformal prediction, we want to construct a prediction set that covers an unseen test point with probability at least  $1 - \alpha$ , assuming only that the test point and the training (empirical) samples are *exchangeable*. Our algorithm for learning confidence ellipsoids directly implies an algorithm for conformal prediction as follows.

**Theorem 5.5.4** (Conformal Prediction with Approximate Volume Optimality). *Fix parameters  $\alpha \in (0, 1)$ ,  $\gamma \in (0, 1)$ , and a condition number upper bound  $\beta \geq 1$ . Let  $\mathcal{E}_\beta$  denote the class of all origin-centered ellipsoids in  $\mathbb{R}^d$  whose condition number is at most  $\beta$ . There is a polynomial-time conformal prediction algorithm that, given  $Y_1, \dots, Y_n \in \mathbb{R}^d$ , outputs a prediction set  $\widehat{C} \subseteq \mathbb{R}^d$  such that:*

(a) Marginal coverage. *If  $Y_1, \dots, Y_{n+1}$  are exchangeable, then*

$$\mathbb{P}[Y_{n+1} \in \widehat{C}] \geq 1 - \alpha.$$

(b) Approximate volume optimality versus  $\beta$ -conditioned ellipsoids. *For any  $\delta \in (0, 1)$ , whenever  $n = \Omega(d^2/\gamma^2)$ , the following holds with probability at least  $1 - \delta$  over the training sample:*

$$\text{vol}(\hat{C})^{1/d} \leq \beta^\gamma \text{vol}(C_\beta^*)^{1/d},$$

where  $C_\beta^*$  is the optimal  $\beta$ -conditioned ellipsoid achieving a slightly stronger coverage constraint, such that for a universal constant  $c > 0$

$$C_\beta^* \in \arg \min_{C \in \mathcal{E}_\beta} \text{vol}(C) \quad \text{s.t.} \quad \mathbb{P}_{Y \sim \mathcal{D}}[Y \in C] \geq 1 - c\alpha\gamma.$$

*Proof.* This is a direct consequence of our meta-theorem for volume-optimal conformal prediction, [Theorem 2.5.1](#), using our ellipsoid finding algorithm as a subroutine.  $\square$

## 5.6 Chapter Acknowledgements

This chapter is based on joint work with Chao Gao, Liren Shan, and Aravindan Vijayaraghavan. We thank Mohit Singh, Hamed Hassani and Xue Chen for helpful discussions.

# Chapter 6

## Online Conformal Prediction

This chapter is based on *Online Conformal Prediction with Efficiency Guarantees*, which appeared at the Symposium on Discrete Algorithms (SODA) 2026 [[Srinivas, 2026](#)].

In this chapter, we study the problem of generating small prediction sets *online*, where the goal is to achieve coverage on average over an arbitrary sequence of target variables  $y_1, \dots, y_t \mathbb{Y}$ , while playing sets of volume that is close to the smallest fixed set that achieves coverage in hindsight. We focus on the simplest setting where  $\mathbb{Y} = [0, 1] \subset \mathbb{R}$  is the unit interval. We see that this problem admits very different guarantees than its offline counterparts, and minimizing volume requires significantly new techniques.

### 6.1 Introduction

Recall the standard *distribution-free* setting of conformal prediction from [Chapter 3](#). We state this in the unsupervised setting, with no features  $X$ .

**Problem 2.1.1** (Conformal prediction (marginal coverage)).

Let  $(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1}) \in \mathcal{X} \times \mathcal{Y}$  be exchangeable random variables. That is, the data points  $(X_i, Y_i)$  are in uniformly random order (formally [Definition 2.1.2](#)).

A conformal predictor that achieves marginal coverage takes

- a target miscoverage rate  $\alpha \in (0, 1]$
- a black-box model  $\hat{Y} = f(X)$  to predict target variable  $Y \in \mathcal{Y}$  from feature variable  $X \in \mathcal{X}$ ,
- calibration data  $(X_1, Y_1), \dots, (X_n, Y_n)$ , and
- the test feature  $X_{n+1}$ ,

and outputs a prediction set  $C = C((X_1, Y_1), \dots, (X_n, Y_n), X_{n+1}) \subseteq \mathcal{Y}$  such that

$$\mathbb{P}_{(X_1, Y_1), \dots, (X_{n+1}, Y_{n+1})} [Y_{n+1} \in C] \geq 1 - \alpha. \quad (\text{MC})$$

In addition, we would like the conformal predictor to be efficient, in that it outputs prediction sets  $C$  that are as small as possible.

We will focus on the most basic setting where  $\mathcal{Y} = [0, 1]$ .

The reason that conformal prediction is built on exchangeability is that it is considered to be the weakest possible assumption that still allows one to make non-trivial predictions that are statistically valid. However, while exchangeability is much more general than, say, assuming the data are drawn i.i.d., it still rules out many scenarios that arise in practice. For example, if the data are time-varying, with even a minor distribution shift, the  $Y_i$  are no longer exchangeable, and standard conformal guarantees are no longer valid. This motivates our main question.

**Question 6.1.1.** *Is it possible to design a provably efficient conformal predictor for fully arbitrary data?*

It is clear that the type of “one step” guarantee that is typically studied in conformal prediction is impossible. If the  $Y_i$  are fully arbitrary, then a conformal predictor essentially has to output the whole space  $\mathcal{Y}$ .<sup>1</sup> However, in an online setting, we can aim to instead achieve coverage *on average* over time.

**Problem 6.1.2** (Efficient Online Conformal Prediction). *Let  $T \in \mathbb{N}$  be a known time horizon. On each day  $t \leq T$ , an algorithm  $\mathcal{A}$  must output a set  $C_t \subseteq [0, 1]$ . Then, an arbitrary  $y_t \in \mathcal{Y}$  is revealed to  $\mathcal{A}$ . We will refer to the (potentially adaptive) input sequence as  $S = (y_1, \dots, y_T)$ .*

*We define the coverage of  $\mathcal{A}$  on  $S$  in an average sense as*

$$\text{coverage}_{\mathcal{A}}(S) = \frac{1}{T} \sum_{i=1}^T \mathbf{1}[y_t \in C_t]. \quad (6.1)$$

*We will also refer to the number of points that  $\mathcal{A}$  does not cover as the number of mistakes that  $\mathcal{A}$  makes (where we want this to be close to  $\alpha T$ ).*

*We measure the average volume (or Lebesgue measure) of the intervals played by  $\mathcal{A}$ ,*

$$\text{volume}_{\mathcal{A}}(S) = \frac{1}{T} \sum_{i=1}^T \text{volume}(C_t).$$

*The goal of  $\mathcal{A}$  is to simultaneously achieve*

$$\text{coverage}_{\mathcal{A}}(S) \gtrsim 1 - \alpha, \quad \text{for all sequences } S,$$

---

<sup>1</sup>More formally, the predictor has to cover at least a  $(1 - \alpha)$ -fraction of  $[0, 1]$  in expectation, which results in a very large prediction set.

and achieve low volume approximation (competitive ratio)  $\mu$ , where

$$\mu \geq \frac{\text{volume}_{\mathcal{A}}(S)}{\text{OPT}_S(\alpha)}, \quad \text{for all sequences } S.$$

In this chapter we will refer to the typical “one step” problem as *standard conformal prediction*, and this online learning framework as *online conformal prediction*. We note that there are other works that consider conformal prediction in an online setting, notably the line of work started by [Gibbs and Candès \[2025\]](#). However, they do not provide provable efficiency guarantees, and in fact we show that the strategies developed in previous work *cannot* achieve an efficiency guarantee in the sense of a small volume approximation factor  $\mu$ . We provide a more in-depth comparison in the related work ([Section 6.1.3](#)).

### 6.1.1 Results

We will study the problem for *arbitrary order* sequences, and *exchangeable sequences*. Recall the definition of exchangeable sequences.

**Definition 2.1.2** (Exchangeability). *A sequence  $(Z_1, \dots, Z_n)$  is exchangeable if*

$$(Z_1, \dots, Z_T) \stackrel{d}{=} (Z_{\sigma(1)}, \dots, Z_{\sigma(T)}),$$

*for all permutations  $\sigma : [T] \rightarrow [T]$ , where  $\stackrel{d}{=}$  means they are equal in distribution.*

This is also referred to as the *random order* model in online algorithms. Note that the class of exchangeable sequences generalizes the class of i.i.d. sequences, as i.i.d. sequences are always exchangeable, but an exchangeable sequence need not be i.i.d..

We observe that this problem can be seen as a *constrained optimization* analogue of the classical *online learning* problem (see [Section 6.1.2](#) for discussion). A beautiful property of



online learning is that going from i.i.d. sequences to arbitrary sequences is essentially without loss— the same optimal bounds can be recovered in both cases. This motivates the following question for our setting.

**Question 6.1.3.** *Is it possible to design an algorithm for fully arbitrary data that recovers the best possible bounds for exchangeable data?*

Surprisingly, the conformal prediction problem exhibits very different trade-offs for these two settings. We establish near-optimal bounds for both the arbitrary and exchangeable settings. We also present a “*no best-of-both worlds*” result, that shows that we cannot hope to design one single algorithm that achieves the best possible guarantee in both the arbitrary and exchangeable settings.

**Results for arbitrary sequences.** Our first result is an algorithm for arbitrary sequences, which achieves a given volume approximation  $\mu$  while making a bounded number of mistakes.

**Theorem 6.1.4** (Informal version of [Corollary 6.2.2](#)). *For a given scale lower bound  $\text{minwidth} > 0$ , multiplicative volume approximation  $\mu > 3$ , target miscoverage rate  $\alpha \geq 0$ , and time horizon  $T$ , we give an algorithm that on any sequence  $S$  of length  $T$  plays intervals of maximum volume (and therefore average volume)*

$$\leq \mu \max\{\text{OPT}_S(\alpha), \text{minwidth}\},$$

*and makes number of mistakes bounded by*

$$O\left(\frac{\log(1/\text{minwidth})}{\log(\mu)}(\alpha T + 1)\right).$$

*Moreover, the algorithm is deterministic and therefore robust to an adaptive adversary.*

This tells us that, for example, it is possible to achieve a constant volume approximation

**Algorithm 6.1** Meta-algorithm for Online Conformal Prediction

---

**Require:** scale lower bound `minwidth`  $> 0$ , multiplicative volume approximation  $\mu \geq 1$ ,

- 1: allowable error rate function  $R(t) : \{0, \dots, T-1\} \rightarrow [0, 1]$
- 2:  $I_{\text{current}} \leftarrow [0, 0]$
- 3: **for** day  $t$  **do**
- 4:   **if**  $I_{\text{current}}$  has empirical coverage  $< 1 - R(t-1)$  over  $\{y_{t'} : t' < t\}$  **then**
- 5:      $\mathcal{F}_t :=$  set of intervals that achieve coverage  $\geq 1 - R(t-1)$  over  $\{y_{t'} : t' < t\}$
- 6:     // set of *feasible* intervals on day  $t$
- 7:      $I_t :=$  arbitrary minimum volume interval  $\in \mathcal{F}_t$
- 8:      $\hat{\mu} := \mu \max\{1, \frac{\text{minwidth}}{\text{vol}(I_t)}\}$
- 9:     // ensure the interval will have volume  $\geq \text{minwidth}$
- 10:     $I_{\text{current}} \leftarrow \hat{\mu} I_t$
- 11:    // for  $s > 0$ , interval  $I = [a, b]$ , define  $sI = \{x \subseteq \mathbb{R} : |x - \frac{a+b}{2}| \leq s \cdot \frac{b-a}{2}\}$
- 12:   **end if**
- 13:   **play**  $I_{\text{current}} \cap [0, 1]$
- 14:    $y_t$  is revealed
- 15: **end for**

---

$\mu = 5$ , and have number of mistakes bounded by  $O(\log(1/\text{minwidth})\alpha T)$ . The algorithm is simple, and is in fact an instantiation of a meta-algorithm, [Algorithm 6.1](#), that we can apply to the exchangeable setting as well. [Algorithm 6.1](#) maintains an interval  $I_{\text{current}}$  that has achieved error rate at most  $R(t)$  by day  $t$ , where for the arbitrary order setting we choose  $R(t) = \alpha \frac{T}{t}$ . If on any day,  $I_{\text{current}}$  no longer meets this requirement, [Algorithm 6.1](#) resets  $I_{\text{current}}$  to be a  $\mu$ -scaling of the smallest volume interval that does achieve the error requirement. The analysis for the arbitrary order case shows that, while this is not explicitly enforced by the algorithm, this procedure ends up doing a doubling search for the scale of the optimal interval.

We highlight that the analysis is not specific to the choice of  $R(t)$ , and different choices of  $R(t)$  achieve different trade-offs between  $\mu$  and the number of mistakes. In the introduction, we present the result for the optimal choice of  $R(t)$  for clarity, and we refer the reader to [Theorem 6.2.1](#) for full details.

Our algorithm exhibits tradeoff in the multiplicative volume approximation and the multiplicative approximation in number of mistakes. We show that this tradeoff is essentially tight.

**Theorem 6.1.5** (Informal version of [Theorem 6.4.1](#)). *For any scale lower bound  $\text{minwidth} > 0$ , miscoverage rate  $\alpha > 0$ , and time horizon  $T$ , there is a set  $\mathcal{S}$  of input sequences of length  $T$  chosen oblivious to any algorithm, such that for any potentially randomized algorithm  $\mathcal{A}$  that outputs confidence sets over  $[0, 1]$  that are not necessarily intervals,*

1. *If  $\mathcal{A}$  plays sets with expected average volume  $\leq \mu_{\text{avg}} \max\{\text{OPT}_{S_i}(\alpha), \text{minwidth}\}$  on every sequence  $S_i \in \mathcal{S}$ , for some value  $\mu_{\text{avg}} > 0$ , then  $\mathcal{A}$  must make*

$$\text{for any } \varepsilon' > 0, \quad \Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\text{avg}})} \alpha^{1+\varepsilon'} T, T \right\} \right) \text{ mistakes}$$

*in expectation on some sequence  $S_j \in \mathcal{S}$ .*

2. *If  $\mathcal{A}$  plays sets with expected maximum volume  $\leq \mu_{\text{max}} \max\{\text{OPT}_{S_i}(\alpha), \text{minwidth}\}$  on any sequence  $S_i \in \mathcal{S}$ , for some value  $\mu_{\text{max}} > 0$ , then  $\mathcal{A}$  must make*

$$\Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\text{max}})} \cdot \alpha T, T \right\} \right) \text{ mistakes}$$

*in expectation on some sequence  $S_j \in \mathcal{S}$ .*

Recall that our algorithm is deterministic, robust to an adaptive adversary, proper (it outputs sets that are intervals), and it has a bound on the volume of the largest interval it ever outputs. This lower bound tells us that it is essentially optimal, even among algorithms that are randomized, only robust to an oblivious adversary, improper (output sets that are not intervals), and have a bound only on the *average* volume of sets that they play.

**Results for exchangeable sequences.** Since conformal prediction is typically studied in the setting where the data is exchangeable, it is natural to consider this in our online formulation as well. Note that the guarantees we establish for this setting are incomparable to the standard “one step” guarantees of conformal prediction: we will establish coverage on average over time (6.1), as opposed to statistical validity on day  $T$  alone (MC).

We show that the same meta-algorithm Algorithm 6.1, instantiated with  $\mu = 1$  and a different choice of  $R(t) = \alpha - O(\sqrt{\log T/T})$  can achieve an optimal guarantee for exchangeable data.

**Theorem 6.1.6** (Informal version of Corollary 6.3.2). *For a given scale lower bound  $\text{minwidth} > 0$ , miscoverage rate  $\alpha$ , and time horizon  $T$ , we have an algorithm that, on an exchangeable sequence  $\mathbf{S}$ , with probability  $\geq 1 - \frac{1}{100}$  over the exchangeability of  $\mathbf{S}$ , plays intervals of maximum volume*

$$\leq \max\{\text{OPT}_{\mathbf{S}}(\alpha), \text{minwidth}\},$$

*and achieves expected coverage  $\geq (1 - \alpha) - O(\sqrt{\log T/T})$ .*

This bound is essentially optimal, as the vanishing error in coverage  $O(\sqrt{\log T/T})$  is what we would get due to sampling error when the sequence is drawn i.i.d., which is a special case of exchangeability.

Similarly to the arbitrary case, our analysis is not specific to the choice of  $R(t)$ , and different choices of  $R(t)$  achieve different trade-offs between  $\mu$  and the number of mistakes. In the introduction, we present the result for the optimal choice of  $R(t)$  for clarity, and we refer the reader to Theorem 6.3.1 for full details.

As a corollary to this analysis, we show that the interval that Algorithm 6.1 plays on day  $T/2$

is actually a statistically valid and volume optimal conformal set for  $y_T$  (Corollary 6.3.3). We note that this guarantee can also be seen as a consequence of Theorem 3.2.6 from Chapter 3, along with uniform convergence for exchangeable sequences (Lemma 2.8.1). We view this observation as evidence that the online conformal prediction problem is more general than the offline version, in a similar spirit to how online learning is a generalization of the offline PAC learning problem.

**“Best of both worlds.”** Since our optimal algorithms for both the arbitrary order setting and the exchangeable setting are instantiations of the same meta-algorithm Algorithm 6.1, it is natural to ask whether there is a single algorithm that simultaneously achieves optimal bounds in both settings. That is, is there a single algorithm  $\mathcal{A}$  such that, on arbitrary inputs  $\mathcal{A}$  achieves a given Pareto optimal tradeoff in  $\mu$  and number of mistakes, and on exchangeable inputs  $\mathcal{A}$  achieves the better tradeoff of  $\mu = 1$  and coverage  $(1 - \alpha) - o(1)$ ? We show that this is indeed not the case.

**Theorem 6.1.7** (Informal version of Theorem 6.5.1). *Fix a scale lower bound  $\text{minwidth} > 0$ , target miscoverage rate  $\alpha < \frac{1}{4}$ , and multiplicative volume approximation  $\mu > 1$ . Take any (potentially randomized) algorithm  $\mathcal{A}$  that on any arbitrary order sequence  $S$  plays intervals of expected average volume*

$$\leq \mu \max\{\text{OPT}_S(\alpha), \text{minwidth}\}.$$

*Then,  $\mathcal{A}$  must make*

$$\Omega\left(\min\left\{\frac{\log(1/\text{minwidth})}{\log(\mu)}, \log(1/\alpha)\right\} \alpha T\right)$$

*mistakes in expectation on some i.i.d. sequence  $S'$ .*

That is, an algorithm that achieves a non-trivial guarantee on every arbitrary sequence must incur a multiplicative overhead in the number of mistakes it makes on i.i.d., and therefore exchangeable sequences.

On the positive side, we show that [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  meets this mistake bound on exchangeable sequences ([Theorem 6.3.4](#)). Recall that this is the same setting of  $R(t)$  that gives an optimal algorithm for arbitrary sequences. Thus, this algorithm is optimal for arbitrary sequences, and gets the best possible mistake bound subject to achieving a guarantee for arbitrary sequences. We remark that [Algorithm 6.1](#) does incur the same multiplicative volume approximation of  $\mu$  for both arbitrary and exchangeable sequences, whereas the lower bound does not rule out the case that a single algorithm can achieve a better volume approximation for exchangeable sequences than arbitrary sequences.

### 6.1.2 Technical Overview

**Contrast to online learning.** This problem is reminiscent of classical online learning. Think of each interval  $I$  as an “expert.” In fact, there are infinitely many intervals, and this poses a genuine issue, but for now, assume there  $n < \infty$  experts, and we will justify this assumption later. On day  $t$ , the expert corresponding to interval  $I$  has loss 0 if  $y_t \in I$  and loss 1 if  $y_t \notin I$ . That is, the loss of the expert is exactly the coverage of the corresponding interval. The classical online learning problem is to pick an expert on each day to achieve average loss that competes with the loss of the best fixed expert in hindsight. For our particular setting, this is trivial, as there is an expert in our set that corresponds to an interval that covers the whole space, and achieves loss 0 on every day, but the online learning problem is of course much more general.

In our problem, instead of loss being an objective that we are trying to minimize, loss

becomes a constraint: we want to achieve average loss  $\leq \alpha$  over the sequence. The objective that we are trying to minimize is an average over some other function over the experts that we choose, for us volume of the corresponding interval. In this sense, our problem becomes an online *constrained optimization* problem, where the set of experts that are feasible with respect to the constraint in hindsight depends on the input sequence.

One of the beautiful properties of classical online learning is that it is possible to design algorithms for arbitrary input sequences that have vanishing regret, i.e., they make  $1 + O(\sqrt{\log n/T})$  times as many errors as the best fixed expert in hindsight (see e.g., [Blum, Hopcroft, and Kannan \[2020\]](#)). This vanishing regret term matches the sampling error that one would face even for i.i.d. sequences. In essence, this says that going from i.i.d. (and therefore exchangeable) data to arbitrary data is without loss, as we are able to achieve the best possible guarantee even in the arbitrary case. Hence there is one algorithm that is able to achieve the “best of both worlds.” Such results are also known in bandit problems [[Bubeck and Slivkins, 2012](#)], online conformal prediction (without efficiency guarantees) [[Angelopoulos, Barber, and Bates, 2024](#)], and other settings.

It turns out that, in contrast to classical online learning, our problem exhibits a large gap between what is possible in the arbitrary setting and the i.i.d. setting. For exchangeable data, we show that it is possible to design an algorithm that has volume approximation  $\mu = 1$ , while having a vanishing fraction of additional mistakes, i.e., the algorithm has coverage  $(1 - \alpha) - \tilde{O}(\sqrt{\log n/T})$ , where we will make what  $n$  means in this setting precise shortly, and  $\tilde{O}$  hides subpolynomial factors in  $T$ . In the arbitrary setting however, we show that any algorithm, even one that is randomized, only robust to an oblivious adversary, and improper (outputs sets that are not intervals), faces a stark tradeoff between volume approximation  $\mu$  and how many times more than  $\alpha T$  mistakes it makes. Additionally, we show that even subject to this lower bound, there can be no algorithm that is simultaneously optimal for

the exchangeable setting and Pareto optimal for the arbitrary setting. This rules out any “best of both worlds” guarantee.

On the positive side, we design a matching algorithm that is able to achieve any Pareto optimal tradeoff. This algorithm is deterministic, which is also in contrast to the online learning problem which exhibits a separation between deterministic and randomized algorithms. Furthermore, our algorithms for exchangeable and arbitrary sequences fit in a unified framework that allows us to interpolate between being optimal for one and optimal for the other.

**Challenges for arbitrary sequences.** Designing an algorithm for arbitrary input sequences presents some challenges. Consider the following example. Our goal is to design a conformal predictor for miscoverage rate  $\alpha = \frac{1}{2}$ . There are two sequences. Let  $\varepsilon > 0$  be some small parameter.  $S^{(1)}$  is drawn uniformly from  $[0, \varepsilon]$  for the first half of the sequence, and then expands in scale to be drawn uniformly from  $[0, 1]$  for the second half of the sequence.  $S^{(2)}$  is also drawn uniformly from  $[0, \varepsilon]$  for the first half of the sequence, but then shrinks in scale to be drawn uniformly from  $[0, \varepsilon^2]$  for the second half of the sequence. That is,

$$\begin{aligned} S^{(1)} = (S_1^{(1)}, \dots, S_T^{(1)}), \quad S_t^{(1)} &\sim \begin{cases} \text{Unif}[0, \varepsilon] & \text{for } t \leq T/2 \\ \text{Unif}[0, 1] & \text{for } t > T/2 \end{cases}, \\ S^{(2)} = (S_1^{(2)}, \dots, S_T^{(2)}), \quad S_t^{(2)} &\sim \begin{cases} \text{Unif}[0, \varepsilon] & \text{for } t \leq T/2 \\ \text{Unif}[0, \varepsilon^2] & \text{for } t > T/2 \end{cases}. \end{aligned}$$

Since half of  $S^{(1)}$  is at scale  $\varepsilon$ , and half of  $S^{(2)}$  is at scale  $\varepsilon^2$ , we have

$$\text{OPT}_{S^{(1)}}(\alpha) \leq \varepsilon, \quad \text{OPT}_{S^{(2)}}(\alpha) \leq \varepsilon^2.$$



Any algorithm  $\mathcal{A}$  that captures any constant fraction  $c > 0$  of points must incur a large multiplicative error in volume.

Since  $S^{(1)}$  and  $S^{(2)}$  are the same for the first  $T/2$  days,  $\mathcal{A}$  must capture at least a  $c/2$ -fraction of points in the first half of both sequences, or in the second half of both sequences. If  $\mathcal{A}$  captures a  $c/2$ -fraction of points on the first half of the sequences, then on sequence  $S^{(2)}$ ,  $\mathcal{A}$  has large average volume, and thus large volume approximation:

$$\text{volume}_{\mathcal{A}}(S^{(2)}) \geq \frac{c\varepsilon}{4} \implies \mu \geq \frac{\text{volume}_{\mathcal{A}}(S^{(2)})}{\text{OPT}_{S^{(2)}}(\alpha)} \geq \frac{c}{4\varepsilon}.$$

However, if  $\mathcal{A}$  captures a  $c/2$ -fraction of points on second half of the sequences, then on sequence  $S^{(1)}$ ,  $\mathcal{A}$  has large volume and thus large volume approximation:

$$\text{volume}_{\mathcal{A}}(S^{(1)}) \geq \frac{c}{4} \implies \mu \geq \frac{\text{volume}_{\mathcal{A}}(S^{(1)})}{\text{OPT}_{S^{(1)}}(\alpha)} \geq \frac{c}{4\varepsilon}.$$

This example highlights two issues. The first is a recurring issue in online learning. As  $\varepsilon \rightarrow 0$ , our volume approximation becomes unbounded. This arises because the aspect ratio of our problem is unbounded. Because the problem is scale-free, a non-trivial algorithm that outputs intervals can be forced to make an unbounded number of mistakes. We address this by introducing a parameter `minwidth`  $> 0$ , and require an algorithm to compete with the smallest volume interval that achieves coverage, among intervals that have volume at least `minwidth`. This, along with the fact that  $\mathcal{Y}$  is the bounded interval  $[0, 1]$ , fixes the aspect ratio of the problem. In essence, this tells us that we may as well think of a grid of size  $1/\text{minwidth}$  over the interval  $[0, 1]$ , and restrict the problem to considering intervals that are between two gridpoints. This reduces the number of intervals we have to consider to  $O(1/\text{minwidth}^2)$ , which we should think of as analogous to the number of experts  $n$  in the earlier analogy to

online learning. Similarly to online learning, we think of  $n \approx (1/\text{minwidth})^2$  as very large, and aim for guarantees that depend on  $\log n \approx \log(1/\text{minwidth})$ .

However, there is a more significant issue. Even if `minwidth` is fixed, the above example still forces any algorithm that captures a constant fraction of points to incur a multiplicative error  $\Omega(1/\text{minwidth})$  in volume, which one should think of as being extremely large. This issue arises because we require the algorithm to achieve coverage very close to the target, i.e., a non-trivial guarantee for  $\alpha = \frac{1}{2}$  requires the algorithm to make  $< T = 2\alpha T$  mistakes. Surprisingly, the problem actually becomes tractable for smaller values of  $\alpha$  if we consider a *bicriteria approximation*, that competes with the best interval that makes at most  $\alpha T$  mistakes, while making a multiplicative factor more than  $\alpha T$  mistakes.

Algorithmically, this example illustrates the difference between online learning and our problem. In online learning, on day  $t$  an algorithm can choose an expert that does well locally, that is, does well around time  $t$ . While this may not be the best expert over the whole sequence, the fact that it is doing as well as the best expert is doing right now, means that in every local time window, the algorithm is only performing as well or better than the best expert in hindsight. However, for our problem, choosing a different expert could incur a vastly different cost than the best expert. For example, if  $\mathcal{A}$  captures a constant fraction of days in the first half of the sequence, on  $S_2$  it only has coverage even higher than the best expert in hindsight, so it satisfies the coverage constraint. In exchange, however, it incurs much higher cost than the best expert. Thus on some days, an efficient algorithm must choose to abstain, and sacrifice coverage to minimize cost.

Put differently, suppose that each day's input  $y_t$  is drawn from some distribution  $\mathcal{D}_t$  (which are potentially different for each day). For each day  $t$ , define  $I_t^*$  to be the minimum volume interval that achieves coverage  $1 - \alpha$  over  $\mathcal{D}_t$ . We refer to the sequence of intervals  $(I_1^*, \dots, I_T^*)$

as the *daily optimum*. It is not necessarily the case that the daily optimum has smaller average volume than the best single interval that achieves average coverage  $1 - \alpha$  over the whole sequence of distributions  $\mathcal{D}_t$ :

$$\begin{aligned} \frac{1}{T} \sum_{i=1}^T \text{volume}(I_t^*) &= \frac{1}{T} \sum_{i=1}^T \min_{I_t} \text{volume}(I_t) \text{ s.t. } \mathbb{P}_{y \sim \mathcal{D}_t} [y \in I_t] \geq 1 - \alpha \\ &\not\leq \min_I \text{volume}(I) \text{ s.t. } \left( \frac{1}{T} \sum_{i=1}^T \mathbb{P}_{y \sim \mathcal{D}_t} [y \in I] \right) \geq 1 - \alpha. \end{aligned}$$

Thus the coverage constraint is a *global constraint* that requires an approach beyond trying to solve each individual day's problem. We discuss this further in the related work (Section 6.1.3).

To approach this problem, we show that Algorithm 6.1 essentially implements a doubling search for the volume of the optimal single interval that achieves global coverage  $1 - \alpha$ . In some sense, we can think of this as implementing a optimization to feasibility reduction, which is a classic idea in optimization. We would like to minimize volume, subject to a global coverage constraint. It turns out to be easy to instead maximize global coverage subject to a volume constraint. The doubling search then essentially performs a reduction from this second “feasibility” problem to the first “optimization” problem. This reduction requires a phases for each step of the doubling search, and therefore incurs some overhead. This leads to the multiplicative factor in the number of mistakes.

We are able to show that this overhead is necessary, by constructing a matching lower bound. As it turns out, the lower bound comes from exactly a “bad instance” for our algorithm, in which we show that no algorithm can do any better.

**Challenges for Exchangeable Sequences.** Our algorithm for exchangeable sequences is another instantiation of the same meta-algorithm [Algorithm 6.1](#). Essentially on each day  $t$ , the algorithm plays an interval  $I_t$ , which is the minimum volume interval that achieves coverage (up to some slack) over the previous days inputs  $S_1, \dots, S_{t-1}$ .

The analysis of the algorithm for exchangeable sequences is tricky because the interval  $I_t$  that [Algorithm 6.1](#) plays on day  $t$  is dependent on the first  $t-1$  days of the sequence  $S_1, \dots, S_{t-1}$ . This is in turn *not* independent of  $S_t$ . Thus the coverage on day  $t$

$$\mathbb{P}[S_t \in I_t(S_1, \dots, S_{t-1})],$$

where the probability is over the exchangeability of  $S$ , is challenging to bound. This is actually the exact setting of standard conformal prediction. However, existing methods only reason about coverage and do not come with provable volume guarantees, which are necessary for this problem.<sup>2</sup>

To understand the challenge, consider the following example. Let  $S$  be a uniformly random permutation of a multiset that has  $(1 - 2\alpha)T$  copies of  $1/2$ ,  $\alpha T$  copies of 0 and  $\alpha T$  copies of 1.  $S$  is exchangeable by construction. On day  $T$ , we have seen all but one element, which could have been  $1/2$  with probability  $1 - 2\alpha$ , 0 with probability  $\alpha$ , or 1 with probability  $\alpha$ .

[Algorithm 6.1](#), for  $\mu = 1$ ,  $R(t) = \alpha - O(\sqrt{\log T/T})$ , will arbitrarily choose a minimum volume interval that achieves coverage  $1 - R(T-1)$  on the first  $T-1$  points. The intervals  $[0, 1/2]$  or  $[1/2, 1]$  will both meet the coverage requirement and have equal (minimal) volume, so say that [Algorithm 6.1](#) tiebreaks by choosing the interval that achieved higher coverage, and choosing  $[0, 1/2]$  if they have equal coverage. Then, if  $S_T = 0$ ,  $[1/2, 1]$  will achieve higher

---

<sup>2</sup>To the best of our knowledge, all known volume optimality guarantees for standard conformal prediction require stronger assumptions than just exchangeability on the data. For example, they typically require the data to be drawn i.i.d.. We refer the reader to [Section 6.1.3](#) for more discussion.

coverage on the prefix and  $S_T$  will not be in the output interval. Similarly, if  $S_T = 1$ ,  $[0, 1/2]$  will achieve higher coverage on the prefix and  $S_T$  will not be in the output interval. Thus  $S_T$  is covered with probability  $\leq 1 - 2\alpha$ , when we are aiming for coverage  $1 - \alpha$ . Our analysis handles this subtle issue by showing that there cannot be many days  $t$  on which this issue arises. This allows us to amortize the loss in coverage by allowing for higher error rates both at the beginning and at the end of the sequence. Carefully accounting for this allows us to show that the standard analysis for i.i.d. sequences goes through even for sequences that are only exchangeable and not necessarily i.i.d..

A key lemma that allows us to get these bounds is the fact that *uniform convergence* for set families of bounded VC-dimension, for example intervals over  $[0, 1]$ , holds over not only i.i.d. inputs, but over exchangeable inputs as well ([Lemma 2.8.1](#)). While this is already known (we provide the writeup in [Section 2.8](#) for completeness), the contribution of this work is to use this to amortize error across the whole exchangeable sequence.

### 6.1.3 Related Work

Conformal prediction is a very well-studied area in statistics with a wealth of recent work. The theoretical work alone in this space is the topic of a recent textbook [[Angelopoulos, Barber, and Bates, 2025](#)]. We provide a brief comparison to some of the work most relevant to the setting we consider, and refer the reader to the textbook for a more in-depth treatment.

**Efficiency guarantees for standard conformal prediction.** Typically, conformal prediction methods are required to provably achieve coverage ([MC](#)). Efficiency, however, is often empirically validated on real and simulated data. There is a line of work that tries to establish efficiency guarantees subject to provably achieving coverage. The work of [Sadinle, Lei, and Wasserman \[2019\]](#), which considers the problem when the set  $\mathcal{Y}$  is a discrete label space.

The line of work including [Lei, Robins, and Wasserman, 2013, Izbicki, Shimizu, and Stern, 2020a,b] work in settings where the data is drawn i.i.d. from a distribution that allows for a good p.d.f. estimator, and then achieving a volume optimal prediction set corresponds to taking a level set of that estimator. However, this approach does not work for arbitrary distributions, that may in general be hard to approximate. In Chapter 3 [Gao, Shan, Srinivas, and Vijayaraghavan, 2025b], we show that when the data is drawn i.i.d., if one chooses a prediction set from a family of potential sets that has bounded VC-dimension, it is sufficient to find the most efficient prediction set that achieves coverage over the samples. To the best of our knowledge, all known volume optimality bounds for standard conformal prediction must make stronger assumptions on the data than just exchangeability.

**Beyond exchangeability.** A line of work has tried to relax the assumption of exchangeability for standard conformal prediction. A line of work including Tibshirani, Barber, Candès, and Ramdas [2019], Barber, Candès, Ramdas, and Tibshirani [2022] designs conformal methods that work when the joint distribution of the data is “approximately” exchangeable. Prinster, Stanton, Liu, and Saria [2024] design conformal methods when the joint distribution of the data is not exchangeable, but requires that the joint distribution is known in advance.

**Online conformal prediction.** For exchangeable sequences, a foundational result of Vovk [2002] shows that a standard conformal method when run repeatedly on prefixes of an online sequence, errs with probability that is independent on each day, and is therefore valid on every day. For arbitrary sequences, the problem of computing confidence sets that achieve coverage on average was initiated by Gibbs and Candès [2021], and studied further by Zaffran, Feron, Goude, Josse, and Dieuleveut [2022], Gibbs and Candès [2025], Bhatnagar, Wang, Xiong, and Bai [2023]. Feldman, Ringel, Bates, and Romano [2023], Angelopoulos, Candès,

and Tibshirani [2023a] give an algorithm that has coverage provably tending to  $1 - \alpha$  as  $T \rightarrow \infty$ . This model does not explicitly optimize for efficiency. Instead, the justification is to think of coverage as being a proxy for efficiency: by ensuring that coverage is not too much more than the target  $1 - \alpha$ , this incentivizes the algorithm to leave some things out of the prediction set.

To the best of our knowledge, ours is the first work to give provable efficiency guarantees for conformal prediction in the online setting, even for exchangeable data. In fact, our lower bound (Theorem 6.4.1) proves that any method that achieves coverage tending to  $1 - \alpha$  on arbitrary sequences must incur a very large multiplicative volume approximation in the worst case.

The algorithm of Angelopoulos, Candes, and Tibshirani [2023a] works by estimating the bottom  $\alpha/2$  and top  $\alpha/2$  quantile of the data in an online fashion. The confidence interval will then be the interval between these two estimated quantiles. This is possible since a given quantile is the minimizer of a convex function called the *pinball loss*, so it is possible to get a low regret estimator using *online convex optimization*. In our setting of optimizing volume, this runs into a couple of issues. The first is that it is subject to an “equal tails” assumption: it always outputs intervals that have approximately equal mass lying outside of the interval on each side. This may not be volume-optimal if the data distribution is skewed.

However, even if the data is subject to an equal tails constraint, where an input sequence must have each  $y_t$  drawn from a distribution  $\mathcal{D}_t$  that is unimodal and symmetric around median  $1/2$ , the lower bound in Theorem 6.4.1 still holds. We can interpret this as saying that the quantile-based algorithm is not conservative enough to achieve a strong volume approximation. Put differently, suppose that on each day  $t$ , the algorithm was told the distribution  $\mathcal{D}_t$  of  $y_t$  before outputting a confidence set for  $y_t$ . In this setting, choosing the

volume optimal set that achieves coverage  $1 - \alpha$  on each day does not necessarily imply that the algorithm will compete in volume with the best set that achieves coverage  $1 - \alpha$  over the sequence of  $y_t$ s. The optimum in hindsight may have lower coverage on days that have spread out distributions and higher coverage on days that have concentrated distributions. This fact that the optimal loss over the sequence could be lower than the sum of the optimal losses on each day poses a roadblock to any strategy that goes through regret minimization with respect to a fixed loss function.

**“Best of both worlds” for Online Conformal Prediction.** We note that the work of [Angelopoulos, Barber, and Bates \[2024\]](#) provides a “best of both worlds” guarantee for online conformal prediction. This does not contradict our result that there is no possible “best of both worlds” guarantee, as the two works are referring to different notions of optimality. [Angelopoulos, Barber, and Bates \[2024\]](#) provide a guarantee that simultaneously achieves average coverage tending to  $1 - \alpha$  on arbitrary sequences, and coverage on each day tending to  $1 - \alpha$  on exchangeable sequences. In this work, we are interested in optimizing efficiency. We consider algorithms that achieve average coverage both for arbitrary and exchangeable sequences, and we would like to show that they achieve the best possible volume approximation ratio in these two settings. We show that this type of guarantee is impossible to achieve.

## 6.2 Algorithmic Guarantees for Arbitrary Sequences

We begin by analyzing the performance of [Algorithm 6.1](#) on arbitrary sequences, for general settings of the allowable error rate  $R(t)$ . In the analysis, we show that, even though it is not explicitly enforced by the algorithm, most times that  $I_{\text{current}}$  is reset ([Algorithm 6.1](#), Line 4), its size increases multiplicatively by a factor that is linear in  $\mu$  (the core of this



argument is illustrated in Figure 6.1). Since the smallest width that  $I_{\text{current}}$  is reset to is  $\text{minwidth}$ , and the largest it can be is 1, the number of times it is resets essentially depends on  $\log_{\mu}(1/\text{minwidth}) = \log(1/\text{minwidth})/\log(\mu)$ .

Call the time window between two subsequent resets of  $I_{\text{current}}$  a “phase.” Over a phase, Algorithm 6.1 plays one fixed interval that is feasible over the whole phase. This allows us to bound the number of mistakes made in a particular phase by the total number of mistakes that any feasible interval is allowed to have made at that time.

**Theorem 6.2.1** (Meta-algorithm Guarantee for Arbitrary Sequences). *Fix a scale lower bound*

*$\text{minwidth} > 0$ , a multiplicative volume approximation  $\mu > 3$ , and an allowable error rate function*

*$R(t) : \{0, \dots, T-1\} \rightarrow [0, 1]$  such that  $R(t) \leq \frac{1}{4}$  for all  $t \geq t^*$ .*

*On a sequence  $S = (y_1, \dots, y_T)$  of length  $T$ , let  $I_t$  be the interval that Algorithm 6.1 plays on each day  $t \in [T]$ . The maximum volume of these intervals is bounded as*

$$\text{vol}(I_t) \leq \mu \max \left\{ \text{OPT}_S \left( \min_{1 \leq t \leq T} \frac{t}{T} R(t) \right), \text{minwidth} \right\} \quad \forall t \in [T].$$

*The number of mistakes of these intervals can be bounded as*

$$\sum_{i=1}^T \mathbf{1}\{y_t \notin I_t\} \leq t^* + O \left( \frac{\log(1/\text{minwidth}) \log T}{\log(\mu)} \left( 1 + \max_{0 \leq t \leq T-1} R(t)t \right) \right).$$

*Moreover, the algorithm is deterministic and therefore robust to an adaptive adversary.*

*Proof.* Any interval that has made  $\leq R(t-1)(t-1)$  mistakes before day  $t \geq 2$  is a feasible

choice for  $I_t$  (Algorithm 6.1, line 7). Therefore, any interval  $I$  that satisfies

$$\frac{1}{T} \sum_{i=1}^T \mathbf{1}\{y_t \notin I\} \leq \min_{1 \leq t \leq T-1} \frac{t}{T} R(t)$$

is a feasible choice for  $I_t$ , on every day  $t$ . In particular,  $\text{OPT}_S \left( \min_{1 \leq t \leq T} \frac{t}{T} R(t) \right)$  satisfies the above by definition, and is therefore always a feasible choice for  $I_t$ .

Lines 7 and 10 ensure that the interval that is played has volume at most  $\mu$  times the maximum of `minwidth` and the volume of any feasible interval. Therefore the volume played by Algorithm 6.1 on each day  $t'$  is bounded by

$$\text{vol}(I_{t'}) \leq \mu \max \left\{ \text{OPT}_S \left( \min_{1 \leq t \leq T} \frac{t}{T} R(t) \right), \text{minwidth} \right\}, \quad \forall t' \in [T].$$

Now we bound the number of mistakes. Let  $t_1, t_2$  be two subsequent times that  $I_{\text{current}}$  is reset. That is, the same  $I_{\text{current}}$  is played on all days  $[t_1, t_2)$ . Call  $[t_1, t_2)$  a *phase*. We begin by bounding the number of mistakes in a phase. Since  $I_{\text{current}}$  is feasible on day  $t_2 - 1$ ,  $I_{\text{current}}$  makes  $\leq R(t_2 - 2)(t_2 - 2)$  mistakes on  $\{y_1, \dots, y_{t_2-1}\}$ . Therefore  $I_{\text{current}}$  makes  $\leq R(t_2 - 2)(t_2 - 2) + 1$  mistakes on this phase:  $\{y_{t_1}, \dots, y_{t_2-1}\}$ . Thus the number of mistakes that the algorithm makes on any phase is

$$\sum_{t=t_1}^{t_2-1} \mathbf{1}\{y_t \in I_{\text{current}}\} \leq 1 + \max_{0 \leq t \leq T-1} R(t)t. \quad (6.2)$$

Now we bound the number of phases. Consider any two days  $t_1, t_2$  such that  $t^* < t_1 < t_2$ , so  $R(t_1 - 1), R(t_2 - 2) \leq \frac{1}{4}$ . Let  $I_1$  be any interval that is feasible at  $t_1$  and  $I_2$  be any interval that is feasible at  $t_2$ .  $I_1$  has captured  $\geq \frac{3}{4}(t_1 - 1)$  points out of  $\{x_1, \dots, x_{t_1-1}\}$ .  $I_2$  has missed  $\leq \frac{1}{4}(t_2 - 1)$  points out of  $\{y_1, \dots, y_{t_1-1}\} \subseteq \{y_1, \dots, y_{t_2-1}\}$ . Therefore as long as  $\frac{t_2-1}{t_1-1} < 3$ ,  $I_1$

and  $I_2$  must capture some point in common and must overlap.

Partition the sequence after day  $t^*$  into  $O(\log T)$  *epochs*, where each epoch  $[t_{\text{start}}, t_{\text{end}}]$  has  $\frac{t_{\text{end}}-1}{t_{\text{start}}-1} < 3$ . We bound the number of mistakes that [Algorithm 6.1](#) makes on each epoch. Fix an epoch  $e$ , and consider the first time in this epoch that [Algorithm 6.1](#) resets  $I_{\text{current}}$ . By line 10,  $I_{\text{current}}$  is set to have volume  $\geq \mu \text{minwidth}$ .

Now consider a subsequent time  $t$  in epoch  $e$  when  $I_{\text{current}}$  is reset. Let  $t_{\text{prev}} < t$  be the previous time in epoch  $e$  when  $I_{\text{current}}$  was reset. Let  $I_{t_{\text{prev}}}$  and  $I_t$  be the minimum volume feasible intervals on day  $t_{\text{prev}}$  and  $t$ , respectively (as chosen in line 7). Let  $I_{\text{current}}^{(t_{\text{prev}})}$  denote the state of  $I_{\text{current}}$  before day  $t$  (after day  $t_{\text{prev}}$ ).

Since  $I_{\text{current}}$  is reset on day  $t$ ,  $I_{\text{current}}^{(t_{\text{prev}})}$  is not feasible on day  $t$ , and in particular  $I_t \not\subseteq I_{\text{current}}^{(t_{\text{prev}})}$ . By the previous argument  $I_{t_{\text{prev}}}$  and  $I_t$  must overlap. Any interval that contains a point in  $I_{t_{\text{prev}}}$  and also contains a point that is not in  $I_{\text{current}}^{(t_{\text{prev}})} = \widehat{\mu} I_{t_{\text{prev}}}$ , must have volume  $\geq \frac{\mu-1}{2} \cdot \text{vol} \left( I_{\text{current}}^{(t_{\text{prev}})} \right)$ . This argument is illustrated in [Figure 6.1](#). Denote the state of  $I_{\text{current}}$  at the end of day  $t$  as  $I_{\text{current}}^{(t)}$ . By line 10, we have that  $I_{\text{current}}^{(t)} \supseteq \mu I_t$ , so

$$\text{vol} \left( I_{\text{current}}^{(t)} \right) \geq \frac{\mu-1}{2} \text{vol} \left( I_{\text{current}}^{(t_{\text{prev}})} \right).$$

Since  $\text{vol}(I_t) \leq 1$  for all days  $t$ , we have that  $\text{vol}(I_{\text{current}}^{(t)}) \leq \mu$  for all days  $t$ . Thus, within an epoch, as long as  $\mu > 3$ , the number of times  $I_{\text{current}}$  can be reset is

$$\leq 1 + \log_{\frac{\mu-1}{2}} \left( \frac{\mu}{\mu \text{minwidth}} \right) = O \left( \frac{\log(1/\text{minwidth})}{\log(\mu)} \right), \quad (6.3)$$

and the total number of times that  $I_{\text{current}}$  is reset after day  $t^*$  is  $O \left( \frac{\log(1/\text{minwidth}) \log T}{\log(\mu)} \right)$ .

Our previous bound on the number of mistakes in a phase ([6.2](#)) allows us to bound the total

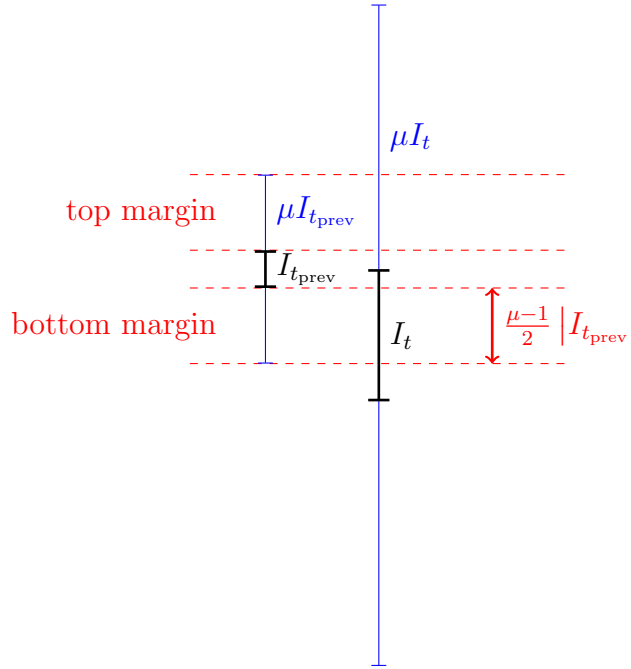


Figure 6.1:  $I_t$  achieved more coverage than  $I_{\text{current}}^{(t_{\text{prev}})} = \mu I_{t_{\text{prev}}}$  on the sequence so far, so  $I_t$  is not contained in  $\mu I_{t_{\text{prev}}}$ . We also know that  $I_{t_{\text{prev}}}$  and  $I_t$  must overlap. Thus,  $I_t$  must fully contain one of the margins that we add to  $I_{t_{\text{prev}}}$  to form  $\mu I_{t_{\text{prev}}}$  (denoted here as the regions between the red dashed lines). This gives a lower bound on the size of  $I_t$  in terms of the size of  $I_{t_{\text{prev}}}$ , and therefore a lower bound on the size of  $I_{\text{current}}^{(t)}$  in terms of the size of  $I_{\text{current}}^{(t_{\text{prev}})}$ .

number of mistakes that [Algorithm 6.1](#) makes as

$$= t^* + O\left(\frac{\log(1/\text{minwidth}) \log T}{\log(\mu)} \left(1 + \max_{0 \leq t \leq T-1} R(t)t\right)\right).$$

□

The previous analysis allows us to choose an  $R(t)$  to optimize the bounds. The following corollary observes that  $R(t) = \frac{\alpha T}{t}$  gives a guarantee in terms of  $\text{OPT}(\alpha)$ . This gives a guarantee that exhibits a tradeoff between  $\mu$ , the multiplicative volume approximation of the intervals played by [Algorithm 6.1](#), and the number of mistakes that [Algorithm 6.1](#) makes. Later, in [Theorem 6.4.1](#) we give a lower bound that shows that the  $\mu$ -mistake tradeoff that

[Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  achieves is optimal. That is, for any setting of  $\mu$ , this algorithm makes a near-optimal number of mistakes subject to achieving multiplicative approximation  $\leq \mu$ .

$R(t) = \frac{\alpha T}{t}$  ensures that the intervals that are feasible on day  $t$  are exactly those that have made  $\leq \alpha T$  mistakes so far, where  $\alpha T$  is the total number of mistakes that  $\text{OPT}(\alpha)$  is allowed to make over the whole sequence. We can think of this algorithm as playing conservatively, and not ruling out any interval until it is demonstrably not  $\text{OPT}(\alpha)$ .

**Corollary 6.2.2** (Algorithm for Arbitrary Sequences). *Fix a scale lower bound  $\text{minwidth} > 0$ , multiplicative volume approximation  $\mu > 3$ , target miscoverage rate  $\alpha \geq 0$ , and time horizon  $T$  (all known to the algorithm). Set  $R(t)$  such that  $R(0) = 1$  and  $R(t) = \alpha \frac{T}{t}$  for  $t > 0$ . With these parameters, on any sequence  $S = (y_1, \dots, y_T)$  of length  $T$ , [Algorithm 6.1](#) plays interval  $I_t$  on each day  $t \in [T]$ , such that the maximum volume of any  $I_t$  is approximately optimal*

$$\text{vol}(I_t) \leq \mu \max \{ \text{OPT}_S(\alpha), \text{minwidth} \} \quad \forall t \in [T],$$

and the number of mistakes is bounded by

$$\sum_{i=1}^T \mathbf{1}\{y_t \notin I_t\} = O \left( \frac{\log(1/\text{minwidth})}{\log(\mu)} (\alpha T + 1) \right).$$

Moreover, the algorithm is deterministic and therefore robust to an adaptive adversary.

*Proof.* The bound on maximum volume follows directly from the volume guarantee in [Theorem 6.2.1](#).

For the bound on number of mistakes, this choice of  $R(t)$  ensures that an interval is feasible on day  $t$  if and only if it has made  $\leq \alpha T$  mistakes up to day  $t$ . Consider  $t_1, t_2$  such that  $2\alpha T + 1 < t_1 < t_2$ . Let  $I_1$  be any feasible interval on day  $t_1$  and  $I_2$  be any feasible interval

on day  $t_2$ . Since  $I_1$  and  $I_2$  each make  $\leq \alpha T$  mistakes over the first  $2\alpha T + 1$  days, they must capture at least one day in common. Therefore  $I_1$  and  $I_2$  overlap. Thus we can think of all days  $t > 2\alpha T + 1$  as forming one epoch, and the proof of [Theorem 6.2.1](#) ([Equation \(6.3\)](#)) tells us that the number of phases in this epoch is

$$= O\left(\frac{\log(1/\text{minwidth})}{\log(\mu)}\right).$$

The proof of [Theorem 6.2.1](#) ([Equation \(6.2\)](#)) also tells us that, for this choice of  $R(t)$ , the number of mistakes in each phase is bounded by  $\leq 1 + \alpha T$ . Thus the total number of mistakes made is

$$\leq (2\alpha T + 1) + O\left(\frac{\log(1/\text{minwidth})}{\log(\mu)}\right)(\alpha T + 1) = O\left(\frac{\log(1/\text{minwidth})}{\log(\mu)}(\alpha T + 1)\right).$$

□

### 6.3 Algorithmic Guarantees for Exchangeable Sequences

In this section we analyze the performance of [Algorithm 6.1](#) on exchangeable sequences. The analysis differs significantly from the arbitrary order setting, because we want to take advantage of concentration. That is, an exchangeable sequence  $\mathbf{S}$  has the property that the coverage of any interval  $I \subseteq [0, 1]$  over one subsequence of  $\mathbf{S}$  should be similar to the coverage of  $I$  over a different subsequence of  $\mathbf{S}$ .

Quantitatively, we can access this property through *uniform convergence*. In the most commonly used form, uniform convergence tells us that for a set family  $\mathcal{F}$  of bounded VC-dimension, for us the family of intervals over  $[0, 1]$ , and a set  $Y$  of  $n$  samples drawn i.i.d. from some distribution  $\mathcal{D}$ , the coverage of every set in  $\mathcal{F}$  over  $Y$  simultaneously converges

to its true coverage over  $\mathcal{D}$ . That is, treating the VC-dimension of  $\mathcal{F}$  and failure probability as constants, the coverage of every set in  $\mathcal{F}$  over  $Y$  is within  $\pm O(1/\sqrt{n})$  of its coverage over  $\mathcal{D}$ .

A similar statement holds even for sets of samples that are exchangeable but not necessarily i.i.d.. Let  $\mathbf{S} = (\mathbf{S}_1, \dots, \mathbf{S}_T)$  be an exchangeable sequence of length  $T$ , and let  $Y$  be some (not necessarily contiguous) subsequence of size  $t \leq T$  of  $\mathbf{S}$ .

Uniform convergence for exchangeable sequences tells us that, treating the VC-dimension of  $\mathcal{F}$  and failure probability as constants, the coverage of every set in  $\mathcal{F}$  over  $Y$  is within  $\pm O(1/\sqrt{t})$  of its coverage over  $\mathbf{S}$ . That is, the coverage of sets in  $\mathcal{F}$  over subsequences of  $\mathbf{S}$  behaves essentially as though the elements of  $Y$  were drawn i.i.d. from  $\sim_u \{\mathbf{S}_1, \dots, \mathbf{S}_T\}$ . A formal version of this statement can be found in [Lemma 2.8.1](#).

In analyzing the algorithm on exchangeable sequences, we must bound the number of mistakes by arguing that on each day  $t$ , the interval  $I_t$  chosen by the algorithm has high coverage. However, this requires us to have a tight bound on the coverage of  $I_t$  over the subsequence of length 1 that only contains day  $t$ , and the previous argument does not give tight bounds for short subsequences. We address this issue by instead bounding the coverage of  $I_t$  on the *suffix* of the input sequence that contains all days between  $t$  and  $T$ . Considering a longer subsequence allows us to get tighter bounds. Then, we observe that the coverage of  $I_t$  on the suffix is a property only of the unordered (multi)set of values in the suffix, and not the order that the values appear. Thus, conditioned on the coverage bound holding, the suffix of the sequence is still exchangeable within itself, so the coverage on day  $t$  is actually equal to the coverage over the whole suffix.

**Theorem 6.3.1** (Meta-algorithm Guarantee for Exchangeable Sequences). *Fix a scale lower bound*

$\text{minwidth} > 0$ , a multiplicative volume approximation  $\mu \geq 1$ , and an allowable error rate function  $R(t) : \{0, \dots, T-1\} \rightarrow [0, 1]$ . Let  $\mathbf{S}$  be an exchangeable input sequence of length  $T$ . Let  $I_t$  be a random variable, which is the interval that [Algorithm 6.1](#) plays on input  $\mathbf{S}$ . Then, the expected maximum volume of the intervals can be bounded as

$$\mathbb{E} \left[ \max_t \text{vol}(I_t) \right] \leq \mu \max \left\{ \text{OPT}_{\mathbf{S}} \left( \min_{0 \leq t < T} \left[ R(t) - O \left( \sqrt{\log T/t} \right) \right] \right), \text{minwidth} \right\} + \frac{1}{T^2},$$

and achieves expected coverage

$$\mathbb{E} \left[ \frac{1}{T} \mathbf{1}_{\{y_t \in I_t\}} \right] \geq 1 - \frac{1}{T} \sum_{t=1}^T R(t-1) - O \left( \sqrt{\log T/T} \right).$$

*Proof.* Fix an exchangeable input sequence  $\mathbf{S} = (Y_1, \dots, Y_T)$ , where the random variable  $Y_i$  is the input on day  $i$ . Let the random variable  $M^{\mathbf{S}}$  be the number of mistakes that [Algorithm 6.1](#) (for  $\text{minwidth}$ ,  $\mu$ ,  $R(t)$ ) makes on  $\mathbf{S}$ . Let  $M_t^{\mathbf{S}}$  be an indicator random variable of whether [Algorithm 6.1](#) makes a mistake on day  $t$ . That is, let  $I_t$  be the interval that [Algorithm 6.1](#) plays on day  $t$ . Then

$$M_t^{\mathbf{S}} = \mathbf{1}[Y_t \notin I_t].$$

We analyze the algorithm's performance on one particular day  $t$ . We will refer to the days before  $t$ , i.e. the window  $[1, t-1]$ , as the *prefix*, and the days  $t$  and after, i.e. the window  $[t, T]$ , as the *suffix*. For an interval  $I \subseteq [0, 1]$  and a window (contiguous subsequence)  $W = [t_1, t_2] \subseteq [1, T]$ , define  $\text{err}_{\mathbf{S}}(I, W)$  to be the fraction of days in  $W$  of  $\mathbf{S}$  that are not covered by  $I$ . Since the set of intervals over  $[0, 1]$  has VC-dimension 2, [Lemma 2.8.1](#) tells us that the following statements hold, each with probability  $\geq 1 - \frac{1}{T^3}$ , for some absolute



constant  $C$ :

$$\forall I \quad \text{err}_{\mathbf{S}}(I, [1, T]) - C\sqrt{\frac{\log T}{t-1}} \leq \text{err}_{\mathbf{S}}(I, [1, t-1]) \leq \text{err}_{\mathbf{S}}(I, [1, T]) + C\sqrt{\frac{\log T}{t-1}}, \quad (6.4)$$

$$\forall I \quad \text{err}_{\mathbf{S}}(I, [1, T-t+1]) - C\sqrt{\frac{\log T}{t-1}} \leq \text{err}_{\mathbf{S}}(I, [t, T]) \leq \text{err}_{\mathbf{S}}(I, [1, T]) + C\sqrt{\frac{\log T}{T-t+1}}. \quad (6.5)$$

We will refer to (6.4) and (6.5) as the *uniform convergence property*. Note that (6.4) and (6.5) are properties only of the set of values that appear in the prefix and suffix, and does not imply anything about the order that elements appear in the suffix. That is, conditioned on (6.4) and (6.5), the suffix is still exchangeable, and we have

$$\forall I, t' \in [t, T] \quad \mathbb{P}_{\mathbf{S}}[Y_{t'} \in I \mid (6.4), (6.5)] = \mathbb{E}_{\mathbf{S}}[\text{err}_{\mathbf{S}}(I, [t, T]) \mid (6.4), (6.5)]. \quad (6.6)$$

This allows us to bound the expected value of  $M_t^{\mathbf{S}}$  conditioned on (6.4) and (6.5). We have deterministically that  $\text{err}_{\mathbf{S}}(I_t, [1, t-1]) \leq R(t-1)$  (Algorithm 6.1, Line 7).

$$\begin{aligned} & \mathbb{E}_{\mathbf{S}}[M_t^{\mathbf{S}} \mid (6.4), (6.5)] \\ &= \mathbb{P}_{\mathbf{S}}[Y_t \in I_t \mid (6.4), (6.5)] \\ &= \mathbb{E}_{\mathbf{S}}[\text{err}_{\mathbf{S}}(I, [t, T]) \mid (6.4), (6.5)] && \text{by (6.6)} \\ &\leq \mathbb{E}_{\mathbf{S}}\left[\text{err}_{\mathbf{S}}(I, [1, T]) + C\sqrt{\frac{\log T}{T-t+1}} \mid (6.4), (6.5)\right] && \text{by (6.5)} \\ &\leq \mathbb{E}_{\mathbf{S}}\left[\text{err}_{\mathbf{S}}(I, [1, t-1]) + C\sqrt{\frac{\log T}{T-t+1}} + C\sqrt{\frac{\log T}{t-1}} \mid (6.4), (6.5)\right] && \text{by (6.4)} \\ &\leq R(t-1) + C\sqrt{\frac{\log T}{T-t+1}} + C\sqrt{\frac{\log T}{t-1}}. \end{aligned}$$

We also have that  $M_t^{\mathbf{S}} \leq 1$ . So in total this gives us that

$$\mathbb{E}_{\mathbf{S}} [M_t^{\mathbf{S}}] \leq R(t-1) + C\sqrt{\frac{\log T}{T-t+1}} + C\sqrt{\frac{\log T}{t-1}} + \frac{2}{T^3}. \quad (6.7)$$

This allows us to bound  $M^{\mathbf{S}}$ .

$$\begin{aligned} \mathbb{E}_{\mathbf{S}} [M^{\mathbf{S}}] &\leq \sum_{t=1}^T [M_t^{\mathbf{S}}] \\ &\leq \sum_{t=1}^T \left[ R(t-1) + C\sqrt{\frac{\log T}{T-t+1}} + C\sqrt{\frac{\log T}{t-1}} + \frac{2}{T^3} \right] \\ &\leq O\left(\sqrt{T \log T}\right) + \sum_{t=1}^T R(t-1). \end{aligned}$$

Thus the expected coverage of [Algorithm 6.1](#) can be bounded as

$$1 - \frac{1}{T} \mathbb{E}_{\mathbf{S}} [M^{\mathbf{S}}] = 1 - \frac{1}{T} \sum_{t=1}^T R(t-1) - O\left(\sqrt{\log T/T}\right).$$

Now we bound the volume of intervals played by [Algorithm 6.1](#). We condition on (6.4) holding simultaneously for all  $t \in [1, T]$ . Since this holds with probability  $\geq 1 - \frac{1}{T^3}$  for each  $t$ , it holds for all  $t$  with probability  $\geq 1 - \frac{1}{T^2}$ .

Consider any interval  $I$  that has coverage  $\geq 1 - \min_{0 \leq t \leq T} \left[ R(t) + C\sqrt{\log T/t} \right]$  over  $\mathbf{S}$  in hindsight. Then, conditioned on (6.4) holding for all  $t$ ,  $I$  must be a feasible choice for  $I_t$  on every day  $t$  ([Algorithm 6.1](#), Line 7). Thus, conditioned on (6.4) holding for all  $t$ , [Algorithm 6.1](#) plays intervals of maximum volume

$$\leq \mu \max \left\{ \text{OPT}_{\mathbf{S}} \left( \min_{0 \leq t < T} \left[ R(t) - O\left(\sqrt{\log T/t}\right) \right] \right), \text{minwidth} \right\}.$$

Even if (6.4) does not hold for all  $t$ , [Algorithm 6.1](#) plays intervals of maximum length 1. So

[Algorithm 6.1](#) plays intervals of expected maximum volume

$$\leq \mu \max \left\{ \text{OPT}_{\mathbf{S}} \left( \min_{0 \leq t < T} \left[ R(t) - O \left( \sqrt{\log T/t} \right) \right] \right), \text{minwidth} \right\} + \frac{1}{T^2}.$$

□

From the previous guarantee, it is simple to derive a setting of  $R(t)$  that achieves volume essentially that of  $\text{OPT}(\alpha)$ , and coverage that is within  $O(\sqrt{\log T/T})$  of the target  $1 - \alpha$ . The  $O(\sqrt{\log T/T})$  term is a standard sampling error that should arise since exchangeable sequences are only more general than i.i.d. sequences.

**Corollary 6.3.2** (Algorithm for Exchangeable Sequences). *Fix a scale lower bound  $\text{minwidth} > 0$ , miscoverage rate  $\alpha$ , and time horizon  $T$  (all known to the algorithm). Set  $\mu = 1$ , and  $R(t) = \alpha + O(\sqrt{\log T/t})$ . Let  $\mathbf{S}$  be an exchangeable input sequence of length  $T$ . Let  $I_t$  (a random variable) be the interval played by [Algorithm 6.1](#), with the above parameters, on input  $\mathbf{S}$ . The expected maximum volume of the  $I_t$  can be bounded as*

$$\mathbb{E} \left[ \max_t \text{vol}(I_t) \right] \leq \max\{\text{OPT}_{\mathbf{S}}(\alpha), \text{minwidth}\} + \frac{1}{T^2},$$

and the algorithm achieves expected coverage

$$\mathbb{E} \left[ \frac{1}{T} \sum_{i=1}^T \mathbf{1}\{y_i \in I_i\} \right] \geq (1 - \alpha) - O \left( \sqrt{\log T/T} \right).$$

*Proof.* Follows directly from [Theorem 6.3.1](#). □

We observe that this near-optimal algorithm for online conformal prediction on exchangeable sequences implies a near-optimal algorithm for standard conformal prediction with exchangeable data (i.e., equivalent to [Theorem 3.2.6](#) in [Chapter 3](#), for the case of  $k = 1$  interval). We

view this simple observation as evidence that online conformal prediction on exchangeable sequences can be seen as a generalization of standard conformal prediction with exchangeable data, in a similar way to how online learning generalizes PAC learning.

**Corollary 6.3.3** (Efficiency Guarantee for Standard Conformal Prediction). *Fix a scale lower bound  $\text{minwidth} > 0$ , and miscoverage rate  $\alpha$ . Let  $\mathbf{S} = (Y_1, \dots, Y_T)$  be exchangeable random variables valued in  $[0, 1]$ .*

*Given  $Y_1, \dots, Y_{T-1}$ , we construct the following conformal predictor for  $Y_T$ . Suppose we run [Algorithm 6.1](#) with parameters  $\text{minwidth}, \mu = 1, \alpha, T, R(t) = \alpha + O(\sqrt{\log T/t})$  for  $\lceil t/2 \rceil$  days using  $Y_1, \dots, Y_{\lceil t/2 \rceil}$  as inputs. Let  $I_{\lceil t/2 \rceil}$  be the interval that [Algorithm 6.1](#) predicts on day  $\lceil t/2 \rceil$ .*

*Then,  $I_{\lceil t/2 \rceil}$  is a conformal predictor for  $Y_T$ , achieving coverage*

$$\mathbb{P}_{\mathbf{S}} [Y_T \in I_{\lceil t/2 \rceil}] \geq (1 - \alpha) - O\left(\sqrt{\frac{\log T}{T}}\right),$$

*and  $I_{\lceil t/2 \rceil}$  has volume bounded by*

$$\text{vol}(I_{\lceil t/2 \rceil}) \leq \max\{\text{OPT}_{\mathbf{S}}(\alpha), \text{minwidth}\} + \frac{1}{T^2}.$$

*Proof.* This follows from the proof of [Theorem 6.3.1](#). Let  $I_{\lceil t/2 \rceil}$  be the interval that [Algorithm 6.1](#) plays on day  $\lceil t/2 \rceil$  of input  $\mathbf{S}$ . Let  $M_{\text{final}}^{\mathbf{S}}$  be an indicator random variable of whether  $Y_T \notin I_{\lceil t/2 \rceil}$ . By [\(6.6\)](#) we have that

$$\mathbb{E}_{\mathbf{S}} [M_{\text{final}}^{\mathbf{S}} \mid \text{(6.4) and (6.5)}] = \mathbb{P}_{\mathbf{S}} [Y_T \in I_{\lceil t/2 \rceil} \mid \text{(6.4) and (6.5)}] = \mathbb{P}_{\mathbf{S}} [Y_{\lceil t/2 \rceil} \in I_{\lceil t/2 \rceil} \mid \text{(6.4) and (6.5)}].$$

Thus (6.7) ensures that

$$\begin{aligned} \mathbb{P}_{\mathbf{S}} [Y_T \notin I_{\lceil t/2 \rceil}] &= \mathbb{E}_{\mathbf{S}} [M_{\text{final}}^{\mathbf{S}}] \\ &\leq R(\lceil t/2 \rceil - 1) + C \sqrt{\frac{\log T}{T - \lceil t/2 \rceil + 1}} + C \sqrt{\frac{\log T}{\lceil t/2 \rceil - 1}} + \frac{2}{T^3} \\ &\leq \alpha + O\left(\sqrt{\frac{\log T}{T}}\right), \end{aligned}$$

and  $I_{\lceil t/2 \rceil}$  achieves coverage

$$\geq (1 - \alpha) - O\left(\sqrt{\frac{\log T}{T}}\right)$$

on  $Y_T$ .

For the efficiency bound, Corollary 6.3.2 ensures that the expected maximum volume of any interval played by Algorithm 6.1, and therefore the expected volume of  $I_{\lceil t/2 \rceil}$  is bounded by

$$\leq \max\{\text{OPT}_{\mathbf{S}}(\alpha), \text{minwidth}\} + \frac{1}{T^2}.$$

□

Finally, we note that for the optimal algorithm for exchangeable sequences we use Algorithm 6.1 with  $R(t) = \alpha - O(\sqrt{\log T/T})$  (Corollary 6.3.2). However, for the optimal algorithm for arbitrary sequences we use Algorithm 6.1 with  $R(t) = \alpha T/t$  (Corollary 6.2.2). This raises the natural question of whether we can design a single algorithm (perhaps but not necessarily by using Algorithm 6.1 with a unified choice of  $R(t)$ ) that simultaneously achieves the optimal guarantee on both exchangeable and arbitrary sequences.

Later in Theorem 6.5.1, we give a lower bound that shows that it is indeed not possible to design any algorithm that is simultaneously optimal in both of these settings. Theorem 6.5.1

quantifies this bound by showing that for a given choice of  $\mu$ , `minwidth`, and  $\alpha$ , any algorithm that has volume bounded by  $\mu \max\{\text{OPT}_S(\alpha), \text{minwidth}\}$  on every *arbitrary* input sequence  $S$ , must make

$$\tilde{\Omega} \left( \min \left\{ \ln(1/\alpha), \frac{\ln(1/\text{minwidth})}{\ln \mu} \right\} \alpha T \right)$$

mistakes in expectation on some *i.i.d.* sequence  $S'$ . Here  $\tilde{\Omega}$  hides subpolynomial factors in  $\alpha$ .

Algorithmically, we show that [Algorithm 6.1](#) with the conservative choice of  $R(t) = \alpha T/t$  is able to meet this mistake bound on exchangeable data, up to a sampling error term. We note that this algorithm has the same multiplicative volume approximation  $\mu$  on both arbitrary and exchangeable data, while our lower bound does not rule out the possibility of an algorithm having a better approximation ratio on exchangeable sequences than on arbitrary sequences.

To match the lower bound, we must do a finer analysis of [Algorithm 6.1](#) on exchangeable sequences that accounts for the number of phases of the algorithm. In the analysis of [Algorithm 6.1](#) on arbitrary sequences ([Theorem 6.2.1](#)), we used the fact that the algorithm plays one fixed interval for a whole phase. This allowed us to amortize the error of the interval that the algorithm plays over the whole phase, and get a mistake bound that depends on the number of phases which is  $O(\log(1/\text{minwidth})/\log \mu)$ . We observe that when the number of phases is small, a similar amortizing argument can get a tighter bound on the number of mistakes that [Algorithm 6.1](#) makes, even in the exchangeable case.

**Theorem 6.3.4** (Simultaneous Guarantee for Arbitrary and Exchangeable Sequences). *Fix a scale lower bound  $\text{minwidth} > 0$ , target miscoverage rate  $\alpha$ , multiplicative approximation  $\mu$ , and time horizon  $T$  (all known to the algorithm). Set  $R(t) = \frac{\alpha T}{t}$ . Let  $\mathbf{S}$  be an exchangeable*

input sequence of length  $T$ . On input  $\mathbf{S}$ , [Algorithm 6.1](#) with the above parameters makes

$$O\left(\min\left\{\sqrt{T\log T}\sqrt{\frac{\log(1/\text{minwidth})}{\log\mu}} + \frac{\log(1/\text{minwidth})}{\log\mu}\alpha T, \quad \sqrt{T\log T} + \log(1/\alpha)\alpha T\right\}\right)$$

mistakes in expectation, and plays intervals of maximum volume

$$\leq \mu \max\{\text{OPT}_{\mathbf{S}}(\alpha), \text{minwidth}\}.$$

*Proof.* We condition on uniform convergence for every window (contiguous subsequence) of  $\mathbf{S}$ . For a window  $W = [t_1, t_2] \subseteq [1, T]$ , and an interval  $I \subseteq [0, 1]$ , define  $\text{err}_{\mathbf{S}}(I, W)$  to be the fraction of days in  $W$  of  $\mathbf{S}$  that are not covered by  $I$ . For each window  $W$ , we have that with probability  $\geq 1 - \frac{1}{T^4}$ , the coverage of every interval  $I$  over  $W$  is within

$$\varepsilon_W = C\sqrt{\frac{\log T}{|W|}}$$

its coverage over all of  $\mathbf{S}$ , for some universal constant  $C$ . Taking a union bound over all  $O(T^2)$  possible windows, we have that

$$\forall W, I \quad \text{err}_S(I, [1, T]) - \varepsilon_W \leq \text{err}_S(I, W) \leq \text{err}_S(I, [1, T]) + \varepsilon_W. \quad (6.8)$$

with probability  $\geq 1 - \frac{1}{T^2}$ . We refer to the property (6.8) as the *uniform convergence property*.

Now let  $\hat{S}$  be any realization of  $\mathbf{S}$  that satisfies the uniform convergence property. Let  $K$  be the number of times sets  $I_{\text{current}}$  [Algorithm 6.1](#) (for  $\mu$ ,  $\text{minwidth}$ ,  $R(t) = \frac{\alpha T}{t}$ ) on  $\hat{S}$ . Let  $t_i$  for  $1 \leq i \leq K$  be the days when  $I_{\text{current}}$  is set, where  $t_1 = 1$ , and we define  $t_{K+1} = T + 1$  for convenience. We refer to the window  $[t_i, t_{i-1})$  as *phase  $i$* .

Let  $M^{\hat{S}}$  be the number of mistakes that [Algorithm 6.1](#) makes on  $\hat{S}$ . Let  $M_t^{\hat{S}}$  be the miscov-

erage rate of [Algorithm 6.1](#) over phase  $i$  of  $\widehat{S}$ . Let  $I_i$  be the value of  $I_{\text{current}}$  on phase  $i$ . If [Algorithm 6.1](#) resets  $I_{\text{current}}$  on day  $t_{i+1}$ , it means that  $I_i$  has error rate  $\leq \frac{\alpha T}{t_{i+1}-1}$  over the window  $[1, t_{i+1} - 1]$  ([Algorithm 6.1](#), Line 4). By the uniform convergence property (6.8),

$$\begin{aligned}
M_i^{\widehat{S}} &= \text{err}_{\widehat{S}}(I_i, [t_i, t_{i+1} - 1]) \\
&\leq \text{err}_{\widehat{S}}(I_i, [1, T]) + \varepsilon_{[t_i, t_{i+1}-1]} \\
&\leq \text{err}_{\widehat{S}}(I_i, [1, t_{i+1} - 1]) + \varepsilon_{[1, t_{i+1}-1]} + \varepsilon_{[t_i, t_{i+1}-1]} \\
&\leq \frac{\alpha T}{t_{i+1} - 1} + C\sqrt{\frac{\log T}{t_{i+1} - 1}} + C\sqrt{\frac{\log T}{t_{i+1} - t_i}} \\
&\leq \frac{\alpha T}{t_{i+1} - 1} + 2C\sqrt{\frac{\log T}{t_{i+1} - t_i}}.
\end{aligned}$$

We bound  $M^{\widehat{S}}$  in terms of the  $M_i^{\widehat{S}}$ s. Since  $R(t) = \frac{\alpha T}{t} \geq 1$  for  $t \leq \alpha T$ , and the initial interval has coverage 0, we have that  $t_2 = \alpha T + 1$  and  $M_1 = 1$ .

$$\begin{aligned}
M^{\widehat{S}} &= \sum_{i=1}^K (t_{i+1} - t_i) M_i^{\widehat{S}} \\
&= \alpha T + \sum_{i=2}^K (t_{i+1} - t_i) M_i^{\widehat{S}} \\
&\leq \alpha T + \sum_{i=2}^K (t_{i+1} - t_i) \left( \frac{\alpha T}{t_{i+1} - 1} + 2C\sqrt{\frac{\log T}{t_{i+1} - t_i}} \right) \\
&\leq \alpha T + \sum_{i=2}^K \left[ (t_{i+1} - t_i) \frac{\alpha T}{t_{i+1} - 1} + 2C\sqrt{\log T} \sqrt{(t_{i+1} - t_i)} \right] \\
&\leq \alpha T + 2C\sqrt{\log T} \sqrt{K-1} \sqrt{T} + \alpha T \sum_{i=2}^K \left[ \frac{t_{i+1} - 1}{t_{i+1} - 1} - \frac{t_i - 1}{t_{i+1} - 1} \right] \quad \text{by concavity of } \sqrt{\cdot} \\
&\leq \alpha T + 2C\sqrt{(K-1)T \log T} + (K-1)\alpha T - \alpha T \sum_{i=2}^K \frac{t_i - 1}{t_{i+1} - 1}. \tag{6.9}
\end{aligned}$$

This upper bound on mistakes is maximized when  $\sum_{i=2}^K \frac{t_i - 1}{t_{i+1} - 1}$  is minimized. Recall that



$t_2 = \alpha T + 1$  and  $t_{K+1} = T + 1$ . Assume without loss of generality that  $\alpha T$  is an integer. So we have

$$\prod_{i=2}^K \frac{t_i - 1}{t_{i+1} - 1} = \frac{t_2 - 1}{t_{K+1} - 1} = \alpha$$

$$\sum_{i=2}^K \ln \left( \frac{t_i - 1}{t_{i+1} - 1} \right) = \ln \alpha.$$

For convenience of arithmetic, define  $q_i = (t_i - 1)/(t_{i+1} - 1)$  where every  $q_i > 0$ , so we have  $\sum_{i=2}^K \ln q_i = \ln \alpha$ . By concavity of  $\ln$  we have

$$\frac{1}{K-1} \sum_{i=2}^K \ln q_i \leq \ln \left( \frac{1}{K-1} \sum_{i=2}^K q_i \right)$$

$$\frac{\ln \alpha}{K-1} + \ln(K-1) \leq \ln \left( \sum_{i=2}^K q_i \right)$$

$$(K-1)\alpha^{1/(K-1)} \leq \sum_{i=2}^K q_i = \sum_{i=2}^K \frac{t_i - 1}{t_{i+1} - 1}.$$

Substituting back into (6.9), we have

$$M^{\hat{S}} \leq \alpha T + \sqrt{(K-1)T \log T} + (K-1)\alpha T - (K-1)\alpha^{1/(K-1)}\alpha T$$

$$\leq \alpha T + \sqrt{(K-1)T \log T} + (K-1) \left(1 - \alpha^{1/(K-1)}\right) \alpha T.$$

Since  $\alpha \geq 0$ , we have that  $(K-1) \left(1 - \alpha^{1/(K-1)}\right) \leq K-1$ . We can also bound this term by

$\ln(1/\alpha)$ :

$$\begin{aligned} \frac{\ln \alpha}{K-1} &= -\frac{\ln(1/\alpha)}{K-1} \\ \alpha^{1/(K-1)} &= e^{-\ln(1/\alpha)/(K-1)} \geq 1 - \frac{\ln(1/\alpha)}{K-1} \\ \frac{\ln(1/\alpha)}{K-1} &\geq 1 - \alpha^{1/(K-1)} \\ \ln(1/\alpha) &\geq (K-1) (1 - \alpha^{1/(K-1)}) . \end{aligned}$$

Putting this together we have

$$M^{\hat{S}} \leq \sqrt{(K-1)T \log T} + \left(1 + \min \{K-1, \ln(1/\alpha)\}\right) \alpha T.$$

We can upper bound the number of phases of [Algorithm 6.1](#) on any input as

$$K-1 \leq \log_{\frac{\mu-1}{2}}(1/\text{minwidth}) = O\left(\frac{\log(1/\text{minwidth})}{\log \mu}\right).$$

So for every realization  $\hat{S}$  of  $\mathbf{S}$  that satisfies the uniform convergence property (6.8), we have

$$M^{\hat{S}} = O\left(\sqrt{T \log T} \sqrt{\frac{\log(1/\text{minwidth})}{\log \mu}} + \min \left\{ \frac{\log(1/\text{minwidth})}{\log \mu}, \log(1/\alpha) \right\} \alpha T\right).$$

We have that  $\mathbf{S}$  satisfies the uniform convergence property with probability  $\geq 1 - \frac{1}{T^2}$ . If  $\mathbf{S}$  does not satisfy the uniform convergence property, the number of mistakes is trivially upper bounded by  $T$ . So we have that the expected number of mistakes by [Algorithm 6.1](#) (for  $\mu$ ,  $\text{minwidth}$ ,  $R(t) = \frac{\alpha T}{t}$ ) is

$$O\left(\sqrt{T \log T} \sqrt{\frac{\log(1/\text{minwidth})}{\log \mu}} + \min \left\{ \frac{\log(1/\text{minwidth})}{\log \mu}, \log(1/\alpha) \right\} \alpha T\right). \quad (6.10)$$

When the number of phases  $\frac{\log(1/\text{minwidth})}{\log \mu}$  is large, the first “sampling error” term will dominate. In this setting, the original bound in [Theorem 6.3.1](#) which does not do a phase-by-phase analysis gives a tighter bound of

$$\begin{aligned}
 M^{\mathbf{S}} &\leq T \left( \frac{1}{T} \sum_{t=1}^T R(t-1) + O\left(\sqrt{\log T/T}\right) \right) \\
 &\leq \sum_{t=1}^T \max \left\{ 1, \frac{\alpha T}{t} \right\} + O\left(\sqrt{T \log T}\right) \\
 &\leq \alpha T + \alpha T \sum_{t=\alpha T+1}^T \frac{1}{t} + O\left(\sqrt{T \log T}\right) \\
 &\leq O\left(\sqrt{T \log T} + \log(1/\alpha)\alpha T\right). \tag{6.11}
 \end{aligned}$$

(6.10) and (6.11) together bound the expected number of mistakes by

$$O \left( \min \left\{ \sqrt{T \log T} \sqrt{\frac{\log(1/\text{minwidth})}{\log \mu}} + \frac{\log(1/\text{minwidth})}{\log \mu} \alpha T, \sqrt{T \log T} + \log(1/\alpha)\alpha T \right\} \right).$$

The volume guarantee follows directly from [Corollary 6.2.2](#). □

## 6.4 Lower Bound for Arbitrary Order Sequences

We provide a lower bound that shows that our algorithm for arbitrary order sequences is optimal. That is, for any given choice of miscoverage rate  $\alpha$ , scale lower bound `minwidth`, and multiplicative volume approximation  $\mu$ , we prove that any algorithm that achieves volume  $\leq \mu \min\{\text{OPT}_S(\alpha), \text{minwidth}\}$  on every arbitrary order sequence  $S$  of length  $T$  must make a multiplicative factor more mistakes than the target  $\alpha T$ . This mistake bound shows that our upper bound of running [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  is near-optimal in all parameters. In particular, we note that [Algorithm 6.1](#) actually plays intervals with maximum volume

bounded by  $\mu \min\{\text{OPT}_S(\alpha), \text{minwidth}\}$ , which is even stronger than playing intervals with average volume bounded by  $\mu \min\{\text{OPT}_S(\alpha), \text{minwidth}\}$ . Among algorithms that have this property, we show that the expected number of mistakes that [Algorithm 6.1](#) makes:

$$O\left(\min\left\{\frac{\log(1/\text{minwidth})}{\log(\mu)} \cdot \alpha T, T\right\}\right),$$

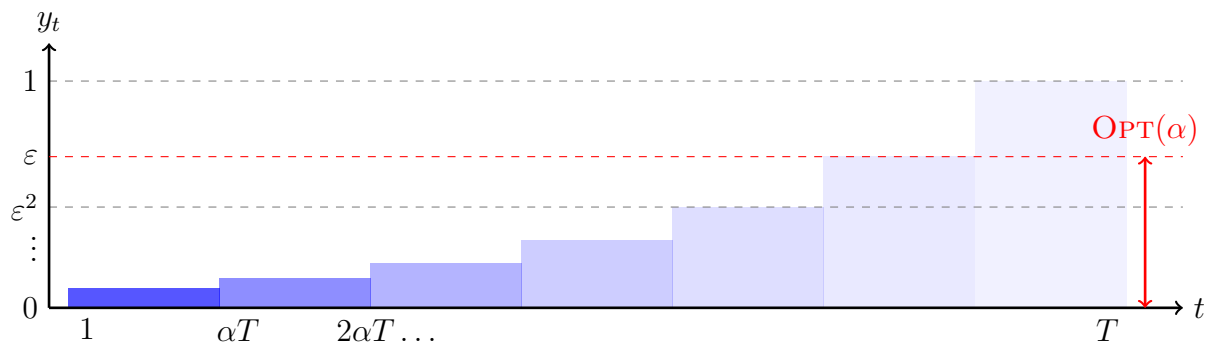
is optimal (see [Corollary 6.2.2](#) for the upper bound).

For algorithms that play intervals that have average volume bounded by  $\mu \min\{\text{OPT}_S(\alpha), \text{minwidth}\}$ , but do not necessarily have maximum volume bounded by this, we show that they must make

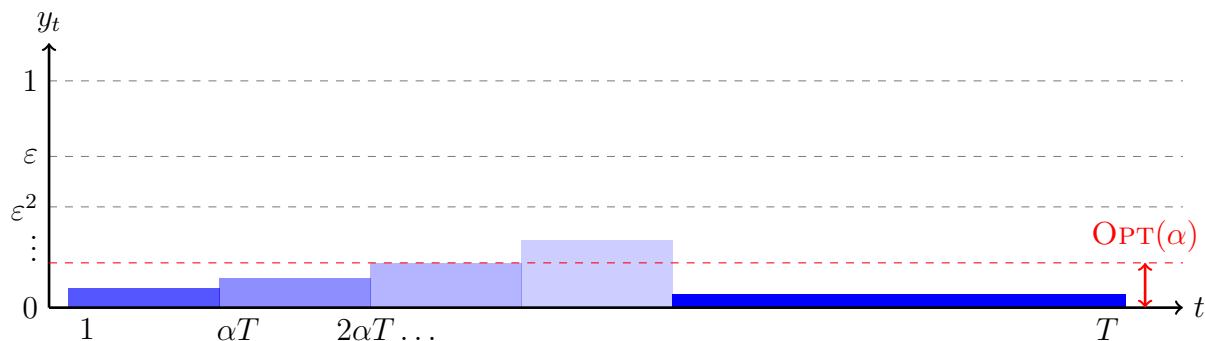
$$\Omega\left(\left\{\frac{\log(1/\text{minwidth})}{\log(\mu) + \log(1/\alpha)} \cdot \alpha T, T\right\}\right)$$

mistakes in expectation. This differs from the earlier bound by a subpolynomial factor in  $\alpha$ , and thus [Algorithm 6.1](#) is near-optimal even among algorithms that meet this weaker condition.

The lower bound construction is directly inspired by a worst-case instance for [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$ . In the analysis ([Theorem 6.2.1](#)), we bound the number of times that [Algorithm 6.1](#) resets  $I_{\text{current}}$ . We then say between two subsequent resets of  $I_{\text{current}}$ , [Algorithm 6.1](#) plays a fixed interval that has accrued less than the target error. So that interval must make at most  $\alpha T$  mistakes over the time that it is feasible, and therefore at most  $\alpha T$  mistakes while it is being played. In the lower bound, we present the algorithm with a series of phases of increasing “scales,” each lasting  $\alpha T$  days. See [Figure 6.2](#) for an illustration. In each phase, the algorithm does not yet know if this phase contains points that are captured by  $\text{OPT}$ , or this is an “outlier” phase. If the algorithm chooses to capture a large fraction of the points,



(a) If an algorithm  $\mathcal{A}$  has low expected coverage ( $\leq \frac{1}{2}$ ) on every phase of this sequence, it must make  $\Omega(T)$  mistakes over the whole sequence.



(b) If an algorithm  $\mathcal{A}$  has high expected coverage ( $\geq \frac{1}{2}$ ) on some phase  $i$  of sequence (a), say phase 4, then it also has high expected coverage on phase 4 of this alternate sequence, as  $\mathcal{A}$  does not yet know which case it is in. Therefore,  $\mathcal{A}$  must incur a high multiplicative volume approximation on this alternate sequence, as the scale of phase 4 is multiplicatively larger than the scales of the other phases.

Figure 6.2: We illustrate the lower bound construction in [Theorem 6.4.1](#). The horizontal axis is the day  $t$  of the sequence and the vertical axis is the value of  $y_t$ .  $y_t$  is drawn according to a uniform distribution, and we illustrate this as a box over the support of  $y_t$  with depth of color corresponding to the density of the distribution. We divide the input sequence into “phases” of length  $\alpha T$ . We then construct an input sequence  $S$  that has the scale of  $y_t$  increasing multiplicatively from one phase to the next ([Figure 6.2a](#)). We also consider a set of alternate sequences  $S_i$ , where  $S_i$  is the same as  $S$  through phase  $i$ , and then drops to a very small scale for the rest of the sequence ([Figure 6.2b](#)). On phase  $i$ , the algorithm does not know if it is in case (a), where the scale will continue to grow, or case (b), where the sequence will drop off.

it will accrue high volume, which leads to high multiplicative ratio in the case that this was an outlier phase. If the algorithm chooses not to capture a large fraction of points, it will accrue a large number of mistakes.

**Theorem 6.4.1** (Lower Bound for Arbitrary Sequences). *For any scale lower bound  $\text{minwidth} > 0$ , miscoverage rate  $\alpha > 0$ , and time horizon  $T$ , such that for any, potentially randomized, algorithm  $\mathcal{A}$  that outputs confidence sets over  $[0, 1]$  that are not necessarily intervals we have:*

1. *If  $\mathcal{A}$  plays sets with expected average volume  $\leq \mu_{\text{avg}} \max\{\text{OPT}_{S_i}(\alpha), \text{minwidth}\}$  on every sequence  $S$ , for some value  $\mu_{\text{avg}} > 0$ , then  $\mathcal{A}$  must make*

$$\Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\text{avg}}) + \log(1/\alpha)} \cdot \alpha T, T \right\} \right)$$

*mistakes in expectation on some sequence  $S'$ . This is  $\Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\text{avg}})} \alpha^{1+\varepsilon'} T, T \right\} \right)$  for any  $\varepsilon' > 0$ .*

2. *If  $\mathcal{A}$  plays sets with expected maximum volume  $\leq \mu_{\text{max}} \max\{\text{OPT}_{S_i}(\alpha), \text{minwidth}\}$  on every sequence  $S$ , for some value  $\mu_{\text{max}} > 0$ , then  $\mathcal{A}$  must make*

$$\Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\text{max}})} \cdot \alpha T, T \right\} \right)$$

*mistakes in expectation on some sequence  $S'$ .*

Furthermore, this lower bound holds against algorithms that only have to be competitive on sequences where the input on each day is drawn from a symmetric unimodal distribution with median  $1/2$ .

*Proof.* Assume without loss of generality that  $1/\alpha$  is an integer, and  $T$  is a multiple of  $1/\alpha$ . Divide the input sequence into  $1/\alpha$  “phases” of length  $\alpha T$ . Let  $K \in \{1, \dots, 1/\alpha\}$  and  $\varepsilon > 0$  be parameters to be chosen later. For each  $1 \leq i \leq K$  we generate a sequence  $S_i^{(K)}$  as follows:

- For phases  $j \leq i$ , all days in phase  $j$  are drawn i.i.d. from  $\sim_u [0, \varepsilon^{K-j}]$ .
- For phases  $j > i$ , all days in phase  $j$  are drawn i.i.d. from  $\sim_u [0, \varepsilon^K]$ .

Fix any potentially randomized algorithm  $\mathcal{A}$ . Consider what  $\mathcal{A}$  does on input sequence  $S_K^{(K)}$ . We say that  $\mathcal{A}$  “hits” phase  $j$  if it captures  $\geq \frac{\alpha T}{2}$  points in phase  $j$ , in expectation. If  $\mathcal{A}$  captures  $< \frac{\alpha T}{2}$  points in phase  $j$  in expectation, we say it “misses.” Either:

- $\mathcal{A}$  misses every phase  $1 \leq j \leq K$ . Then  $\mathcal{A}$  makes  $\geq K \frac{\alpha T}{2}$  mistakes in expectation.
- $\mathcal{A}$  hits some phase  $1 \leq j \leq K$ . Since phase  $j$  is drawn i.i.d.  $\sim_u [0, \varepsilon^{K-j}]$ , the expected average volume that  $\mathcal{A}$  plays on phase  $j$  must be  $\geq \frac{\varepsilon^{K-j}}{2}$ .

Now consider what  $\mathcal{A}$  would do on sequence  $S_j^{(K)}$ . Since  $S_j^{(K)}$  and  $S_K^{(K)}$  are identical through phase  $j$ , this means that  $\mathcal{A}$  must have expected average volume  $\geq \frac{\varepsilon^{K-j}}{2}$  on phase  $j$  of  $S_j^{(K)}$  as well. Thus, on  $S_j^{(K)}$ ,  $\mathcal{A}$  must play average volume  $\geq \frac{\alpha \varepsilon^{K-j}}{2}$  and maximum volume  $\geq \frac{\varepsilon^{K-j}}{2}$  in expectation.

For  $S_j^{(K)}$ , we know that the interval  $[0, \varepsilon^{K-j+1}]$  achieves coverage 1 on every phase other than phase  $j$ . Thus,  $\text{OPT}_{S_j^{(K)}}(\alpha) \leq \varepsilon^{K-j+1}$ . This means that on  $S_j^{(K)}$ ,  $\mathcal{A}$  plays intervals of average volume

$$\geq \frac{\alpha}{2\varepsilon} \text{OPT}_{S_j^{(K)}}(\alpha)$$

in expectation, and intervals of maximum volume

$$\geq \frac{1}{2\varepsilon} \text{OPT}_{S_j^{(K)}}(\alpha)$$

in expectation.

We will set  $\varepsilon, K$  such that  $\text{minwidth} = \frac{1}{2}\varepsilon^K$ . If  $\mathcal{A}$  plays intervals with expected average

volume  $\leq \mu_{\text{avg}} \max\{\text{OPT}_{S_j^{(K)}}(\alpha), \text{minwidth}\}$  on  $S_j^{(K)}$  we have that, either  $\mathcal{A}$  makes  $\geq K \frac{\alpha T}{2}$  mistakes in expectation, or

$$\begin{aligned} \frac{\alpha}{2\varepsilon} \text{OPT}_{S_j^{(K)}}(\alpha) &\leq \mu_{\text{avg}} \max\{\text{OPT}_{S_j^{(K)}}(\alpha), \text{minwidth}\} \\ \min\left\{\frac{\alpha}{2\varepsilon}, \frac{\alpha}{2\varepsilon^{K+2}} \text{OPT}_{S_j^{(K)}}(\alpha)\right\} &\leq \mu_{\text{avg}} \\ \frac{\alpha}{2(\text{minwidth}/2)^{1/K}} &= \frac{\alpha}{2\varepsilon} \leq \mu_{\text{avg}}, \end{aligned} \tag{6.12}$$

where the last line follows because  $\text{OPT}_{S_j^{(K)}}(\alpha) \geq \varepsilon^{K+1}$ . Similarly, if  $\mathcal{A}$  plays intervals with expected maximum volume  $\leq \mu_{\text{max}} \max\{\text{OPT}_{S_j^{(K)}}(\alpha), \text{minwidth}\}$  on  $S_j^{(K)}$  we have that, either  $\mathcal{A}$  makes  $\geq K \frac{\alpha T}{2}$  mistakes in expectation, or

$$\frac{1}{2(\text{minwidth}/2)^{1/K}} = \frac{1}{2\varepsilon} \leq \mu_{\text{max}}. \tag{6.13}$$

Setting  $K = \min\left\{\left\lfloor \frac{\log(2/\text{minwidth})}{\log(1/\alpha) + \log(2\mu_{\text{avg}})} \right\rfloor, 1/\alpha\right\}$  ensures that (6.12) is satisfied. Thus if  $\mathcal{A}$  has expected average volume bounded as  $\leq \mu_{\text{avg}} \max\{\text{OPT}_{S_j^{(K)}}(\alpha), \text{minwidth}\}$  for all  $S_j^{(K)} \in \mathcal{S}$ , then  $\mathcal{A}$  must make

$$\begin{aligned} &\geq \min\left\{\left(\frac{\log(2/\text{minwidth})}{\log(1/\alpha) + \log(2\mu_{\text{avg}})} - 1\right) \alpha T, \frac{T}{2}\right\} \\ &= \Omega\left(\min\left\{\frac{\log(1/\text{minwidth})}{\log(\mu_{\text{avg}}) + \log(1/\alpha)} \alpha T, T\right\}\right) \\ &= \Omega\left(\min\left\{\frac{\log(1/\text{minwidth})}{\log(\mu_{\text{avg}})} \alpha^{1+\varepsilon'} T, T\right\}\right) \quad \text{for any } \varepsilon' > 0, \end{aligned}$$

mistakes in expectation on  $S_K^{(K)}$ . Similarly, Setting  $K = \min\left\{\left\lfloor \frac{\log(2/\text{minwidth})}{\log(2\mu_{\text{max}})} \right\rfloor, 1/\alpha\right\}$  ensures that (6.13) is satisfied and  $K \leq 1/\alpha$ . Thus if  $\mathcal{A}$  has expected maximum volume bounded as



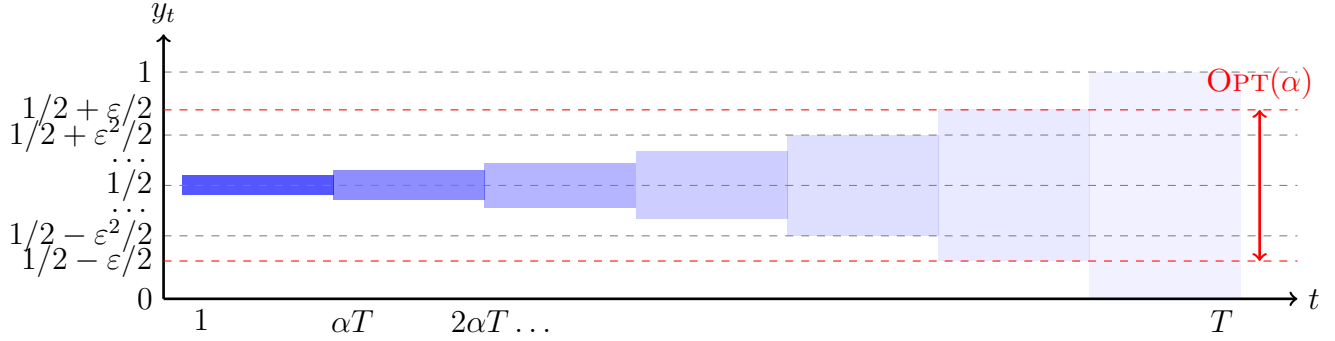


Figure 6.3: Symmetric version of the lower bound construction in Figure 6.2

$\leq \mu_{\max} \max\{\text{OPT}_{S_j}(\alpha), \text{minwidth}\}$  for all  $S_j^{(K)} \in \mathcal{S}^{(K)}$ , then  $\mathcal{A}$  must make

$$\begin{aligned} &\geq \min \left\{ \left( \frac{\log(2/\text{minwidth})}{\log(2\mu_{\max})} - 1 \right) \alpha T, \frac{T}{2} \right\} \\ &= \Omega \left( \min \left\{ \frac{\log(1/\text{minwidth})}{\log(\mu_{\max})} \alpha T, T \right\} \right) \end{aligned}$$

mistakes in expectation on  $S_K^{(K)}$ .

Finally, we note that while this argument considered the performance of  $\mathcal{A}$  on intervals with one end at 0 for ease of notation, we could have instead considered a symmetric version of the sequences, where we map each phase that is drawn

$$\sim_u [0, b] \rightarrow \sim_u \left[ \frac{1}{2} - \frac{1}{2}b, \frac{1}{2} + \frac{1}{2}b \right],$$

(see Figure 6.3). This leads to the same lower bounds on volume, while ensuring that the input distribution on each day is unimodal and symmetric around  $1/2$ . This gives a lower bound against algorithms that are competitive on such sequences.  $\square$

## 6.5 Lower Bound Against “Best of Both Worlds” Algorithms

We provide a lower bound that says that it is impossible to for a single algorithm to achieve the best possible guarantee both for arbitrary sequences and exchangeable sequences. Recall that for an arbitrary sequence  $S$ , [Corollary 6.2.2](#) shows that for choices of the multiplicative volume approximation  $\mu$ , scale lower bound `minwidth`, and target miscoverage  $\alpha$ , [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  plays intervals of average volume  $\leq \mu \max\{\text{OPT}_S(\alpha), \text{minwidth}\}$ , and makes  $O(\log(1/\text{minwidth})/\log(\mu)\alpha T)$  mistakes. [Theorem 6.4.1](#) tells us that this is optimal up to subpolynomial factors in  $\alpha$ .

For exchangeable sequences however, we can do better. [Corollary 6.3.2](#) tells us that [Algorithm 6.1](#) with  $R(t) = (1 - \alpha) + O(\sqrt{\log T/t})$  plays intervals of average volume  $\lesssim \text{OPT}_S(\alpha)$  and has expected coverage  $\geq (1 - \alpha) - O(\sqrt{\log T/t})$  (that is,  $\leq (1 + O(\sqrt{\log T/t}))\alpha T$  mistakes in expectation).

This leads to the natural question: is there a single algorithm that can achieve both guarantees? It turns out that these two bounds are indeed at odds. That is, achieving an algorithm that achieves the optimal tradeoff for arbitrary sequences must make a large number of mistakes on some exchangeable sequence, and an algorithm that achieves the optimal number of mistakes for exchangeable sequences must have a large multiplicative volume approximation.

The following lower bound gives a tradeoff between the two cases. [Theorem 6.3.4](#) tells us that [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  actually achieves this tradeoff, and thus it is optimal in the sense that it gets as close as possible to a “best of both worlds” guarantee. That is, for a fixed choice of  $\mu$ , `minwidth`, and  $\alpha$ , subject to playing intervals of average volume  $\leq \mu \max\{\text{OPT}_S(\alpha), \text{minwidth}\}$  on every arbitrary sequence  $S$ , [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$

makes the lowest possible number of mistakes on both arbitrary and exchangeable sequences. We remark, however, that the lower bound does not say anything about whether the multiplicative factor of  $\mu$  that [Algorithm 6.1](#) incurs in the exchangeable case is necessary.

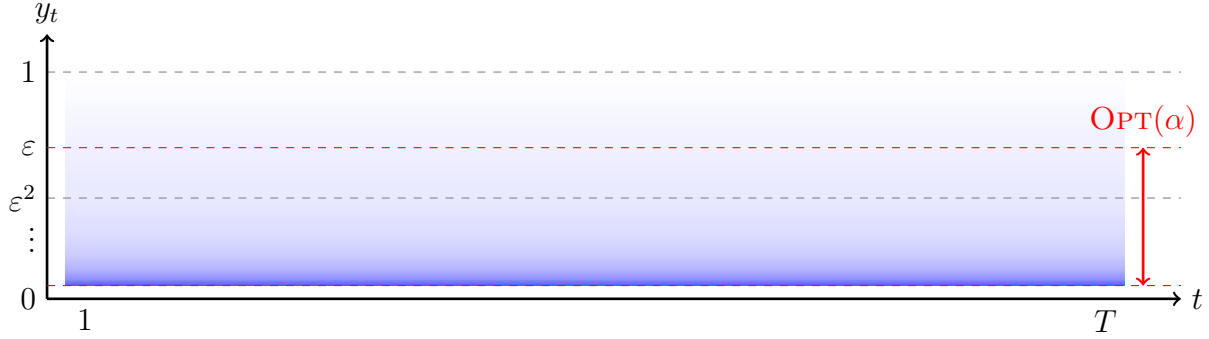
The construction of this lower bound, like the lower bound for arbitrary sequences, is inspired by observing the performance of [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  on i.i.d. sequences. The optimal algorithm for exchangeable sequences will play intervals that have coverage very close to the target  $1 - \alpha$  from very early on. [Algorithm 6.1](#) with  $R(t) = \frac{\alpha T}{t}$  however, will be more conservative and play intervals that have substantially lower coverage, to hedge against the possibility the scale of the problem will shrink in the future, and that many of the points seen so far were actually “outliers.” Intuitively, such a strategy should be optimal subject to a worst case guarantee.

The lower bound proof essentially directly argues that this conservative strategy is optimal. That is, we design a distribution  $\mathcal{D}$ , and then argue that any algorithm  $\mathcal{A}$  must fall into one of two categories.

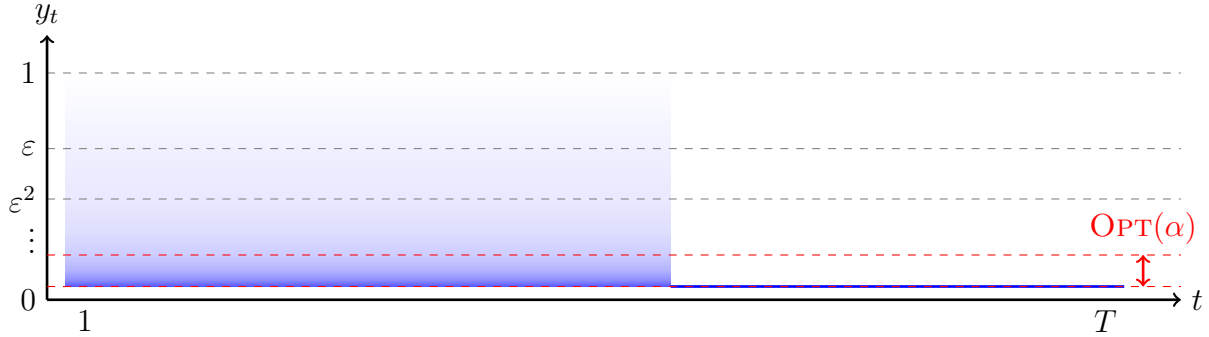
- (a) On every day  $t$ ,  $\mathcal{A}$  plays an interval that has expected miscoverage  $\geq \frac{1}{2} \cdot \frac{\alpha T}{t}$ . Summing over  $t$ , we see that this leads to  $\approx \ln(1/\alpha)\alpha T$  mistakes in expectation overall.
- (b) On some day  $t$ ,  $\mathcal{A}$  plays an interval with expected miscoverage  $\leq \frac{1}{2} \cdot \frac{\alpha T}{t}$ . Then, we show that  $\mathcal{A}$  has not been conservative, and there exists an alternate sequence on which  $\mathcal{A}$  incurs a large volume approximation.

A core part of the argument is in designing a distribution  $\mathcal{D}$  that achieves the desired tradeoff between the miscoverage of the algorithm and the multiplicative volume approximation  $\mu$ . We provide an illustration of the argument in [Figure 6.4](#).

**Theorem 6.5.1** (No “Best of Both Worlds” Algorithm). *Fix a scale lower bound  $\text{minwidth} >$*



(a) In this sequence, every day's input is drawn i.i.d. from  $\mathcal{D}$ . If an algorithm  $\mathcal{A}$  has miscoverage  $\geq \frac{1}{2} \cdot \frac{\alpha T}{t}$  on every day, it will make a large number ( $\geq \frac{1}{2} \ln(1/\alpha) \alpha T$ ) of mistakes over the whole sequence.



(b) In this sequence, the prefix is drawn i.i.d. from  $\mathcal{D}$ , but later the sequence drops off to be deterministically one value. If the algorithm  $\mathcal{A}$  does not play conservatively on the prefix of the sequence, it risks accumulating average volume much larger than  $\text{OPT}$ .

Figure 6.4: We illustrate the lower bound construction in [Theorem 6.5.1](#). The horizontal axis is the day  $t$  of the sequence and the vertical axis is the value of  $y_t$ .  $y_t$  is drawn according to a distribution  $\mathcal{D}$  that we illustrate with depth of color corresponding to the density of the distribution.  $\mathcal{D}$  has the property that the minimum volume interval that achieves miscoverage  $\leq \alpha e^i$  over  $\mathcal{D}$  is multiplicatively larger than the minimum volume interval that achieves miscoverage  $\leq \alpha e^{i+1}$ .

0 and target miscoverage rate  $\alpha > 0$ . Let  $\mathcal{A}$  be any algorithm, deterministic or randomized.

1. If  $\mathcal{A}$  plays sets with expected average volume  $\leq \mu_{\text{avg}} \max \{ \text{OPT}_S(\alpha), \text{minwidth} \}$  on every arbitrary order sequence  $S$ , for some  $\mu_{\text{avg}} > 0$ , then  $\mathcal{A}$  must make

$$\Omega \left( \min \left\{ \ln(1/\alpha), \frac{\ln(1/\text{minwidth})}{\ln \mu_{\text{avg}} + \ln(1/\alpha)} \right\} \alpha T \right)$$

*mistakes in expectation on some i.i.d. sequence.*

2. If  $\mathcal{A}$  plays sets with expected maximum volume  $\leq \mu_{\max} \max\{\text{OPT}_S(\alpha), \text{minwidth}\}$  on every arbitrary order sequence  $S$ , for some  $\mu_{\text{avg}} > 0$ , then  $\mathcal{A}$  must make

$$\Omega\left(\min\left\{\ln(1/\alpha), \frac{\ln(1/\text{minwidth})}{\ln \mu_{\max}}\right\} \alpha T\right)$$

*mistakes in expectation on some i.i.d. sequence.*

Furthermore, this lower bound holds against algorithms that only have to be competitive on sequences where the input on each day is drawn from a symmetric unimodal distribution with median  $1/2$ .

*Proof.* Let  $0 < \varepsilon \leq 1/2$  and  $0 \leq K \leq \ln(1/\alpha)$  be parameters that we will set later. We define a distribution  $\mathcal{D}^{(K)}$  over  $[0, 1]$ , such that

$$\begin{aligned} \mathbb{P}_{x \sim \mathcal{D}^{(K)}}[x < \varepsilon^{K+1}] &= 0, \\ \mathbb{P}_{x \sim \mathcal{D}^{(K)}}[x \leq \varepsilon^{1+K-i}] &= 1 - \alpha e^{K-i} && \text{for } 0 \leq i \leq K, \\ \mathbb{P}_{x \sim \mathcal{D}^{(K)}}[x \leq \varepsilon + t(1 - \varepsilon)] &= 1 - \alpha(1 - t) && \text{for } 0 \leq t \leq 1, \end{aligned} \tag{6.14}$$

where  $e$  is the base of the natural logarithm. Note that  $1 - \alpha e^K$  is a valid probability because  $\ln(\alpha e^K) = K - \ln(1/\alpha) \leq 0$  so  $\alpha e^K \leq 1$ .

We compute  $v^*(c)$ , which is the minimum volume of any set that achieves coverage  $c$  over

$\mathcal{D}^{(K)}$ . We take the c.d.f. of  $\mathcal{D}^{(K)}$ ,

$$F_{\mathcal{D}^{(K)}}(x) = \begin{cases} 0 & \text{for } 0 \leq x \leq \varepsilon^{K+1} \\ 1 - \frac{\alpha}{e} x^{1/\ln \varepsilon} & \text{for } \varepsilon^{K+1} \leq x \leq \varepsilon \\ 1 - \alpha \frac{1-x}{1-\varepsilon} & \text{for } \varepsilon \leq x \leq 1 \end{cases}.$$

From the c.d.f. we can compute the p.d.f. of  $\mathcal{D}^{(K)}$ . For  $0 \leq x < \varepsilon^{K+1}$ ,  $\mathcal{D}^{(K)}$  has density 0. At  $x = \varepsilon^{K+1}$ ,  $\mathcal{D}^{(K)}$  has a point mass of probability  $1 - \alpha e^K$ . For  $\varepsilon^{K+1} < x \leq \varepsilon$ , we compute the density of  $\mathcal{D}^{(K)}$  by taking the derivative of the c.d.f. with respect to  $x$ :

$$f_{\mathcal{D}^{(K)}}(x) = \begin{cases} \frac{\alpha}{e \ln(1/\varepsilon)} x^{-\frac{1}{\ln(1/\varepsilon)} - 1} & \text{for } \varepsilon^{K+1} < x < \varepsilon \\ \frac{\alpha}{1-\varepsilon} & \text{for } \varepsilon < x \leq 1 \end{cases}.$$

This p.d.f. is non-increasing with  $x$  for  $\varepsilon^{K+1} \leq x \leq 1$ . At  $x = \varepsilon^{K+1}$  the p.d.f. is infinite. The first piece is proportional to  $x$  to a constant negative power, and the second piece is constant. So it suffices to verify that the density is non-increasing where the two pieces meet. At  $x = \varepsilon$ , the first piece would give density  $\alpha \frac{1/\varepsilon}{\ln(1/\varepsilon)}$ . Since  $(\ln z) + 1 \leq z$  for all  $z > 0$ ,

$$\frac{1}{\varepsilon} - 1 \geq \ln(1/\varepsilon) \implies \frac{1/\varepsilon}{\ln(1/\varepsilon)} \geq \frac{1}{1-\varepsilon} \implies \alpha \frac{1/\varepsilon}{\ln(1/\varepsilon)} \geq \frac{\alpha}{1-\varepsilon}.$$

Since the density is non-increasing for  $x \geq \varepsilon^{K+1}$ , and  $\mathcal{D}^{(K)}$  has no mass on  $x < \varepsilon^{K+1}$ , for any coverage level  $0 \leq c \leq 1$ , a minimum volume set achieving coverage  $c$  over  $\mathcal{D}^{(K)}$  is of the form  $[\varepsilon^{K+1}, F_{\mathcal{D}^{(K)}}^{-1}(c)]$ , so

$$v^*(c) = F_{\mathcal{D}^{(K)}}^{-1}(c) - \varepsilon^{K+1},$$

(where we define  $F_{\mathcal{D}^{(K)}}^{-1}(0) = \varepsilon^{K+1}$ ). Since the derivative (p.d.f.) of  $F_{\mathcal{D}^{(K)}}$  is non-increasing for  $\varepsilon^{K+1} \leq x \leq 1$ ,  $F_{\mathcal{D}^{(K)}}(x)$  is concave over  $x \in [\varepsilon^{K+1}, 1]$ . So  $F_{\mathcal{D}^{(K)}}^{-1}(c)$  is convex over  $c \in [0, 1]$ ,

and  $v^*(c)$  is also convex.

The convexity of  $v^*(c)$  along with Jensen's inequality imply the following.

**Fact 6.5.2.** *Let  $\mathbf{I}$  be a (randomized) set of intervals over  $[0, 1]$  that have average expected coverage  $\geq c$  over  $\mathcal{D}^{(K)}$ . Then, the average expected volume of the intervals  $\mathbf{I}$  is*

$$\geq v^*(c) = F_{\mathcal{D}^{(K)}}^{-1}(c) - \varepsilon^{K+1}.$$

We use  $\mathcal{D}^{(K)}$  to design a lower bound instance. Let  $S^{(K)}$  be the sequence of  $T$  i.i.d. draws from  $\mathcal{D}^{(K)}$ . Assume also that  $\text{minwidth} \leq \varepsilon^{K+1}$ . We will analyze the behavior of an algorithm  $\mathcal{A}$  based on its expected performance on  $S^{(K)}$ . Fix an algorithm  $\mathcal{A}$ . Let  $M$  be the expected number of mistakes that  $\mathcal{A}$  makes on input  $S^{(K)}$ . Let  $M_t$  be the expected miscoverage of  $\mathcal{A}$  on day  $t$  on input  $S^{(K)}$ . That is,  $M_t$  is the marginal probability that  $\mathcal{A}$  does not capture a point on day  $t$  of  $S^{(K)}$ , and  $M = M_1 + \dots + M_t$ .

Let  $1 \leq w \leq T$  be a window length that we will set later, and let  $W_t$  denote the window (sequence of consecutive days) of length  $w$  that ends on day  $t$ . If  $\mathcal{A}$  has average expected miscoverage  $\leq \frac{1}{e} \frac{\alpha T}{t}$  over  $W_t$  on input  $S^{(K)}$ , then we say that  $\mathcal{A}$  “hits”  $W_t$ . Otherwise,  $\mathcal{A}$  has average expected miscoverage  $> \frac{1}{e} \frac{\alpha T}{t}$  over  $W_t$  on input  $S^{(K)}$ , and we say that  $\mathcal{A}$  “misses”  $W_t$ .

We analyze by cases. Either  $\mathcal{A}$  hits some window  $W_t$  for  $T/e^{K-1} \leq t \leq T/e$ , or  $\mathcal{A}$  misses every window  $W_t$  for  $T/e^{K-1} \leq t \leq T/e$ .

- (a)  $\mathcal{A}$  hits some window  $W_t$  for  $T/e^{K-1} \leq t \leq T/e$ . By definition, this means that  $\mathcal{A}$  has average expected coverage  $\geq 1 - \frac{1}{e} \frac{\alpha T}{t}$  over  $W_t$ . Fact 6.5.2 tells us that we can bound the expected volume of the intervals that  $\mathcal{A}$  plays over  $W_t$  using the inverse c.d.f.. We use

the original definition of  $\mathcal{D}^{(K)}$  (6.14) for convenience. For  $T/e^{K-1} \leq t \leq T/e$ , expected coverage of  $1 - \frac{1}{e} \frac{\alpha T}{t}$  corresponds to  $i = K + 1 - \ln(T/t)$ . Thus, the expected average volume of  $\mathcal{A}$  over  $W_t$  is

$$\geq \varepsilon^{1+K-i} - \varepsilon^{K+1} = \varepsilon^{\ln(T/t)} - \varepsilon^{K+1}.$$

Consider an alternate input sequence  $S_t^{(K)}$ , that has the first  $t$  days drawn i.i.d. from  $\mathcal{D}^{(K)}$ , and has future days deterministically set to  $\varepsilon^{K+1}$ . As  $T \rightarrow \infty$  (and the sampling error disappears, since  $t \geq T/e^{K+1}$  is also going to infinity),  $\text{OPT}_{S_t^{(K)}}(\alpha)$  will converge to the volume of the smallest interval that achieves coverage  $\frac{\alpha T}{t}$  over  $\mathcal{D}^{(K)}$ . For  $T/e^K \leq t \leq T/e$ , we fall into the middle case of (6.14) and we have

$$\text{OPT}_{S_t^{(K)}}(\alpha) \rightarrow \varepsilon^{1+\ln(T/t)} - \varepsilon^{K+1} \quad \text{as } T \rightarrow \infty. \quad (6.15)$$

Let  $\text{vol}_{\max}$  be the expected maximum volume of any interval played by  $\mathcal{A}$  on  $S_t^{(K)}$ , and let  $\text{vol}_{\text{avg}}$  be the expected average volume played by  $\mathcal{A}$  on  $S_t^{(K)}$ . By definition, we have that

$$\text{vol}_{\max} \leq \mu_{\max} \max\{\text{OPT}_{S_t^{(K)}}(\alpha), \text{minwidth}\}, \quad \text{vol}_{\text{avg}} \leq \mu_{\text{avg}} \max\{\text{OPT}_{S_t^{(K)}}(\alpha), \text{minwidth}\}.$$

To lower bound the multiplicative ratio, we need to be in the case where  $\text{OPT}$  is larger than  $\text{minwidth}$ . Using our assumption that  $\text{minwidth} \leq \varepsilon^{K+1}$ , it suffices to show that

$$\varepsilon^{1+\ln(T/t)} - \varepsilon^{K+1} \geq \varepsilon^{K+1} \iff \varepsilon^{1+\ln(T/t)} \geq 2\varepsilon^{K+1}.$$

Since  $\varepsilon \leq 1/2$ , we have  $2\varepsilon^{K+1} \leq \varepsilon^K$ , and the above is implied by  $\varepsilon^{1+\ln(T/t)} \geq \varepsilon^K$ . Finally



$\ln(T/t) \leq K-1 \iff T/e^{K-1} \leq t$  ensures that  $\text{OPT}_{S_t^{(K)}}(\alpha) \geq \text{minwidth}$ . Thus we have

$$\text{vol}_{\max} \leq \mu_{\max} \text{OPT}_{S_t^{(K)}}(\alpha), \quad \text{vol}_{\text{avg}} \leq \mu_{\text{avg}} \text{OPT}_{S_t^{(K)}}(\alpha). \quad (6.16)$$

Now we analyze the performance of  $\mathcal{A}$  on  $S_t^{(K)}$ . Since  $S_t^{(K)}$  is identical to  $S^{(K)}$  for the first  $t$  days,  $\mathcal{A}$  must have the same expected performance over  $W_t$  on  $S_t^{(K)}$  as it does on  $S^{(K)}$ . Thus by our earlier bound, we have that the expected average volume of  $\mathcal{A}$  over  $W_t$  of  $S_t^{(K)}$  is  $\geq \varepsilon^{\ln(T/t)} - \varepsilon^{K+1}$ . We have that

$$\text{vol}_{\max} \geq \varepsilon^{\ln(T/t)} - \varepsilon^{K+1}, \quad \text{vol}_{\text{avg}} \geq \frac{w}{T}(\varepsilon^{\ln(T/t)} - \varepsilon^{K+1}).$$

Combining with (6.16) and our bound on  $\text{OPT}$  (6.15) we get

$$\mu_{\max} \geq \frac{\varepsilon^{\ln(T/t)} - \varepsilon^{K+1}}{\varepsilon^{1+\ln(T/t)} - \varepsilon^{K+1}} \geq \frac{1}{\varepsilon}, \quad \mu_{\text{avg}} \geq \frac{w}{T} \cdot \frac{\varepsilon^{\ln(T/t)} - \varepsilon^{K+1}}{\varepsilon^{1+\ln(T/t)} - \varepsilon^{K+1}} \geq \frac{w}{T} \cdot \frac{1}{\varepsilon}. \quad (6.17)$$

(b)  $\mathcal{A}$  misses every window  $W_t$  for  $T/e^{K-1} \leq t \leq T/e$ . We lower bound the expected number of mistakes  $\mathcal{A}$  makes on  $S^{(K)}$ .

$$\begin{aligned} M &= \sum_{t=1}^T M_t \geq \sum_{t=w}^T \frac{1}{w} \sum_{j=0}^w M_{t-j} \\ &\geq \sum_{t=\max\{w, T/e^{K-1}\}}^{T/e} \frac{1}{e} \frac{\alpha T}{t} \\ &= \frac{\alpha T}{e} (H_{T/e} - H_{\max\{w, T/e^{K-1}\}-1}) \quad H_n \text{ the } n\text{th harmonic number} \\ &\geq \frac{\alpha T}{e} (\ln(T/e) - (\ln(\max\{w, T/e^{K-1}\}) + 1)) \\ &= \frac{1}{e} (\min\{\ln(T/ew), K-2\} - 1) \alpha T. \end{aligned} \quad (6.18)$$

We set  $w/T = e^{-K}$ . Then, assuming  $\text{minwidth} \leq \varepsilon^{K+1}$ , since  $K \leq \ln(1/\alpha)$ , we have that  $\mathcal{A}$  must either have

$$\mu_{\max} \geq \frac{1}{\varepsilon} \quad \text{and} \quad \mu_{\text{avg}} \geq \frac{\alpha}{\varepsilon}$$

from (6.17), or it must make

$$\geq \frac{1}{e}(K-3)\alpha T$$

mistakes in expectation on the i.i.d. sequence  $S^{(K)}$ , from (6.18).

To get a bound based on  $\mu_{\max}$ , we set  $\varepsilon = \frac{1}{\mu_{\max}}$ , which is possible as long as  $\mu_{\max} \geq 2$ . We set  $K$  such that  $\text{minwidth} = \varepsilon^{K+1}$ , which is possible for  $\text{minwidth} \geq (1/\mu_{\max})^{\ln(1/\alpha)+1}$ . This gives us that  $K = \Omega(\ln(1/\text{minwidth})/\ln \mu_{\max})$ . As  $\text{minwidth}$  gets smaller, the problem only becomes more general, so we have that for any  $\text{minwidth} > 0$  and  $\mu_{\max} \geq 2$ , either  $\mathcal{A}$  has maximum multiplicative approximation  $\geq \mu_{\max}$  or  $\mathcal{A}$  makes

$$\Omega\left(\min\left\{\ln(1/\alpha), \frac{\ln(1/\text{minwidth})}{\ln \mu_{\max}}\right\}\alpha T\right)$$

mistakes in expectation on some i.i.d. sequence.

To get a bound based on  $\mu_{\text{avg}}$ , we set  $\varepsilon = \frac{\alpha}{\mu_{\text{avg}}}$ , which is possible as long as  $\mu_{\text{avg}} \geq 2\alpha$ . We set  $K$  such that  $\text{minwidth} = \varepsilon^{K+1}$ , which is possible for  $\text{minwidth} \geq (\alpha/\mu_{\text{avg}})^{\ln(1/\alpha)+1}$ . This gives us that  $K = \Omega(\ln(1/\text{minwidth})/(\ln \mu_{\text{avg}} + \ln(1/\alpha)))$ . As  $\text{minwidth}$  gets smaller, the problem only becomes more general, so we have that for any  $\text{minwidth} > 0$  and  $\mu_{\text{avg}} \geq 2\alpha$ , either  $\mathcal{A}$  has average multiplicative approximation  $\geq \mu_{\text{avg}}$  or  $\mathcal{A}$  makes

$$\Omega\left(\min\left\{\ln(1/\alpha), \frac{\ln(1/\text{minwidth})}{\ln \mu_{\max} + \ln(1/\alpha)}\right\}\alpha T\right)$$

mistakes in expectation on some i.i.d. sequence.

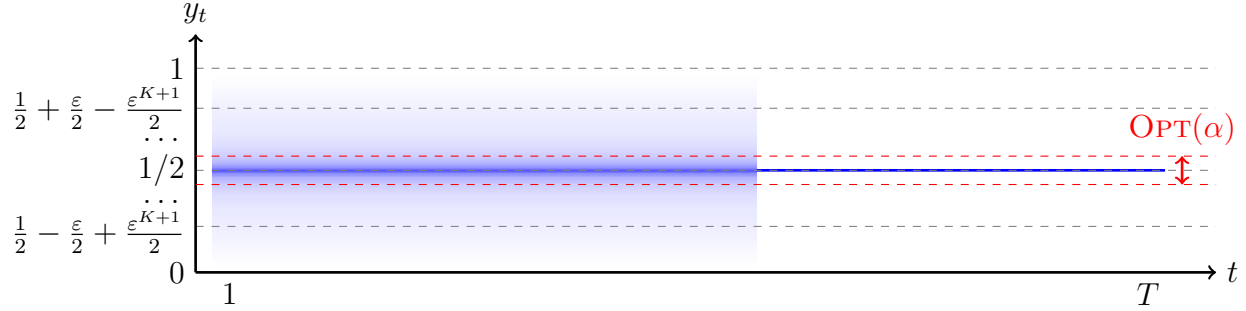


Figure 6.5: Symmetric version of i.i.d. lower bound construction in Figure 6.4

Finally, we note that while this argument considered the performance of  $\mathcal{A}$  on intervals with one end at  $\varepsilon^{K+1}$  for ease of notation, we could have instead considered a symmetric version of the sequences, where we map each day  $t$ 's input that is drawn  $y_t \sim \mathcal{D}_t$  to

$$y'_t = \frac{1}{2} + \frac{\sim_u \{\pm 1\}}{2} (y_t - \varepsilon^{K+1}).$$

see for example Figure 6.5. This leads to the same lower bounds on volume. This also ensures that the input distribution on each day is unimodal and symmetric around median  $1/2$ , because the distributions  $\mathcal{D}_t$  in this construction are not supported below  $\varepsilon^{K+1}$ , and have decreasing density with  $y$  increasing from  $\varepsilon^{K+1}$ . Thus the same argument gives a lower bound against algorithms that are competitive on such sequences.  $\square$

## 6.6 Future Directions

In this work we provide a new framework to measure the efficiency of online conformal prediction algorithms. We apply this to studying the setting where the response variables are in the unit interval  $[0, 1]$ , the notion of efficiency is minimizing the volume (Lebesgue measure) of the prediction set, and the family of sets that we compete against is subintervals  $[a, b] \subseteq [0, 1]$ . We study this in the unsupervised (featureless) setting for arbitrary and

exchangeable input sequences. We view this as the most basic and fundamental setting in which to explore efficiency optimality. We believe that there are many opportunities for future work to explore efficiency optimality for online conformal prediction in a broad range of other settings that are widely studied in conformal prediction and present new challenges. We provide a few examples of these settings.

- *Other prediction sets.* Even in the setting where the target variables live in the unit interval  $[0, 1]$ , there are other families of prediction sets that we can consider competing against. For example, in the standard setting, in [Chapter 3](#) we provide an algorithm that competes with the volume-optimal *union of  $k$  intervals*. Can we design an algorithm that competes with unions of intervals in the online setting?
- *Other geometries.* It would be interesting to extend this result to other spaces of target variables  $\mathcal{Y}$ . For example, what if instead of being points in the unit interval, our target variables were points in high-dimensional space? Are there natural classes of prediction sets that we can compete against in the online setting?
- *Other notions of efficiency.* In this work we consider minimizing the volume (Lebesgue measure) of the prediction sets played. Depending on the task that the conformal predictor is used for, there could be other natural notions of efficiency. For example, we could consider minimizing other measures of the prediction set. Alternatively, the cost of a prediction set could correspond to some downstream decision-making cost. Can we design online conformal prediction algorithms that minimize other natural notions of cost on the sets played?
- *Other forms of distribution shift.* In this work, we consider exchangeable sequences, which exhibit no distribution shift, and arbitrary sequences that exhibit arbitrary distribution shifts. One conclusion of this work is that, even in our basic setting, these

two types of sequences admit very different guarantees. Are there natural but more benign forms of distribution shift for which we can design algorithms that have stronger guarantees than what is possible for fully arbitrary sequences?

- *Supervised settings and conditional guarantees.* In the supervised setting of conformal prediction, each target variable  $y_t \in \mathcal{Y}$  is associated by a feature variable  $x_t \in \mathcal{C}$  which may be predictive of it. On each day, the algorithm observes  $x_t$  and then produces a prediction set  $C(x_t)$  to try to capture  $y_t$ . It would be interesting to explore efficiency optimality in this supervised setting. This could involve interesting new notions of optimality. Now that we have access to the  $x_t$ , it does not necessarily make sense to compete with the best *fixed* prediction set in hindsight, as that eliminates strategies that vary as a function of  $x_t$ . It would be interesting to use the features to get guarantees against stronger baselines that are allowed to depend on the features.

## 6.7 Chapter Acknowledgements

We thank Aravindan Vijayaraghavan for helpful discussions, and for the proof that uniform convergence extends to exchangeable sequences ([Section 2.8](#)).

# Chapter 7

## Statistical Limits of Learning

### Prediction Sets

This chapter is based on joint work with Shaddin Dughmi and Aravindan Vijayaraghavan.

#### 7.1 Volumetric Learnability

In [Chapter 3](#), we introduced the notion of  $\mathcal{C}$ -restricted volume optimality. Now, we ask what sets  $\mathcal{C}$  are *statistically* learnable with respect to restricted volume optimality.

**Definition 7.1.1** ( $(\varepsilon, \delta)$ -Volumetric Learnability). *A prediction set family  $\mathcal{C}$  is volumetrically learnable with approximation  $\varepsilon > 0$ , coverage slack factor  $\delta > 0$ , miscoverage rate  $\alpha$ , and  $m$  samples, if there is a function  $f : (\mathbb{R}^d)^m \rightarrow \mathcal{C}$ , such that for  $Y_1, \dots, Y_m \sim_{\text{iid}} \mathbb{R}^d$ , we have that*

$$\begin{aligned}\hat{C} &= f(Y_1, \dots, Y_m), \\ P(\hat{C}) &\geq 1 - \alpha - \delta,\end{aligned}\tag{7.1}$$

$$\text{vol}(\widehat{C}) \leq (1 + \varepsilon) \inf_{C \in \mathcal{C}, P(C) \geq 1 - \alpha} \text{vol}(C) \quad (7.2)$$

with high probability.

The above definition restates the definition of restricted volume optimality ([Problem 2.2.1](#)) in a way that is easier to think about learnability.

In [Chapter 3](#), we observe that bounded VC-dimension of a family  $\mathcal{C}$  suffices to establish volumetric learnability. This is because classes with bounded VC-dimension exhibit *uniform convergence*. This means that we can estimate the coverage of every set in the family simultaneously with bounded a number of samples, so we can easily find the smallest set that achieves coverage.

**Theorem 7.1.2** (Bounded VC-dimension suffices for volumetric learnability). *For a class of prediction sets  $\mathcal{C}$ , it is possible to volumetrically learn  $\mathcal{C}$  with approximation  $\varepsilon = 0$ , coverage slack factor  $\delta > 0$ , with probability  $\geq 1 - \eta$ , with*

$$m = O\left(\frac{\text{VC}(\mathcal{C}) + \log(1/\eta)}{\delta^2}\right) \text{ samples,}$$

where  $\text{VC}(\mathcal{C})$  is the VC-dimension of  $\mathcal{C}$ .

*Proof.* By uniform convergence for classes of bounded VC-dimension (the fundamental theorem of statistical learning [see e.g. Theorem 6.8 in [Shalev-Shwartz and Ben-David, 2014](#)]), we have that  $m = O\left(\frac{\text{VC}(\mathcal{C}) + \log(1/\eta)}{\delta^2}\right)$  samples suffice to ensure that for samples  $Y_1, \dots, Y_m \sim_{\text{iid}} P$ , with probability  $\geq 1 - \eta$ , for all  $C \in \mathcal{C}$  simultaneously,

$$P(C) - \delta \leq \frac{1}{m} \sum_{i \in [m]} \mathbf{1}[Y_i \in C] \leq P(C) + \delta. \quad (7.3)$$

Take the function  $f(Y_1, \dots, Y_m)$  that selects the smallest volume set  $C \in \mathcal{C}$  that achieves

empirical coverage  $\geq 1 - \alpha - \delta$  over the samples  $Y_i$ .

We have that by (7.3), the optimal prediction set for  $\mathcal{C}$ ,  $C^* = \min_{C \in \mathcal{C}: P(C) \geq 1-\alpha} \text{vol}(C)$  achieves empirical coverage  $\geq 1 - \alpha - \delta$  over the samples. Therefore, conditioned on (7.3) holding,  $C^*$  is a feasible choice for  $\hat{C} = f(Y_1, \dots, Y_m)$ , thus

$$\text{vol}(\hat{C}) \leq \text{vol}(C^*).$$

Since the set selected by  $f(Y_1, \dots, Y_m)$  achieves empirical coverage  $\geq 1 - \alpha - \delta$ , conditioned on (7.3), it achieves coverage  $\geq 1 - \alpha - 2\delta$  over  $P$ .

Taking  $\delta = \delta'/2$ , we establish the claim.  $\square$

We know that learnability in the sense of classification error is governed by the VC-dimension. That is, a class  $\mathcal{C}$  is PAC-learnable if and only if it has bounded VC-dimension, and the sample complexity of learning the set is governed by the VC-dimension of  $\mathcal{C}$  [see e.g. Theorem 6.8 in [Shalev-Shwartz and Ben-David, 2014](#)].

This leads us to the following natural question.

**Question 7.1.3.** *Is the volumetric learnability ([Definition 7.1.1](#)) of a class  $\mathcal{C}$  governed by the VC-dimension of  $\mathcal{C}$ ?*

In this chapter we show that this is not the case. There exist classes  $\mathcal{C}$  with unbounded VC-dimension, which are  $(\varepsilon, \delta)$ -volumetrically learnable for all  $\varepsilon, \delta > 0$ . In fact, a simple example is the class of *convex sets* in  $d$ -dimensions, which have unbounded VC-dimension for  $d \geq 2$ . For the case of large volume approximations ( $\mu \approx d^d$ ), this follows (computationally efficiently) from learning ellipsoids, which is a class of bounded VC dimension, along with John's theorem (see [Section 5.5.1](#) for details). However, establishing small volume



approximations of  $(1 + \varepsilon)$  requires more work.

## 7.2 Convex Sets are Volumetrically Learnable

We show that the family of convex sets in  $\mathbb{R}^d$  is  $(\varepsilon, \delta)$ -volumetrically learnable (Definition 7.1.1) for all  $\varepsilon, \delta > 0$ . This is interesting, because it is well-known that the set of convex sets has unbounded VC-dimension, even in 2-dimensions ( $d = 2$ ).

**Theorem 7.2.1** (Convex sets are volumetrically learnable). *The set  $\mathcal{K}_d$  of convex sets in  $\mathbb{R}^d$  is  $(\varepsilon, \delta)$ -volumetrically learnable (Definition 7.1.1) for any  $\varepsilon, \delta > 0$ . That is, for any  $\eta > 0, \alpha \in [0, 1]$ , there is a function that given*

$$m = \exp \left( O \left( \frac{d^2 \log 1/\eta}{\delta^2(1 - \alpha - \delta)} \right) + O \left( \frac{1}{\eta \delta^2} \right) \right) \cdot O \left( \frac{d}{\delta^2} \left( \frac{18d^d \cdot \Gamma(d/2 + 1)}{\Gamma(d/2) \cdot \varepsilon} \right)^{2d} \right)$$

*i.i.d. samples from an arbitrary distribution  $P$ , with probability  $\geq 1 - \eta$ , outputs a convex set  $\hat{K}$  such that  $\hat{K}$  achieves coverage over  $P$ ,*

$$P(\hat{K}) \geq (1 - \alpha) - \delta,$$

*and  $\hat{K}$  has approximately minimal volume,*

$$\text{vol}(\hat{K}) \leq (1 + \varepsilon) \text{vol}(K^*),$$

*where  $K^*$  is the minimum volume convex set that captures at least  $1 - \alpha$  probability mass of  $P$ . (Above,  $\Gamma(\cdot)$  is a function that generalizes factorial to non-integers, and arises in the expressions for the volume and surface-area of the Euclidean ball in high dimensions.)*

The high level strategy is to start by approximating  $K^*$  by an ellipsoid. By John's theorem,

we know that there is an ellipsoid that contains  $K^\star$  and has a bounded approximation in volume to  $K^\star$ . Since ellipsoids have bounded VC-dimension, they allow uniform convergence, so we can hope to find this ellipsoid.

Once we have captured (most of)  $K^\star$  in an ellipsoid that has a bounded volume approximation to  $K^\star$ , we can use this to precondition the space, so that  $K^\star$  is now contained in the unit ball, and takes up at least some fixed positive volume. We can then take a fine grid, or net, over the bounded space, and approximate  $K^\star$  as a union of cells of the grid. Taking the grid finer and finer gives us a better and better additive volume approximation in the bounded space. This translates to a multiplicative approximation in the original space.

**Lemma 7.2.2** (Learning Bounding Ellipsoid). *Let  $P$  be an arbitrary distribution over  $\mathbb{R}^d$ , and let  $K^\star$  be the minimum volume convex set that contains at least  $1 - \alpha$  probability mass of  $P$ . Fix parameters  $\delta, \eta > 0$ . With*

$$m = O\left(\frac{d^2 \log 1/\eta}{\delta^2(1 - \alpha - \delta)}\right) + O\left(\frac{1}{\eta\delta^2}\right)$$

*samples from  $P$ , with probability at least  $1 - \eta$ , we can learn a list of  $2^m$  ellipsoids  $\mathcal{E}$ , such that there exists an  $\hat{E} \in \mathcal{E}$  such that captures most of the probability mass of  $K^\star$ ,*

$$P(\hat{E} \cap K^\star) \geq (1 - \alpha) - \delta,$$

*and has volume that is an approximation to  $K^\star$ ,*

$$\text{vol}(\hat{E}) \leq d^d \cdot \text{vol}(K^\star).$$

*Proof.* Draw a set  $S$  of samples from  $P$ , with

$$|S| = O\left(\frac{d^2 \log 1/\eta_1}{\delta^2(1-\alpha-\delta)}\right) + O\left(\frac{1}{\eta_1 \delta^2}\right),$$

where we set  $\eta_1$  later. The first term ensures that  $(1-\alpha-\delta)|S|$  samples suffice for uniform convergence for ellipsoids up to coverage error  $\delta$  with probability at least  $1-\eta_1$ , since the class of ellipsoids has VC-dimension  $O(d^2)$ .

The second term ensures that  $K^\star$  has coverage at least  $1-\alpha-\delta$  over  $S$ , with probability  $\geq 1-\eta_1$ .  $|S \cap K^\star|$  follows a binomial distribution with  $|S|$  trials and probability  $1-\alpha$ . Thus, the empirical coverage of  $K^\star$  can be bounded by Chebyshev's inequality as

$$\frac{|S \cap K^\star|}{|S|} \geq (1-\alpha) - \sqrt{\frac{\alpha(1-\alpha)}{\eta_1 |S|}} \geq (1-\alpha) - \delta \quad \text{w.p.} \geq 1-\eta_1. \quad (7.4)$$

Let  $\mathcal{E}$  be the set of minimum enclosing ellipsoids for all  $2^{|S|}$  subsets of  $S$ . One (unknown) ellipsoid  $\hat{E} \in \mathcal{E}$  is the minimum volume enclosing ellipsoid for  $S \cap K^\star$ .

Let  $P_{|K^\star}$  be the distribution  $P$  conditioned on  $K^\star$ . We condition on the uniform convergence of ellipsoids over  $P_{|K^\star}$ .  $S \cap K^\star$  is a collection of independent samples from  $P_{|K^\star}$ . Thus by (7.4), we have that the empirical coverage of  $\hat{E}$  on  $(1-\alpha)-\delta$  independent samples from  $P_{|K^\star}$  is 1. By uniform convergence of ellipsoids over  $P_{|K^\star}$ , we have

$$P_{|K^\star}(\hat{E}) \geq 1-\delta \quad \text{w.p.} \geq 1-\eta_1. \quad (7.5)$$

Thus, we have

$$\begin{aligned}
 P(\widehat{E} \cap K^\star) &= P(K^\star) \cdot P_{|K^\star}(\widehat{E}) \\
 &\geq (1 - \alpha)(1 - \delta) \\
 &\geq (1 - \alpha) - \delta, \quad \text{w.p. } 1 - 2\eta_1,
 \end{aligned} \tag{7.6}$$

where the failure probability comes from a union bound over (7.4) and (7.5).

We also have that  $\widehat{E}$  is the minimum volume enclosing ellipsoid for  $S \cap K^\star$ . Thus, by John's theorem [John, 1948], we have

$$\text{vol}(\widehat{E}) \leq d^d \cdot \text{vol}(\text{convhull}(S \cap K^\star)) \leq d^d \cdot \text{vol}(K^\star).$$

Taking  $\eta_1 = \eta/2$  establishes the claim.  $\square$

**Lemma 7.2.3** (Learning convex set in bounding ball). *Let  $P$  be an arbitrary distribution over  $\mathbb{R}^d$ . Let  $K^\star$  be the minimum volume convex set that is contained in the unit ball, that contains at least  $1 - \alpha$  probability mass of  $P$ , if it exists. Let  $\varepsilon, \delta, \eta > 0$  be arbitrary. With*

$$m = O\left(\frac{d \log 1/\eta}{\delta^2} \left(\frac{18\pi^{d/2}}{\Gamma(d/2) \cdot \varepsilon}\right)^{2d}\right)$$

*samples, with probability  $\geq 1 - \eta$ , if  $K^\star$  exists, we can learn a convex set  $\widehat{K}$  such that  $\widehat{K}$  achieves the target coverage,*

$$P(\widehat{K}) \geq (1 - \alpha) - \delta,$$

*and the volume of  $\widehat{K}$  is an additive approximation to the volume of  $K^\star$ ,*

$$\text{vol}(\widehat{K}) \leq \text{vol}(K^\star) + \varepsilon.$$

*Proof.* Let  $N$  be an  $\varepsilon_2$  net for the unit ball in  $\mathbb{R}^d$ . That is  $N$  is a  $\varepsilon_2$  cover for the unit ball, with  $|N| \leq (3/\varepsilon_2)^d$ . Let  $S$  be a set of samples drawn i.i.d. from  $P$ , with

$$|S| = O\left(\frac{d \log 1/\eta}{\delta_2^2}\right),$$

where  $\varepsilon_2, \delta_2 > 0$  are parameters that we will set later.

The number of samples ensures the uniform convergence of balls up to coverage error  $\delta_2$  with probability at least  $1 - \eta$ . Conditioning on uniform convergence, for any set  $C$  that is a union of balls around net points, we can estimate the coverage of  $C$  over  $P$  using the samples  $S$ , up to additive error  $\delta_2|N|$ . That is, with probability  $\geq 1 - \eta$ , simultaneously for all subsets of the net  $N' \subseteq N$

$$C_{N'} = \bigcup_{x \in N'} B(x, \varepsilon_2),$$

where  $B(x, \varepsilon_2)$  is the Euclidean ball with center  $x$  and radius  $\varepsilon_2$ , we have

$$\left| \frac{|C_{N'} \cap S|}{|S|} - P(C_{N'}) \right| \leq \delta_2|N|. \quad (7.7)$$

We generate the following prediction set  $\hat{K}$ . If  $E$  is low-dimensional in the ambient space, we do the following with respect to  $d'$ , the true dimension of  $E$ .

$$\hat{K} = \underset{K \subseteq B(0,1)}{\operatorname{argmin}} \operatorname{vol}(K) \quad (7.8)$$

$$\text{s.t. } \exists N' \subseteq N,$$

$$K = \operatorname{convhull}(C_{N'}),$$

$$\frac{|C_{N'} \cap S|}{|S|} \geq (1 - \alpha) - \delta_2|N|.$$

If no such set is feasible, we output “fail.” Conditioning on (7.7), we have that

$$P(\widehat{K}) \geq (1 - \alpha) - 2\delta_2|N|. \quad (7.9)$$

Let  $\widetilde{C}$  be the union of net balls that intersect  $K^\star$ ,

$$\widetilde{C} = \bigcup_{\substack{x \in N: \\ B(x, \varepsilon_2) \cap K^\star \neq \emptyset}} B(x, \varepsilon_2).$$

Since  $K^\star \subseteq \widetilde{C}$ , we have that  $P(\widetilde{C}) \geq 1 - \alpha$ , so conditioned on (7.7) we have that

$$\frac{|\widetilde{C} \cap S|}{|S|} \geq (1 - \alpha) - \delta_2|N|.$$

Thus,  $\widetilde{K} = \text{convhull}(\widetilde{C})$  is feasible for (7.8), so  $\text{vol}(\widehat{K}) \leq \text{vol}(\widetilde{K})$ .

Now we bound the volume of  $\widetilde{K}$ . We know that  $\widetilde{K}$  is the convex hull of  $K^\star$  and some collection of points that have distance at most  $2\varepsilon_2$  to  $K^\star$ . Thus,

$$\widetilde{K} \subseteq K^\star + B(0, 2\varepsilon_2),$$

where here  $+$  is the Minkowski sum. Using a property of the Minkowski sum for a convex set and a ball, we have

$$\text{vol}(K^\star + B(0, 2\varepsilon_2)) = \text{vol}(K^\star) + \text{surfacearea}(K^\star) \cdot 2\varepsilon_2 + \text{vol}(B(0, 2\varepsilon_2)).$$

Since  $K^\star$  is a convex set that is contained in the unit ball, it can have surface area at most

that of the unit ball, which is  $\frac{2\pi^{d/2}}{\Gamma(d/2)}$ .<sup>1</sup> Thus,

$$\begin{aligned} \text{vol}(\hat{K}) &\leq \text{vol}(\tilde{K}) \leq \text{vol}(K^\star) + \frac{2\pi^{d/2}}{\Gamma(d/2)} \cdot 2\varepsilon_2 + \frac{\pi^{d/2}}{\Gamma(\frac{d}{2} + 1)} \varepsilon_2^d \\ &= \text{vol}(K^\star) + 6 \cdot \frac{\pi^{d/2}}{\Gamma(d/2)} \cdot \varepsilon_2. \end{aligned}$$

Taking  $\varepsilon = 6 \cdot \frac{\pi^{d/2}}{\Gamma(d/2)} \cdot \varepsilon_2$ , and  $\delta = 2\delta_2|N|$ , we get that

$$\begin{aligned} \varepsilon_2 &= \frac{\Gamma(d/2)}{6\pi^{d/2}} \cdot \varepsilon \\ |N| &\leq \left( \frac{18\pi^{d/2}}{\Gamma(d/2) \cdot \varepsilon} \right)^d \\ \delta_2 &= \frac{\delta}{2|N|} \\ &\geq \frac{\delta}{2} \cdot \left( \frac{\Gamma(d/2) \cdot \varepsilon}{18\pi^{d/2}} \right)^d \\ |S| &= O\left( \frac{d \log 1/\eta}{\delta_2^2} \right) \\ &= O\left( \frac{d \log 1/\eta}{\delta^2} \left( \frac{18\pi^{d/2}}{\Gamma(d/2) \cdot \varepsilon} \right)^{2d} \right), \end{aligned}$$

samples suffice so that, with probability  $\geq 1 - \eta$ , we find a  $\hat{K}$  with  $P(\hat{K}) \geq (1 - \alpha) - \delta$ , and  $\text{vol}(\hat{K}) \leq \text{vol}(K^\star) + \varepsilon$ .  $\square$

*Proof of Theorem 7.2.1.* First we use

$$m_1 = O\left( \frac{d^2 \log 1/\eta_1}{\delta_1^2(1 - \alpha - \delta_1)} \right) + O\left( \frac{1}{\eta_1 \delta_1^2} \right)$$

and Lemma 7.2.2 to learn a list of  $2^m$  ellipsoids  $\mathcal{E}$ , containing a  $\hat{E} \in \mathcal{E}$  such that captures most of the probability mass of  $K^\star$ ,  $P(\hat{E} \cap K^\star) \geq (1 - \alpha) - \delta_1$ , and has volume that is an

---

<sup>1</sup>Where  $\Gamma(\cdot)$  generalizes the factorial to non-integers.

approximation to  $K^\star$ ,  $\text{vol}(\widehat{E}) \leq d^d \cdot \text{vol}(K^\star)$ .

We construct a list  $\mathcal{L}$  of potential prediction sets. For each  $E \in \mathcal{E}$ , let  $P_E$  denote the probability distribution normalized by  $E$ . That is, for

$$E = \{(x - \mu_E^\top) M_E^{-1} (x - \mu_E) \leq 1 : x \in \mathbb{R}^d\},$$

let  $P_E$  be the distribution of  $M_E^{-1/2}(x - \mu_E)$  for  $x \sim P$ . We apply the method from [Lemma 7.2.3](#), with miscoverage parameter  $\alpha_2 = \alpha - \delta_1$  to samples from  $P_E$  with

$$m_2 = O\left(\frac{d \log 1/\eta_2}{\delta_2^2} \left(\frac{18\pi^{d/2}}{\Gamma(d/2) \cdot \varepsilon_2}\right)^{2d}\right)$$

fresh samples for each ellipsoid  $E$ , for parameters  $\eta_2, \delta_2, \varepsilon_2 > 0$  to be set later. This ensures that with probability at least  $1 - \eta_2$ , if  $\widehat{E}$  contains at least  $(1 - \alpha) - \delta_1$  probability mass of  $P$ , then [Lemma 7.2.3](#) will learn a set  $\widehat{K}_E$  that has coverage at least  $(1 - \alpha) - \delta_1 - \delta_2$  over  $P_E$ , and have volume bounded as  $\text{vol}(\widehat{K}_E) \leq \text{vol}(K_E^\star) + \varepsilon$ , where  $K_E^\star$  is the minimum volume convex set contained in the unit ball that has coverage at least  $(1 - \alpha) - \delta_1$  over  $P_E$ . If [Lemma 7.2.3](#) succeeds, we add  $(M_E^{1/2} \widehat{K}_E) + \mu_E$  to  $\mathcal{L}$ . If it fails, we do not add any set to  $\mathcal{L}$  for  $E$ . Finally, our method outputs the minimum volume  $\widehat{K} \in \mathcal{L}$ .

We establish coverage of  $\widehat{K}$ . Each iteration of [Lemma 7.2.3](#) succeeds with probability  $1 - \eta_2$ . By a union bound, with probability at least  $1 - |\mathcal{E}|\eta_2$ , we have that every set in  $\mathcal{L}$  achieves coverage at least  $(1 - \alpha) - \delta_1 - \delta_2$  over  $P$ . Thus,  $\widehat{K}$  also achieves this coverage.

Now we establish the volume of  $\widehat{K}$ . Let  $\widehat{E} \in \mathcal{E}$  be the ellipsoid guaranteed by [Lemma 7.2.2](#). Conditioning on the success of [Lemma 7.2.3](#),

$$\text{vol}(K_{\widehat{E}}) \leq \text{vol}(M_{\widehat{E}}^{-1/2}(K^\star - \mu_{\widehat{E}})) + \varepsilon_2.$$



Because  $\text{vol}(\hat{E}) \leq d^d \cdot \text{vol}(K^\star)$  (Lemma 7.2.2), we have that

$$\varepsilon_2 = \varepsilon_2 \cdot \frac{\text{vol}(B(0, 1))}{\text{vol}(B(0, 1))} \leq \varepsilon_2 \cdot \frac{d^d \cdot \text{vol}(M_{\hat{E}}^{-1/2}(K^\star - \mu_{\hat{E}}))}{\text{vol}(B(0, 1))}.$$

This gives us that

$$\begin{aligned} \text{vol}(K_{\hat{E}}) &\leq \left(1 + \varepsilon_2 \cdot \frac{d^d}{\text{vol}(B(0, 1))}\right) \text{vol}(M_{\hat{E}}^{-1/2}(K^\star - \mu_{\hat{E}})). \\ \text{vol}((M_{\hat{E}}^{-1/2}K_{\hat{E}}) + \mu) &\leq \left(1 + \varepsilon_2 \cdot \frac{d^d}{\text{vol}(B(0, 1))}\right) \text{vol}(K^\star). \end{aligned}$$

Thus, there exists at least one set in  $\mathcal{L}$  with this volume approximation to  $K^\star$ , so the minimum volume set that is output must also have this volume approximation to  $K^\star$ .

Finally, we take  $\delta_1 = \delta_2 = \delta/2$ . We take  $\eta_1 = \eta/2$ , and  $\eta_2 = \eta/2^{m_1+1}$ . We take  $\varepsilon = \varepsilon_2 \cdot d^d / \text{vol}(B(0, 1))$ . This tells us that with  $m_1 + 2^{m_1} \cdot m_2$  samples, we get that with probability  $\geq 1 - \eta$ , we output a convex set  $K$  such that  $P(K) \geq (1 - \alpha) - \delta$ , and  $\text{vol}(K) \leq (1 + \varepsilon)\text{vol}(K^\star)$ .

The total sample complexity comes to

$$\begin{aligned} m_1 + 2^{m_1} \cdot m_2 &\leq O\left(\frac{d^2 \log 1/\eta_1}{\delta_1^2(1 - \alpha - \delta_1)}\right) + O\left(\frac{1}{\eta_1 \delta_1^2}\right) \\ &\quad + \exp\left(O\left(\frac{d^2 \log 1/\eta_1}{\delta_1^2(1 - \alpha - \delta_1)}\right) + O\left(\frac{1}{\eta_1 \delta_1^2}\right)\right) \cdot O\left(\frac{d \log 1/\eta_2}{\delta_2^2} \left(\frac{18\pi^{d/2}}{\Gamma(d/2) \cdot \varepsilon_2}\right)^{2d}\right) \\ &\leq \exp\left(O\left(\frac{d^2 \log 1/\eta}{\delta^2(1 - \alpha - \delta)}\right) + O\left(\frac{1}{\eta \delta^2}\right)\right) \cdot O\left(\frac{d}{\delta^2} \left(\frac{18d^d \cdot \Gamma(d/2 + 1)}{\Gamma(d/2) \cdot \varepsilon}\right)^{2d}\right). \end{aligned}$$

□

## Part II

# Solving Related Instances

## Chapter 8

# Introduction to Solving Related Instances

### 8.1 Solving Related Instances of a Problem

Consider a setting where we are repeatedly solving instances of the same task. For example, perhaps we have a routing problem that must be solved every day with different demands.

In such a setting, if the instances are related, we could hope to solve the *sequence* of instances much faster than solving each one from scratch. For example, in our routing problem, perhaps we can learn patterns in the network traffic that allow us to solve instances more efficiently than in the worst case.

In recent years, two related but distinct areas of *algorithms with predictions* and *data-driven algorithms* have emerged to address these kinds of questions. Algorithms with predictions focuses on using unreliable predictions to make algorithms more efficient. Data-driven algo-

rithms designs methods to learn predictions that can help in algorithmic tasks.

In the following [Chapter 9](#) and [Chapter 10](#), we will directly tackle the end goal of solving sequences of related instances. We design broad frameworks that can apply to a wide range of problems and existing algorithms. In addition to building on the techniques of algorithms with predictions and data-driven algorithms, we design new methods that are able to bridge the gap between these two paradigms.

In the following two chapters, we will take two different approaches to solving a series of related instances of a task. These approaches leverage two natural kinds of learned information to achieve speed ups.

In [Chapter 9](#), we consider learning information about the *instances* that will arrive in the future. We consider *dynamic algorithms*, which models sequences of instances that are changing slowly over time. We give a framework based on algorithms with predictions that applies to a broad range of dynamic algorithms, and uses predictions about future changes to the instance to design faster algorithms.

In [Chapter 10](#), we consider learning information about the *solutions* to the instances that will arrive in the future. We consider *arbitrary* sequences of instances, and we will design an *end-to-end* framework that can solve a sequence of instances faster than one at a time, when the sequence is sufficiently related in hindsight. This builds on work in algorithms with predictions and data-driven algorithms to show both that the instances can be solved significantly faster with good predictions, and that these good predictions can indeed be learned. Our framework applies to a broad range of algorithms that admit good “warm start” algorithms with predictions.

In this chapter we provide preliminaries on algorithms with predictions ([Section 8.2](#)) and

data-driven algorithms (Section 8.3). These frameworks have become some of the leading approaches for designing algorithms that use black-box machine learning models. We provide these sections as preliminaries for our results in Chapter 9 and Chapter 10, and to give a introductory overview of these areas and what kinds of guarantees they typically provide. These are not strictly necessary to understand the technical contributions of Chapter 9 and Chapter 10, but serve to place the work in context, and discuss our modeling choices. In Section 8.4, we overview the results Chapter 9 and Chapter 10.

## 8.2 Preliminaries: Algorithms with Predictions

Algorithms with predictions is formalized with the following desiderata.

**Definition 8.2.1** (Algorithms with Predictions). *An algorithm with predictions (or learning-augmented algorithm)  $\mathcal{A}$  for a particular task, takes an instance  $I$  of the task, and a prediction  $P$  of some information.  $\mathcal{A}(I, P)$  always computes a valid solution for instance  $I$ .*

*The performance of  $\mathcal{A}$  simultaneously achieves the following guarantees.*

- (1) Consistency. *When the prediction  $P$  is correct, the performance of  $\mathcal{A}$  is competitive with the best promise algorithm, which was given  $P$  with the promise that the information was perfectly accurate.*
- (2) Robustness. *Even when the prediction  $P$  is arbitrarily inaccurate, the performance of  $\mathcal{A}$  is competitive with the best worst-case algorithm, which ignores  $P$  entirely.*
- (3) Graceful degradation. *The performance of  $\mathcal{A}$  degrades gracefully as a function of the magnitude of the error in the prediction  $P$ , from the performance of the promise algorithm to the performance of the worst-case algorithm.*

*Remark 8.2.2.* The above definition is modified from [Lykouris and Vassilvitskii, 2021], who

defined a version specifically for online algorithms. The desiderata are found in most formalizations of algorithms with predictions in the literature, but the terminology varies. Another common terminology, as in [Lykouris and Vassilvitskii, 2021], denotes these goals as (1) *consistency*, (2) *competitiveness*, and (3) *robustness*.

The above “definition” is underspecified by design— it is a design principle rather than a technical specification. In evaluating whether a particular task is a good candidate for an algorithms with predictions approach, it is important to address the following design choices.

**Defining performance.** Algorithms with predictions has been applied in many algorithmic settings, including designing fast algorithms (“warm starts”), online algorithms, streaming algorithms, mechanism design, and as we will see in Chapter 9, dynamic algorithms, and more. For each of these settings, the measure of performance is different: e.g., runtime, competitive ratio, memory usage, revenue, update time, etc.. Naturally, each notion of performance requires its own toolkit which draws on the corresponding areas of algorithm design.

**Choice of prediction.** For a given task, there are many choices of predicted information that could potentially improve the performance of the algorithm. When designing prediction model, there are many competing objectives to keep in mind.

- (1) *Potential performance gain.* Item 1 in Definition 8.2.1 measures the performance of the algorithm with predictions against the best promise algorithm that has access to the prediction  $P$ . For this baseline to be meaningful, it must be that the prediction  $P$  has the potential to greatly improve the performance of the algorithm over the worst-case algorithm that ignores the prediction entirely.
- (2) *Realistic/learnable predictions.* For an algorithm with predictions to be useful, we must

have access to predictions of good quality. Thus it is important to request information that we could reasonably predict. This is often justified with theoretical learning guarantees, or empirical evidence that high quality predictions of this type are available in practice.

- (3) *Algorithmic tractability*. Finally, the choice of the prediction may itself make it easier or more difficult to design algorithms with predictions.

Typically, the algorithms with predictions literature focuses on giving quantifiable guarantees for the first item (1). That is, they justify that the algorithm with predictions has the potential to significantly outperform a baseline algorithm without predictions.

The second item (2) is often not quantified in the algorithms with predictions literature. The related area of *data-driven algorithms* focuses on addressing this point in a slightly different setting, which we overview in [Section 8.3](#).

**Measuring prediction error.** Item (3) in [Definition 8.2.1](#) requires that the performance of the algorithm with predictions “degrades gracefully” as a function of the magnitude of the error in the prediction  $P$ . This is highly dependent on how the prediction error magnitude is measured. For such guarantees to be meaningful, we require that the error of typical predictions still result in an improvement over the worst-case baseline.

We refer the reader to the book chapter of [Mitzenmacher and Vassilvitskii \[2021\]](#) for an in depth overview of algorithms with predictions, with the caveat that there has been lots of work in this space since the chapter was published. We also refer the reader to this running open source bibliography of papers in this area [[Lindermayr and Megow, 2021](#)].

### 8.3 Data-driven Algorithm Design

*Data-driven algorithm design* is often formalized as follows.

**Problem 8.3.1** (Data-driven algorithm design). *We have a family  $\mathcal{P}$  of algorithms for a certain task. Each algorithm  $\mathcal{A}_\theta \in \mathcal{P}$  is specified by a parameter setting  $\theta \in \Theta$ . On instance  $I$ , we have that for all  $\theta \in \Theta$ ,  $\mathcal{A}_\theta(I)$  always returns a valid solution for  $I$ , and  $\mathcal{A}_\theta$  has performance  $\text{perf}(\mathcal{A}_\theta, I)$  on  $I$ .*

A data-driven algorithm sees some number of sample instances drawn i.i.d. from an unknown distribution  $\mathcal{D}$  over instances, and returns a parameter setting  $\hat{\theta}$  that maximizes performance:

$$\mathbb{E}_{I \sim \mathcal{D}} [\text{perf}(\mathcal{A}_{\hat{\theta}}, I)] = \max_{\theta \in \Theta} \mathbb{E}_{I \sim \mathcal{D}} [\text{perf}(\mathcal{A}_\theta, I)].$$

*Remark 8.3.2.* The above [Problem 8.3.1](#) is stated in terms of a distribution of instances. Another common setting studied in this line of work is the *online* setting, where the algorithm sees a sequence of instances  $I_1, \dots, I_T$ , one by one. On each day  $t$ , the algorithm chooses a parameter setting  $\theta_t$  to apply to  $I_t$ . The algorithm then receives feedback about the performance of  $\mathcal{A}_{\theta_t}$  on  $I_t$ . In the *full information* setting, this includes what the performance of every parameter setting  $\theta \in \Theta$  would have been on  $I_t$ . In the *partial information* or *bandit* setting, this includes only the performance of the chosen  $\theta_t$  on  $I_t$ . The goal, as in online learning, is to achieve total performance that is close to the best fixed parameter setting  $\theta^* \in \Theta$  for the entire sequence in hindsight.

One can think of the parameterized family of algorithms  $\mathcal{P}$  in [Problem 8.3.1](#) as an algorithm with predictions. Whether  $\theta$  is called a “parameter” or a “prediction” is simply a matter of perspective. The difference from algorithms with predictions is the focus of the guarantee. In algorithms with predictions, the goal is to design the parameterized family of algorithms.



In data driven algorithm design, the goal is to learn the right parameter to plug into the parameterized family. Each of these tackles different, important aspects of the problem that need to be addressed to get *end-to-end* guarantees for groups of related instances.

As in the case of algorithms with predictions, this framework is more of a guideline than a precise definition. We discuss some important design considerations when designing data driven algorithms.

**Defining performance.** As in the case of algorithms with predictions, this framework can be applied to problems from many domains that have different notions of performance. The nature of performance, and its dependence on the parameter and input instance can have a large impact on whether a good parameter is learnable.

**Statistical vs. computational efficiency.** Work in the space of data driven algorithms often focuses on providing guarantees that the optimal parameter  $\hat{\theta}$  can be learned *statistically efficiently*, i.e., with only a small number of training instances. There is some work that establish computationally efficient algorithms as well, but this can be challenging to establish.

**Potential improvement via the parameterized family.** An important consideration in data driven algorithm design is that the parameterized family  $\mathcal{P}$  actually contains algorithms that are likely to outperform a baseline worst-case algorithm on the distribution  $\mathcal{D}$  of interest. This captures two phenomena:

- (a) *Expressivity of  $\mathcal{P}$ .* For the algorithms in  $\mathcal{P}$  to outperform a baseline worst-case algorithm that is chosen independently of the input distribution  $\mathcal{D}$ ,  $\mathcal{P}$  must be sufficiently expressive to fit to certain subclasses of instances.

- (b) *Clustering of  $\mathcal{D}$* . For there to exist an algorithm in  $\mathcal{P}$  that does much better for the input distribution  $\mathcal{D}$  than a default worst-case baseline, it must be that  $\mathcal{D}$  is sufficiently “clustered,” so that there is some consistent structure in the instances that an application specific algorithm can take advantage of.

In the literature, this is typically justified empirically, and many works focus on settings like *hyperparameter tuning*, where we know that choosing the right parameters is a significant bottleneck in practice.

We refer the reader to the book chapter of Balcan [2021] for an in depth overview of data driven algorithm design, with the note that there has been much work in this space since that chapter was published.

## 8.4 Overview of Results in Chapters 9 and 10

Both of these chapters tackle the problem of solving sequences of related instances, with different viewpoints. In the first approach, we consider learning information about the *instances* that will arrive in the future. In the second approach, we consider learning information about the *solutions* to the instances that will arrive. We will see that both of these approaches are capable of providing speed ups for sequences of instances, via rather different techniques.

**Dynamic Algorithms with Predictions.** In Chapter 9, we consider learning information about future instances, to speed up the process of solving a sequence of related instances. We tackle this via *dynamic algorithms*. This is a classical paradigm of algorithm design that is designed to capture sequences of related instances, where here “related” means that the instances are changing slowly over time.

For example, consider a task where the instance is a graph. Then a standard *edge inser-*

*tion/deletion* model would consist of a series of instances  $I_1, \dots, I_T$ , where each subsequent pair of graphs  $I_t, I_{t+1}$  differ by one edge. On each day  $t$ , the algorithm is told the edge that differs from  $I_{t-1}$ , and must output the solution to the task on day  $I_t$ . This framework can also be applied to other (not necessarily graph) problems, where the instance consists of a subset of some group set of all possible elements (i.e., in the graph case, edges of the complete graph). Because the instance is constrained to change “slowly,” for many problems we can hope to design algorithms that process the day  $t$  update faster than solving instance  $I_t$  from scratch.

A major bottleneck in the design of dynamic algorithms is an information gap. That is, for many problems, compared to the standard *fully dynamic* setting, we know much more efficient algorithms for the *offline dynamic* setting, where the entire sequence of instances  $I_1, \dots, I_T$  is known in advance, and the algorithm is asked to compute all  $T$  solutions to each day  $t$  in one batch. At a high level, knowing what elements will arrive and leave in the future allows the algorithm to optimize data structures and schedule computations much more efficiently.

This has lead to a wealth of work studying *partially* and *offline dynamic* models where some or all of the future updates are known to the algorithm in advance. Some of these models are practically motivated, like the *incremental* and *decremental* models, which capture cases where elements are only inserted or only deleted, which often show up in practice. Other settings are theoretically motivated, like the *offline dynamic* setting, where the hope is that algorithmic insights from these models can inspire new techniques in the more general *fully dynamic* setting.

In [Chapter 9](#), we show a general way to lift *offline* and *partially dynamic* algorithms to the *fully dynamic* setting when we have access to *unreliable predictions* of future updates. Our

framework is general, and applies to a broad range of problems that have existing offline or partially dynamic algorithms, including including *all-pairs shortest paths*, *k-connectivity*, *depth-first-search trees*, *maxflow/mincut*, *strongly connected components*, *sparsest cut*, and *submodular maximization* (see [Figure 9.1](#) for detailed list). The result is that we get

*faster algorithms for solving dynamic instances, when future updates are predictable.*

We formulate this in the framework of *algorithms with predictions*, and show that our algorithms achieve

- (1) *Consistency*. Recover the *offline* or *partially dynamic* performance when the predictions are perfect.
- (2) *Robustness*. Never do worse than the best known *fully dynamic* algorithm for the task.
- (3) *Graceful Degradation*. Performance of the algorithm smoothly trades off in the magnitude of the error of the prediction.

**Competitive Strategies for “Warm Start” Algorithms with Predictions.** In [Chapter 10](#) we take a different viewpoint on the problem of solving sequences of related instances. In dynamic algorithms, we think of instances as being related if they are very similar in the instance space. In this section we will think of instances as being related if they have very similar *solutions*. This distinction is subtle, but important. Indeed, the main challenge in designing dynamic algorithms is that for many interesting problems, two instances being very similar does *not* mean that their solutions are similar. The converse can also be true. Two very different looking instances can have very similar solutions.

In [Chapter 10](#), we take an *end-to-end* viewpoint. Let  $I_1, \dots, I_T$  be an *arbitrary* sequence of

instances of some task. Our goal is to minimize the total runtime to solve the entire sequence of instances.

Our framework adopts a natural strategy, which is to take an *algorithm with predictions*  $\mathcal{A}(I, P)$  for this task, and use the viewpoint of *data driven algorithms* to learn good predictions  $P_t$  for each instance. Then, the hope is that  $\mathcal{A}$  along with predictions  $P_t$  may allow us to improve on the worst case cost of solving each instance from scratch.

However, there are some new technical challenges that we must address to actually bridge the gap between the data driven algorithms and the algorithms with predictions viewpoints. In particular, for such a strategy to actually gain over the worst case algorithm that solves every instance from scratch, it must be that

*the extra computation spent to learn the predictions is made up in the gain to the algorithm's performance.*

This work, is to the best of our knowledge, the first to establish such a statement, which is a key part of an *end-to-end guarantee*.

To analyze this algorithm, we adopt the viewpoint of *competitive analysis*, and compare to baselines that do well when the sequence of *solutions* to the instances  $I_1, \dots, I_T$  are well-clustered in hindsight (more formally, they admit a good  $k$ -server solution). Thus, when the sequence has structure, the algorithm is able to take advantage of it. Our framework applies to a broad range of algorithmic tasks that admit warm start algorithms with predictions.

[Chapter 10](#) also considers other settings in which we could hope to learn *approximately* optimal predictions to use with *warm start algorithms with predictions*.

## Chapter 9

# Dynamic Algorithms with Predictions

This chapter is based on *The Predicted-Updates Dynamic Model: Offline, Incremental, and Decremental to Fully Dynamic Transformations*, which is joint work with Quanquan C. Liu, and appeared at the Conference on Learning Theory (COLT) 2024 [Liu and Srinivas, 2024].

### 9.1 Introduction

**Dynamic algorithms.** The goal of *dynamic algorithms* is to efficiently compute statistics of an object that is changing over time. For example, the object could be a graph that undergoes a series of  $T$  edge insertions and deletions, which we refer to as happening on  $T$  days. Then, the dynamic algorithm must maintain a statistic of the graph, e.g. a minimum spanning tree, that is correct with respect to the updated graph at all times. The goal of the algorithm is to minimize the total computation, or *work*, it performs to execute these updates.

The baseline for a dynamic algorithm is to perform less work than recomputing the statistic

from scratch on each of the  $T$  days. For a graph problem, recomputing the solution from scratch results in total work of  $\Omega(T \cdot (|E| + |V|))$ , where  $|E| + |V|$  is the size of the graph, since the algorithm must at least read the input graph on each day.

A “good” dynamic algorithm has average update time that is sublinear in the size of the object on each day. For a graph problem, this would be work  $o(T \cdot (|E| + |V|))$ . This certainly beats the baseline, as it is faster than reading the graph each day from scratch.

The “gold standard” for a dynamic algorithm is to reuse computation so well, that the total work done on any day  $t$  is as though the instance on day  $t$  was the only instance the algorithm ever had to solve. If the graph problem itself can be solved in near-linear time, this looks like total work  $O(T \cdot \text{poly log}(|E| + |V|))$ .

In this chapter we will show a general strategy to use *black-box predictions* of future updates in a dynamic problem. The runtime of the algorithm will be a function of the *prediction error*. Even when the predictions are wrong or even misleading, we will always fall back on the “good” guarantee of the best known fully-dynamic algorithm, and achieve total work  $O(T \cdot \text{poly}(|E| + |V|))$ . When the predictions are perfect, or even exhibit moderate error, we will achieve a much better “gold standard” guarantee, with total work  $O(T \cdot \text{poly log}(|E| + |V|))$ . Our strategy is a reduction from our predicted-updates problem to a broad class of *offline dynamic* algorithms, and can be applied to a large range of problems that have had longstanding bottlenecks at polynomial update time in the fully-dynamic (i.e., worst-case dynamic) setting.

The methods we develop will be completely agnostic to the way the predictions are generated. In particular, the performance of our method will depend only on the error of the prediction *in hindsight*. Thus the model that provides the predictions of future updates does *not* need to satisfy any provable guarantees, such as providing *unbiased* predictions, or even

estimating the error of the predictions up front. The predicted-updates dynamic methods will be able to adapt to any degree of prediction error, on the fly.

**Algorithms with predictions.** Algorithms with predictions and traditional dynamic algorithms have fundamentally similar goals: to efficiently provide up-to-date solutions to problems in the face of rapidly changing data. While algorithms with predictions typically use machine learning methods to take advantage of structure in data, traditional dynamic algorithms consider worst-case adversarial inputs, and algorithms are designed to be efficient in spite of such worst-case, potentially unstructured, instances. In this chapter, we combine these two paradigms under a single framework that encompasses the three major settings traditionally studied within dynamic algorithms: the *offline dynamic*, *incremental*, and *decremental* settings. Specifically, we show that we can *transform* broad classes of offline, incremental, and decremental algorithms into *fully dynamic* algorithms provided predictions of the update times of dynamic events such as insertions and/or deletions. We show that we achieve the same runtimes as the original offline, incremental, and decremental algorithms, up to polylogarithmic factors, when the  $\ell_1$  error of our predictions is near-linear in the number of updates that are performed. Not only does this allow us to achieve (sometimes exponential) improvements in runtime for certain problems, but it also allows us to transfer algorithmic techniques between models in novel ways.

This work is, to the best of our knowledge, the first to bring the study of algorithms with predictions to dynamic algorithms, along with independent concurrent work of [van den Brand, Forster, Nazari, and Polak \[2023\]](#), [Henzinger, Saha, Seybold, and Ye \[2023\]](#).

In algorithms with predictions, an algorithm solicits *predictions* from an untrusted source (e.g. a machine learning model) to help make decisions. The goal is to design algorithms that achieve the following three desiderata (see [Section 8.2](#) for a more in depth introduction).



- (1) *Consistency*. If the predictions are of high quality, the algorithm performs much better than a worst-case algorithm.
- (2) *Robustness*. If the predictions are of low quality, the algorithm does not perform any worse than a worst-case algorithm.
- (3) *Graceful Degradation*. The performance of the algorithm degrades gracefully as a function of the prediction error.

Additionally, we want to solicit predictions that can be reasonably obtained in practice. An example is link-prediction for dynamic graphs (where edges are inserted and deleted) which has been gaining great interest recently in the machine learning community, with a set of promising results. A small sample includes [Kumar, Zhang, and Leskovec, 2019, Nguyen, Lee, Rossi, Ahmed, Koh, and Kim, 2018, Poursafaei, Huang, Pelrine, and Rabbany, 2022, Quach, Nguyen, Le, Truong, Duong, Tran, and Luu, 2021, Rossi, Chamberlain, Frasca, Eynard, Monti, and Bronstein, 2020, Zhang and Chen, 2018, Wang, Chang, Liu, Leskovec, and Li, 2021].

These desiderata present a challenge. While we would like to achieve all three for some reasonable notion of prediction, it is not a priori clear that such a guarantee is even possible. For some problems and settings, it is indeed not possible, and algorithms are designed to trade off these objectives. Specifically, in the setting of *online algorithms with predictions*, which is closely related to our work, we see inherent tradeoffs between consistency and robustness (e.g. in the the classic problem of rent-or-buy [Kumar, Purohit, and Svitkina, 2018, Gollapudi and Panigrahi, 2019, Wei and Zhang, 2020]). Such tradeoffs are unavoidable, as online problems often involve an aspect of “commitment.” An algorithm must commit to either following the worst-case strategy or following the prediction. In either case, the adversary can then choose inputs to make the algorithm lose in either consistency or robustness based on this choice.

Thus, the main research direction for online algorithms with predictions is in quantifying these tradeoffs.

Another setting that is closely related to our work is that of *warm starts*, where we design algorithms for static problems that take advantage of a predicted solution. The goal in this setting is to minimize runtime on a static input. Here, it is usually possible to achieve both consistency and robustness, by simply running the learning-augmented procedure in parallel with a worst-case procedure, and outputting the answer of the algorithm which completes first. However, unlike the online setting where we only need predictions of future events, warm starts often require the prediction to be very large [Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2021, Chen, Silwal, Vakilian, and Zhang, 2022, Davies, Moseley, Vassilvitskii, and Wang, 2023], typically encoding a *solution* to the problem. In the context of graph problems, such solutions could have size proportional to the size of the graph, and even reading the prediction can take nontrivial time, much less verifying it.<sup>1</sup>

Thus, it is particularly interesting that for dynamic algorithms, it is indeed possible to achieve the best of online algorithms and warm starts: we only need predictions of future events (instead of future solutions) and we *simultaneously achieve all three desiderata* of algorithms with predictions. Our model has the added benefit that the requested predictions are not problem-specific, and the same predictions could be used multiple times to solve different unrelated problems (e.g. one set of predictions about a dynamic graph can be used to solve a collection of graph problems).

Our framework expands upon the traditional model used for dynamically changing data.

---

<sup>1</sup>As an aside, an interesting question here is how to learn these large predictions in a way that allows us to speed up the process of solving a sequence of related instances of a problem. This is the question that we study in the next chapter [Chapter 10](#). This is very different from the setting of dynamic algorithms, where we hope to achieve *sublinear* update times by taking advantage of the structured way in which the graph changes by only one edge at a time.

The main bottleneck of designing traditional dynamic algorithms is the unknown nature of the update sequence. This is demonstrated by the large gap in efficiency between algorithms for the offline and partially dynamic settings and the fully dynamic (online) setting for many graph problems (see e.g., [Abraham, Chechik, and Krinninger, 2017, Bernstein and Chechik, 2018, Baswana, Chaudhury, Choudhary, and Khan, 2019, Bhattacharya and Kulka-rni, 2020, Chen, Duan, Wang, Zhang, and Zhang, 2018, Chen, Goranci, Henzinger, Peng, and Saranurak, 2020, Das, Gutenberg, and Wulff-Nilsen, 2022, Eppstein, Galil, Italiano, and Nissenzweig, 1997, Forster, Goranci, and Henzinger, 2021, Forster, Nazari, and Gutenberg, 2023b, Goranci, 2019, Goranci, Henzinger, and Saranurak, 2019, Galil, Italiano, and Sarnak, 1999, Gao, Liu, and Peng, 2023, Goldberg and Rao, 1998, Goranci, Räcke, Saranurak, and Tan, 2021, Haeupler, Kavitha, Mathew, Sen, and Tarjan, 2012, Holm and Rotenberg, 2020, Jin and Sun, 2022, Klein and Subramanian, 1998, Peng, Sandlund, and Sleator, 2017]).

To address this, there has been much interest in using the techniques from the offline and partially dynamic settings to obtain fully dynamic algorithms for certain problems (see e.g. [Abraham, Durfee, Koutis, Krinninger, and Peng, 2016, Abboud and Williams, 2014, Bentley, 1979, Baswana, Khurana, and Sarkar, 2012, Bentley and Saxe, 1980, Dobkin and Suri, 1991, Henzinger and King, 1999, Henzinger, Krinninger, and Nanongkai, 2016, Henzinger, Krinninger, Nanongkai, and Saranurak, 2015, Holm, De Lichtenberg, and Thorup, 2001, King, 1999, Kopelowitz, Pettie, and Porat, 2016, Overmars and Van Leeuwen, 2005, Overmars and van Leeuwen, 1979, Probst Gutenberg, Vassilevska Williams, and Wein, 2020, Roditty and Zwick, 2004]).

Following this line of work, Chan [2011] and Peng and Rubinfeld [2023] gave an elegant systematic transformation for *incremental* algorithms to fully dynamic algorithms in the *known-deletion* model. An incremental algorithm is one that produces answers after insertions of elements, while a fully dynamic algorithm is one that produces answers after both

element insertions and deletions. The *known-deletion* model is an intermediate model, between incremental and fully dynamic, in which an inserted element arrives tagged with the precise timestamp of when it will be deleted.<sup>2</sup>

In this chapter, we define the *predicted-updates model* to bring *beyond-worst-case analysis* [Roughgarden, 2021] to the study of dynamic algorithms. In particular, we provide transformations from offline, incremental, and decremental dynamic algorithms to fully-dynamic algorithms in our new model. In the offline dynamic setting, the entire update sequence is provided to the algorithm at once, and the algorithm’s task is to compute all of the solutions corresponding to each day in the sequence. In the incremental setting, an algorithm only needs to handle element insertions and not deletions. Similarly, in the decremental setting, an algorithm only needs to handle element deletions. Our framework can convert an offline dynamic algorithm into a fully-dynamic algorithm using predicted time-stamps for all types of updates. Our framework can also be extended to convert an incremental algorithm into a fully-dynamic algorithm, with only predictions of deletion events, and a decremental algorithm into a fully-dynamic algorithm, with only predictions of insertion events. In particular, our result for incremental algorithms generalizes the aforementioned result of Peng and Rubinfeld [2023].

The techniques in this chapter bring the study of beyond-worst-case analysis for dynamic graph algorithms into focus. Much of the study of dynamic algorithms has, thus far, focused on worst-case analysis with many fundamental problems hitting boundaries in efficiency. Furthermore, state-of-the-art dynamic solutions are often quite complex and require the use of heavy machinery. The framework we introduce in this chapter is fundamentally *simple*, *does not* use heavy machinery, but is applicable to a *broad* range of problems and settings. With the help of predictions, we are able to lift simpler, more implementable, and faster

---

<sup>2</sup>Peng and Rubinfeld [2023] refer to this model as the *deletion lookahead model*.

algorithms from the offline and partially dynamic settings to the fully dynamic setting.

Our work also provides motivation for a new *noisy setting* in offline and partially dynamic algorithms, where the input sequence to the algorithm is drawn from a stochastic model. In particular, our framework gives worst-case guarantees in terms of the  $\ell_1$ -error of the prediction. This already subsumes an interesting stochastic model, where the realized timestamp of each event is drawn from a distribution centered at the predicted time of the event. Similar models have been studied for dynamic data structures by [Lin, Luo, and Woodruff \[2022\]](#), [Cao, Chen, Chen, Lambert, Peng, and Sleator \[2022\]](#).

Finally, our framework gives polynomial, sometimes even exponential, improvements in runtime over the best fully dynamic algorithm for APSP, triconnectivity, dynamic DFS, maxflow/mincut, submodular maximization,  $k$ -edge connectivity, and others (see [Fig. 9.1](#)). Our techniques are also inherently parallel (over small  $\text{poly log } n$  depth) and may have additional implications in scalable models like the work-depth and distributed models.

## 9.2 Technical Overview and Summary of Contributions

First, we define and motivate the *predicted-updates dynamic model*. Then, we introduce and provide a technical overview of our framework to design algorithms for this model. Finally, we describe specific problems for which our framework is able to outperform state-of-the-art fully dynamic algorithms.

### 9.2.1 The Predicted-Updates Dynamic Model

We introduce the *predicted-updates dynamic model*, which is a general model for algorithms with predictions in the dynamic setting that applies to a wide range of problems in the traditional offline, incremental, and decremental dynamic settings. In this chapter, we refer

to *days* to indicate an ordering to the update operations. Throughout, we use *time* and *timestamp* interchangeably with *day*.

Let  $\mathcal{S}$  be the ground set of all possible elements in the dataset. We define an *event* to be an update of a certain type (e.g. insertion, deletion) on an element  $e \in \mathcal{S}$ . We differentiate between *real* and *predicted* events to be updates that actually occur on a day versus an update that is predicted to occur on a day.

**Definition 9.2.1** (Predicted-Updates Dynamic Model). *In the predicted-updates dynamic model, we consider a ground set  $\mathcal{S}$  of size  $|\mathcal{S}|$ .*

*At the outset, the algorithm is given a set  $P$  of predicted updates of the form  $(\text{event}, t)$ , where an event is the specified insertion or deletion of an item  $e \in \mathcal{S}$ , and  $t \in [T]$  is the timestep on which the event is predicted to occur.*

*Then, on each day, exactly one of the following occurs:*

- (a) An element  $e$ , either in  $P$  or not, is inserted,*
- (b) A previously inserted element is deleted, and provides a prediction of when it will be reinserted (if ever).*

*An algorithm computes a function  $f(\cdot)$ , which is a solution to a problem  $\mathcal{P}$ , in the predicted-updates dynamic model, if on every day  $t$ , the algorithm outputs  $f(S)$ , where  $S \subseteq \mathcal{S}$  is the working subset induced by the true (not predicted) sequence of element insertions and deletions that occur in time-steps  $t \in [T]$ .*

*Remark 9.2.2.* It is not strictly necessary that the time-horizon  $T$  is known in advance, or that all of the predictions are given up front. Later we will show that our methods can accomodate predictions arriving in batches as time progresses.

*Remark 9.2.3.* We will also consider the *predicted-deletion* setting, where only the dele-

tion times are predicted, and the insertion times are allowed to be fully arbitrary (Definition 9.4.1). In this setting we will reduce to an *incremental dynamic algorithm*. The same principle will apply to designing a *predicted-insertion* algorithm (Definition 9.5.1) using a *decremental dynamic algorithm*. We defer the formal definition of these settings to the technical sections.

Note that an algorithm in this model *must* compute each day’s output *correctly*. The runtime of the algorithm, however, may depend on the prediction error. We motivate the predicted-updates dynamic model by examining some key features.

**Modularity.** In many settings of the algorithms-with-predictions domain, the requested predictions are problem-specific. This is necessary for some problems, where the purpose of the prediction is to provide a partial solution. However, this requires algorithm designers to provide a custom prediction model, error metric, and algorithm tailored to each new problem. In our model, our requested predictions of element update times are not problem-specific, and are instead applicable to any dynamic problem. Hence, techniques can translate across wide classes of problems, as demonstrated by our framework. We also emphasize that our model makes no distributional assumptions about the input. Rather, our guarantees are in the form of *worst-case analysis* with respect to a new parameter.

**Practical and efficient machine learning.** A main motivation of algorithms-with-predictions is to take advantage of the predictive power of machine learning models. Our model and framework allows us to train a model once e.g., to predict insertion/deletion times of edges in a graph (as a vector of values), and use it many times for a wide range of problems e.g., a collection of dynamic graph problems. Additionally, our framework can handle predictions that are ill-formed, i.e. not feasible update sequences, so we do not have to take into account possibly complicated feasibility requirements in this induced learning

problem. These properties allow us to avoid the costly process of designing and training models that predict problem-specific parameters for each new problem. Additionally, since our requested predictions do not depend on the solutions to a specific problem  $f(\cdot)$ , we *do not* need to compute  $f(\cdot)$  on the dynamic graphs we encounter in the training stage. This could be a very large saving, as computing  $f(\cdot)$  on each training instance, in many cases, takes time superlinear in the size of the instance. Finally, we note that a rapidly growing area of empirical research shows that it is possible to use machine learning to generate high-quality predictions of edge insertions and deletions in graphs [Kumar, Zhang, and Leskovec, 2019, Nguyen, Lee, Rossi, Ahmed, Koh, and Kim, 2018, Poursafaei, Huang, Pelrine, and Rabbany, 2022, Quach, Nguyen, Le, Truong, Duong, Tran, and Luu, 2021, Rossi, Chamberlain, Frasca, Eynard, Monti, and Bronstein, 2020, Wang, Chang, Liu, Leskovec, and Li, 2021, Zhang and Chen, 2018].

**Interpolation.** This model provides a *beyond-worst-case paradigm* for dynamic algorithms that interpolates between the offline and partially dynamic settings (with zero prediction error) and fully dynamic setting (with large prediction error). Beyond-worst-case models can give us insight into what bottlenecks are inherent to a problem and model, and inspire new algorithmic techniques that could transfer between models (see Roughgarden [2021] for an in-depth discussion). We hope that studying the predicted-updates dynamic model can provide insights that inspire better fully dynamic algorithms and lower bounds.

## 9.2.2 Framework for Predicted-Updates Dynamic Algorithms

In conjunction with our model, we design an algorithmic framework that “lifts” offline, incremental, and decremental algorithms to the predicted-updates, fully dynamic setting. In the technical section of the chapter, we use the term *work* to denote *runtime*.

**Theorem 9.2.4** (Predicted-Updates Dynamic Model Framework (Informal version of The-



orem 9.3.13)). Given a balanced, offline divide-and-conquer algorithm,  $\mathcal{A}$ , which performs  $\tilde{O}(|W|^c)$  work per subproblem  $W$  of size  $|W|$ , we can construct an algorithm in the predicted-updates model that, over  $T$  real events, does total work,

$$\begin{aligned} &\tilde{O}(T + \|\text{error}\|_1), \quad \text{when } c \leq 1, \\ &\tilde{O}((T + \|\text{error}\|_1) \cdot T^{c-1}), \quad \text{when } c > 1, \end{aligned}$$

in expectation and with high probability, where  $\|\text{error}\|_1$  is the sum over all elements of the absolute difference between the predicted deletion day and the actual deletion day.

Furthermore, given a “backstop” fully-dynamic algorithm  $B$  for the problem, that does at most  $R_B(T)$  total work by day  $T$ , we can get an algorithm that, over  $T$  updates, does total work

$$\tilde{O}(\min \{(T + \|\text{error}\|_1) \cdot \text{update}(\mathcal{A}), R_B(T)\}),$$

in expectation and with high probability (where  $\text{update}(\mathcal{A})$  depends on  $c$ , see above, in the offline case). (We use  $\tilde{O}(\cdot)$  to hide polylogarithmic factors in  $|\mathcal{S}|$  and  $T$ .)

*Remark 9.2.5.* We also establish the corresponding transformations from incremental dynamic algorithms to predicted-deletion dynamic algorithms (Theorem 9.4.6), and from decremental dynamic algorithms to predicted-insertion dynamic algorithms (Theorem 9.5.2).

Surprisingly, we are able to achieve all three desiderata of algorithms-with-predictions: consistency, competitiveness, and robustness, simultaneously. Note also that this guarantee holds *in hindsight*. That is, the algorithm does not require an estimate of the magnitude of  $\|\text{error}\|_1$  to achieve this goal. Additionally, the potential to include a backstop means that, asymptotically, one can design algorithms that can take advantage of predictions *for free*, without losing the assurance of a worst-case guarantee. We believe that this framework is

practically implementable, and does not rely on any algorithmic “heavy machinery.” This is particularly attractive, as the intention of the model is to provide algorithms that perform well in practice. Furthermore, it is often the case that offline, incremental, and decremental algorithms are simpler than their fully dynamic counterparts, allowing us to utilize these algorithms in real-world settings.

**The  $\ell_1$  Error Metric** The results of [Theorem 9.2.4](#) give runtime bounds that depend on the  $\ell_1$  norm of the prediction error. We use this as a shorthand to denote the sum of absolute differences between the predicted time and realized time of each event,

$$\sum_{\text{events } e} |\text{predictedtime}(e) - \text{truetime}(e)|.$$

One way to interpret this, is that if  $\mathbf{u}$  is a vector indexed by possible events that contains the true update times of each event, and  $\mathbf{p}$  is another vector indexed by possible events that contains the predicted update times of each event, the  $\ell_1$  error of the prediction is  $\|\mathbf{u} - \mathbf{p}\|_1$ . If an event occurs but is never predicted, or vice versa, that event contributes  $T$  to the  $\ell_1$  error. Also, for events that occur multiple times, it suffices to consider the closest matching of duplicated events.

The  $\ell_1$  error metric is a natural one that has been well-studied in applications to warm-starts [[Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2021](#), [Davies, Moseley, Vassilvitskii, and Wang, 2023](#)]. To contextualize the bound, consider predictions that, on average, are within  $\text{poly} \log(T)$  of the true event times. Then, the  $\ell_1$  error will be near-linear in  $T$ , and asymptotically, we achieve the runtime of the offline algorithm (up to  $\text{poly} \log(T)$  factors). On the other hand, the  $\ell_1$  error of any prediction is  $O(T^2)$ . So in the worst case, the algorithm will take runtime equivalent to solving  $T$  independent offline instances (which can be improved by providing the algorithm a worst-case backstop).

*Remark 9.2.6.* Another related error metric is  $T \cdot \|\mathbf{u} - \mathbf{p}\|_\infty$ , where  $\|\mathbf{u} - \mathbf{p}\|_\infty$  is the largest absolute error over all predictions. This bound is always at least the  $\ell_1$  error and is sometimes comparable in magnitude. In general, however, it could be a significantly weaker bound. As an example, the  $\ell_1$  error can be (near-)linear in  $T$  even when there are a polylogarithmic number of events, which occur, that were never predicted. In this same case,  $T \cdot \|\mathbf{u} - \mathbf{p}\|_\infty$  would be  $\Omega(T^2)$ , giving a very different bound. Thus, we focus getting this tighter dependence on  $\ell_1$  error rather than  $T \cdot \ell_\infty$ .

The subsequent work of [McCauley, Moseley, Niaparast, Niaparast, and Singh \[2025\]](#) shows that for the  $T \cdot \ell_\infty$  error metric, the framework that we develop in this work can be simplified, and work with a much broader class offline-dynamic algorithms. In particular, their reduction is deterministic (and thus robust to an adaptive adversary), and does not require the offline-dynamic algorithm to be balanced.

**Random Partition-Tree Framework** Our main technical contribution is the *random partition-tree* data structure ([Definition 9.3.1](#)). This is a divide-and-conquer data structure in the shape of a random tree. We show that this data structure allows us to confine the effects of prediction errors “locally.” Specifically, we present a way to fix the divide-and-conquer data structure on the fly, as prediction errors become apparent. We fix the structure by recomputing nodes (representing recursive subproblems in the divide-and-conquer) in the partition-tree and their descendants. Splitting the recursive subproblems randomly guarantees that the expected amount of recomputation that is triggered by prediction errors is  $\tilde{O}(|t' - t|)$ , where  $t'$  is the predicted time and  $t$  is the real time of the event. Hence, we perform as much work per element as the prediction error,  $|t' - t|$ , associated with that element. This allows our total  $\ell_1$  prediction error to be as large as the number of real updates,  $\tilde{O}(T)$ , while maintaining work matching that of the corresponding offline, incremental, or

decremental algorithm. This can also be seen as low-depth metric embedding for the line, which corresponds to time in our setting.

**Other Technical Contributions of the Framework** In addition to our main contribution, we also solve a variety of other technical challenges listed below.

- (*Early update problem*): an element's true update time is earlier than its predicted update time. That is, we arrive at day  $t$  to see that element  $e$  is updated, but we expected  $e$  not to be updated until some later day  $t' > t$ . This means that some subproblems in our divide-and-conquer recursion tree were based on an incorrect list of events, and are therefore not correct. These subproblems and their descendants need to be recomputed. Our random partition-tree, as described above, allows us to perform the computation in work linearly proportional to the prediction error, with small overhead.
- (*Late update problem*): an element's true update time is later than its predicted update time. That is, we reach a day  $t$  when we expect to see an update to element  $e$ , but element  $e$  is not updated on that day (i.e. there is no real event on element  $e$ ).

We address this by reinserting  $e$ 's update as a prediction for a later time. We do this via a “guess-and-double” procedure. That is, the first time  $e$ 's update is reinserted, we reinsert it for 1 day in the future, then 2 days in the future, then 4 days, etc. Then, we can again fix the data structure to account for this rescheduling using the random partition-tree, like the early deletion problem.

- (*Overscheduling problem*): the predicted event times in  $P$  could schedule arbitrarily many events for one day  $t \in [T]$ . This breaks our divide-and-conquer framework, which distributes work based on the crucial assumption that a small number of events

happens per day. This problem is exacerbated by the repeated rescheduling we do to address the early deletion problem.

We show how to reassign the scheduled element deletions so that not too many are scheduled for one day. We do this by framing our scheduling problem as an instance of *online metric matching* for the line metric (representing time), and using a known competitive algorithm of [Gupta and Lewi \[2012\]](#). Finally, we account for the rescheduling by assigning a batch of  $O(\log T)$  events to each day, and accounting for these in our total work.

- (*Boosting and backstopping for competitiveness*): our work bounds are proven in expectation and our framework may perform worse than state-of-the-art *without backstopping*. We show how to boost our expected work bounds to work bounds with high probability by running  $O(\log(T))$  independent instances of our algorithm.

Furthermore, we show how to “backstop” the algorithm by observing that we can compose two dynamic algorithms  $A$  and  $B$  to get one algorithm that, asymptotically, achieves the minimum amortized runtime of both  $A$  and  $B$ . We compose our framework with a traditional fully dynamic algorithm using our backstop procedure to get a composite algorithm that has runtime comparable to the state-of-the-art.

**Reducing Incremental to Divide-and-Conquer** Using an incremental algorithm, we show that our framework can be used to design algorithms for a stronger setting, in which elements are inserted fully arbitrarily, and our algorithm only receives predictions about deletion times. That is, when an element is inserted, it is tagged with a predicted deletion time. Utilizing our offline framework, we begin by using the incremental algorithm to design an offline dynamic algorithm for this problem. Our previous theorem shows that we can

lift such an algorithm to the setting where we are given predictions for both insertion and deletion events. We then make two main observations that allow us to drop the need for predictions of insertion events.

- In our original offline to fully-dynamic reduction, we prepare a tentative schedule of events during preprocessing. This is no longer possible, because we do not have access to any predicted information before the algorithm starts. However, we achieved this using an *online* algorithm. Thus, we can use the online algorithm to create the schedule, as we learn about new events.
- In our original offline to fully-dynamic reduction, we run a tentative version of the divide-and-conquer algorithm during preprocessing. This is no longer possible, again because we do not have access to the predicted events. We observe that, because of the specific structure of the offline algorithms that arise from incremental algorithms, we can run the computations associated with the nodes of the divide-and-conquer tree using a *just-in-time* approach.

This extension of our framework is described in detail in [Section 9.4](#).

**Reducing Decremental to Divide-and-Conquer** We also apply our framework to lifting decremental algorithms to the fully-dynamic setting, given predictions of insertion events. We do this via a simple reduction to the incremental case. In particular, we reinterpret a decremental algorithm over elements as an incremental algorithm over “anti-elements.” Then, we are able to directly apply the previous result. This is described in detail in [Section 9.5](#).

### 9.2.3 Applications

Our framework transforms a variety of incremental algorithms into the predicted-updates, fully dynamic model to obtain guarantees (often) much better than their fully dynamic counterpart. Our results are shown in [Figure 9.1](#).

| Problem                       | Best Fully Dynamic Runtimes                                                                           |                                                                                 | New Predicted-Update Runtimes<br>(Theorems 9.6.4 to 9.6.6)                              |                                                                                   |
|-------------------------------|-------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Planar Digraph APSP           | $\tilde{O}(n^{2/3})$                                                                                  | <a href="#">Fakcharoenphol and Rao [2006]</a> ,<br><a href="#">Klein [2005]</a> | $\tilde{O}(\sqrt{n})$                                                                   | <a href="#">Das et al. [2022]</a>                                                 |
| Triconnectivity               | $O(n^{2/3})$                                                                                          | <a href="#">Galil et al. [1999]</a>                                             | $\tilde{O}(1)$                                                                          | <a href="#">Holm and Rotenberg [2020]</a> ,<br><a href="#">Peng et al. [2017]</a> |
| $k$ -Edge Connectivity        | $n^{o(1)}$                                                                                            | <a href="#">Jin and Sun [2022]</a>                                              | $\tilde{O}(1)$                                                                          | <a href="#">Chalermsook et al. [2021]</a>                                         |
| Dynamic DFS Tree              | $\tilde{O}(\sqrt{mn})$                                                                                | <a href="#">Baswana et al. [2019]</a>                                           | $\tilde{O}(n)$                                                                          | <a href="#">Baswana et al. [2019]</a> ,<br><a href="#">Chen et al. [2018]</a>     |
| Minimum Spanning Forest       | $\tilde{O}(1)$                                                                                        | <a href="#">Holm et al. [2001]</a>                                              | $\tilde{O}(1)$                                                                          | <a href="#">Eppstein [1994]</a>                                                   |
| APSP                          | $\left(\frac{256}{k^2}\right)^{4/k}$ -Approx<br>$\tilde{O}(n^k)$ update<br>$\tilde{O}(n^{k/8})$ query | <a href="#">Forster et al. [2023a]</a>                                          | $(2r-1)^k$ -Approx<br>$\tilde{O}(m^{1/(k+1)}n^{k/r})$                                   | <a href="#">Chen et al. [2020]</a>                                                |
| AP Maxflow/Mincut             | $O(\log(n) \log \log n)$ -Approx<br>$\tilde{O}(n^{2/3+o(1)})$                                         | <a href="#">Chen et al. [2020]</a>                                              | $O(\log^{sk}(n))$ -Approx.<br>$\tilde{O}(n^{2/(k+1)})$                                  | <a href="#">Goranci [2019]</a> ,<br><a href="#">Goranci et al. [2019]</a>         |
| MCF                           | $(1+\varepsilon)$ -Approx<br>$\tilde{O}(1)$ update<br>$\tilde{O}(n)$ query                            | <a href="#">Chen et al. [2020]</a>                                              | $O(\log^{sk}(n))$ -Approx.<br>$\tilde{O}(n^{2/(k+1)})$ update<br>$\tilde{O}(P^2)$ query | <a href="#">Goranci [2019]</a> ,<br><a href="#">Goranci et al. [2019]</a>         |
| Strongly Connected Components | $\Omega(m^{1-\varepsilon})$ query or update                                                           | <a href="#">Abboud and Williams [2014]</a>                                      | $\tilde{O}(m)$                                                                          | <a href="#">Roditty [2013]</a>                                                    |
| Uniform Sparsest Cut          | $2^{O(\log^{5/6}(n))}$ -Approx<br>$2^{O(\log^{5/6}(n))}$ update<br>$O(\log^{1/6}(n))$ query           | <a href="#">Goranci et al. [2021]</a>                                           | $O(\log^{sk}(n))$ -Approx<br>$\tilde{O}(n^{2/(k+1)})$<br>$O(1)$ query                   | <a href="#">Goranci [2019]</a> ,<br><a href="#">Goranci et al. [2019]</a>         |
| Submodular Max                | $1/4$ -Approx<br>$\tilde{O}(k^2)$                                                                     | <a href="#">Dütting et al. [2023]</a>                                           | $0.3178$ -Approx<br>$\tilde{O}(\text{poly}(k))$                                         | <a href="#">Feldman et al. [2022]</a>                                             |

Figure 9.1: Table of the best fully dynamic update runtimes for a variety of problems vs. our update times obtained via our framework in the predicted-deletion model assuming  $\|\text{error}\|_1 = \tilde{O}(T)$ . Our query times match the fully dynamic query times for every problem. The acronyms are as follows: APSP: all-pairs shortest paths, DFS: depth-first search, AP: all-pairs, MCF: multi-commodity concurrent flows. The variable  $P$  is the number of queried pairs in the multi-commodity flow result. If only one runtime is shown, the same runtime holds for both update and query. Problem definitions are provided in [Section 9.6](#). Here,  $m$  is the maximum number of edges in the graph at any time. The submodular maximization problem's runtime is measured in terms of the number of function evaluations and  $k$  is the rank of the matroid constraint. Some of our results match the state-of-the-art, although, they are potentially simpler to implement than the state-of-the-art so we present them for completeness.

### 9.2.4 Comparison with Concurrent, Independent Work

Concurrent work of [van den Brand, Forster, Nazari, and Polak \[2023\]](#) and [Henzinger, Saha, Seybold, and Ye \[2023\]](#) also study algorithms in dynamic graph models with prediction. [Henzinger, Saha, Seybold, and Ye \[2023\]](#) focus on lower bounds in their paper under different prediction models from our work. [van den Brand, Forster, Nazari, and Polak \[2023\]](#) also study different types of prediction models including a model very similar to our predicted-deletions model, where instead of using the  $\ell_1$ -error of a prediction, they instead look at the number of element-wise inversions between the predicted *deletion* sequence and the real sequence. They give a deterministic algorithm for incremental algorithms with predicted deletions based on [Peng and Rubinstein \[2023\]](#). Their reduction maintains the state of an incremental algorithm, in which elements are inserted in approximately reverse deletion order. In this chapter, we give a reduction from the decremental setting with predicted *insertions* to the incremental setting that can also be applied to their result. The reduction applied to their construction can be interpreted as doing essentially the reverse: it maintains the state of a decremental algorithm, in which elements are deleted in approximately reverse insertion order.

Their framework does not handle offline to fully dynamic transformations for two reasons:

- (1) It is unclear how to simultaneously insert elements in both approximately reverse deletion order *and* reverse insertion. order when given noisy predictions
- (2) Their reduction is tied to incremental algorithms and it is even unclear what types of algorithms to reduce to in the offline setting.

We sidestep both issues in our framework with the key data structure used in our solution: the random *partition-tree* ([Definition 9.3.1](#)). A more detailed comparison of the approaches



is provided in [Section 9.4.1](#)

Independent and concurrent work of [Agarwal and Balkanski \[2024\]](#) studies the problem of *dynamic submodular function maximization with cardinality constraints* in different prediction model than the one we consider. This form of update time is incomparable to runtimes in the form of our result. They also study a different form of submodular maximization than we do in this work, we consider matroid constraints as opposed to cardinality.

## 9.3 Offline Divide-and-Conquer to Fully-Dynamic Transformation

### 9.3.1 Preliminaries

We denote the *total* number of updates in our update sequence by  $T$ . We use  $[t]$  to denote the set of positive integers up to  $t$ , ie.  $\{1, \dots, t\}$ . We use  $\tilde{O}(\cdot)$  to hide  $\text{poly}(\log(|\mathcal{S}| \cdot T))$  factors where  $\mathcal{S}$  is our ground set for the elements and  $T$  is the total number of updates. Often, it holds that  $|\mathcal{S}| = \text{poly}(T)$ . Our explicit bounds are presented in the formal theorem statements.

Below, we first define the partition-tree (which is generated randomly in our algorithm), the main data structure in our framework.

**Definition 9.3.1** (Partition-Tree). *A partition-tree of a sequence of days is a binary tree such that every node in the tree is associated with a window (interval of time), with the following properties:*

1. *The root window of the tree is the full sequence of  $T$  days.*
2. *For each internal (parent) window  $W$ , its children windows partition  $W$ . We refer to*

the parent of window  $W$  as  $\text{parent}(W)$ .

3. Each leaf window is associated with a single day.

We will refer to the size of a window associated with a node in the partition-tree as the size of the node. Henceforth, we will refer to all nodes in the partition-tree as **windows**.

Our framework works for any *c-divide-and-conquer* algorithm, defined below. For our offline to fully dynamic transformations, we show a number of interesting algorithms fall under this definition. For our incremental and decremental to fully dynamic transformations, we show that *any* incremental or decremental algorithm with *worst-case* update time can be framed as a *c-divide-and-conquer* algorithm and can, hence, be used in our framework.

**Definition 9.3.2** (Divide-and-conquer algorithm). *Given a partition-tree  $T$ , a  $c$ -divide-and-conquer algorithm  $\mathcal{A}$ , for some constant  $c > 0$ , computes a function  $f(\cdot)$  (to solve a problem  $\mathcal{P}$ ) over subsets of the ground set  $\mathcal{S}$  (generally,  $|\mathcal{S}| = \text{poly}(T)$ ), such that:*

1. A polynomial amount of read-write memory  $\mathbf{M}$  is given to the root window;
2. Algorithm  $\mathcal{A}$  associates a computation with each window  $W$  of  $T$ , where the computation is given only the unordered set of events occurring in  $W$ , and read-write access to  $\mathbf{M}_{\text{parent}(W)}$  which is the state of  $\mathbf{M}$  after  $\text{parent}(W)$  has terminated; Notably,  $\mathbf{M}$  is passed down to  $W$  on a path from the root, and is not mutated by any sibling windows;
3. The expected work done at window  $W$  is  $O(\Gamma \cdot |W|^c)$ , where  $\Gamma$  is constant in  $|W|$  and usually represents  $O(\text{poly log}(|\mathcal{S}| \cdot T))$  factors or  $\text{update}(\mathcal{A})$  where  $\mathcal{A}$  is a worst-case incremental or decremental algorithm.<sup>3</sup> For simplicity of expression in our proofs,<sup>4</sup> we

---

<sup>3</sup> $\text{update}(\mathcal{A})$  is the worst-case update time of an incremental or decremental algorithm.

<sup>4</sup>We omit the factor since  $\Gamma$  shows up repeatedly in all of our expressions, and this expression would fall under our  $\tilde{O}(\cdot)$  bound for the offline transformations and is explicitly given for our incremental/decremental transformations.

*sometimes omit  $\Gamma$  in our calculations.*

4. *The solution to  $\mathcal{P}$  is obtained from the windows in the tree and the memory,  $\mathbf{M}$ , within each window.*

The requirement in [Item 3](#) ensures that the work done by the divide-and-conquer algorithm is evenly distributed across the tree. This is the analogue of our condition for the incremental/decremental to fully-dynamic setting, where we require the incremental algorithm to have a worst-case (not amortized) bound.<sup>5</sup>

### 9.3.2 Algorithm Description

We obtain two separate categories of inputs to the algorithm: offline inputs received during preprocessing, and online inputs. [Algorithm 9.1](#) uses the partition-tree  $\mathcal{T}$ , the divide-and-conquer algorithm  $\mathcal{A}$  that computes  $f(\cdot)$ , and the predicted sequence of dynamic updates  $P$ . The predicted sequence is given as tuples  $(e, \text{type}, t, i)$  where  $e$  is the element, type is the type of update (an insertion or deletion),  $t$  is the predicted day of the update, and  $i$  is the number of times the event has been rescheduled. Initially, all  $i = 0$  for all events in the prediction sequence. Then, the actual set of updates is given as an online sequence of dynamic updates  $U$ . The algorithm outputs an answer to the function  $f(\cdot)$  after each update on each day  $t \in [T]$  where  $U_t$  consists of the set of updates that occurred on days  $[1, \dots, t] \subseteq [T]$ .

The raw update predictions  $P$  may be infeasible: it may have multiple events scheduled for the same day, or deletion events predicted to happen before their corresponding insertion events. The first step of our algorithm is to convert  $P$  into a feasible sequence of events in

---

<sup>5</sup>This requirement was lifted by the subsequent work of [McCauley, Moseley, Niaparast, Niaparast, and Singh \[2025\]](#), at the expense of a bound that depends on the  $T \cdot \ell_\infty$  error metric. See [Remark 9.2.6](#) for discussion.

our model, with error comparable to the original set of update predictions. In particular, the error of our converted sequence of predictions has error at most  $O(\log(T))$  factor worse than the original set  $P$ . In [Algorithm 9.1](#), we do this preprocessing. We include the pseudocode for `PREPROCESSPREDICTIONS( $P$ )` in [Section A.1](#).

To do this, we first convert  $P$  into an instance of metric online bipartite matching where  $P$  represents the requests, and the set of days  $t \in [T]$  are our servers. We would like to produce a matching such that at most one event in  $P$  is matched to each  $t \in [T]$ . We use the harmonic algorithm of [Gupta and Lewi \[2012\]](#), which iteratively matches each request to its nearest open server on the right or left, with harmonic probabilities. We can run the iterations efficiently using a union-find data structure to produce a matching via the online bipartite matching procedure. Then, finally, we do a post-processing on the produced matching. In the order of the days, we perform a linear scan to check if any deletions of events occur before their respective insertions. For any such deletions, we move the event to the same day as the insertion. We can do this in  $O(T)$  time by maintaining hash maps keyed by the events. After doing this post-processing, we obtain our feasible sequence of events  $\bar{P}$  which consists of at most two events per day: at most one insertion and at most one deletion. Then, we insert each predicted event in  $\bar{P}$  into their corresponding predicted days. This procedure can also be applied to any constant number of event types with precedence constraints (i.e. one event type must be performed before another).

[Algorithm 9.1](#) iterates through each day  $t \in [T]$  and checks for each *real* and *predicted* event assigned to day  $t$ . A *real* event is an event that is assigned to day  $t$  by the update sequence. A *predicted* event is one that assigned by our algorithm (i.e. an event that is predicted to occur on day  $t$ ). For each real event of type  $T$  and on element  $e$ , we find the corresponding prediction for this event in  $\bar{P}$ . If this event is not in  $\bar{P}$ , then we assign the default prediction of  $(e, T, \infty, 0)$  where  $\infty$  indicates that we predict the event to occur at the

very end of the update stream. If our current day  $t$  is less than the predicted day  $t_{\text{predict}}$ , then we process the event as one that occurs *earlier* than predicted. We use the procedure `PROCESSEVENTEARLIERTHANPREDICTION`( $E, t, t_{\text{predict}}$ ), explained below.

After iterating through the real events, we now iterate through the *predicted* events on day  $t$ . If the predicted event  $E = (e, T)$  was not a real event that occurred on day  $t$ , then, we update our prediction since the real event will occur on a later day. To update our prediction, we call the procedure, `PROCESSEVENTLATERTHANPREDICTION`, for handling events that occur *later* than the predicted day. Finally, we call the function `OUTPUT` that is obtained from  $\mathcal{A}$  for computing the output of our dynamic algorithm at  $t$ .

We now describe each of our individual procedures that is called within our main algorithm:

- **Retrigger**( $t_1, t_2$ ): This procedure retriggers the processing of all events in *every descendant* of the smallest window  $W$  in  $\mathcal{T}$  that contains both  $t_1$  and  $t_2$ . The processing of the events uses the procedure `PROCESSEVENTS` that is obtained from  $\mathcal{A}$ .
- **ProcessEventEarlierThanPrediction**( $E, t, t_{\text{predict}}$ ): This procedure processes a real event that occurs on a day  $t$  prior to its predicted day  $t_{\text{predict}}$ . First, we delete the predicted event  $E$  from day  $t_{\text{predict}}$ . Then, we add  $E$  to the day  $t$  as a real event. Finally, we find the smallest window  $W$  in  $\mathcal{T}$  that contains both  $t$  and  $t_{\text{predict}}$  and retrigger  $W$ .
- **ProcessEventLaterThanPrediction**( $E, i, t$ ): This procedure reschedules events which occur later than the prediction; specifically, the event is rescheduled to a future day. We first remove the predicted event  $E$  from day  $t$  ([Algorithm 9.1](#)). We then increase the predicted day for the corresponding event from  $t$  to  $t + 2^i$  where  $i$  the number of times it has been rescheduled before. Then, we find the smallest window  $W$  that contains both  $t$  and  $(t + 2^i)$  and change the corresponding prediction to reflect the new

prediction. We now need to manage settings where there are multiple possible updates associated with an element and ordering constraints among the types of updates on the elements. In the case of edge insertions and deletions, the insertion for an edge must happen before the deletion of the edge (such is an example of an ordering constraint). Thus, we look for all corresponding predicted deletion events that occurs on an earlier day than  $(t + 2^i)$  and reschedule them to day  $(t + 2^i)$ . Finally, retrigger on  $W$ .

### 9.3.3 Preprocessing and Scheduling of Predictions

Our preprocessing step in [Algorithm 9.1](#) performs two main functions. The first is to create an initial schedule of events. The second is to compute an initial set of solutions for these scheduled events using a divide-and-conquer algorithm. Both the schedule and the solutions will be updated throughout the run of the algorithm.

We begin by rescheduling the input predictions to ensure feasibility. That is, the input predictions could be malformed—either by having many events predicted for a single day, or by having deletion events predicted to be before insertion events. Having too many predicted events on a single day leads to issues in showing our expected work bounds using our partition-tree analysis (discussed later). Thus, we preprocess these predictions to form a feasible sequence of events, that will be a well-formed input to our divide-and-conquer algorithm. We show that it is possible to do this in near-linear time, while only blowing up the  $\ell_1$  error of the predictions by a polylogarithmic factor.

The ability to handle ill-formed predictions is a key functionality of our algorithm. This allows the associated learning problem to be the simple problem of learning a vector that is close to our target prediction vector in  $\ell_1$  distance, without having to account for potentially complicated feasibility constraints.

We can formulate our predicted sequence of dynamic updates to a prediction vector  $\mathbf{p}$  by assigning each event two unique positions in  $\mathbf{p}$ , one for insertions and one for deletions. Then, in the corresponding position, we predicted day in  $[T]$  that the event occurs. Then, our preprocessing produces a vector  $\mathbf{a}_0$  where each day contains at most two events (at most one insertion and one deletion). We can also create a vector  $\mathbf{r}$  of the real events that occur online. For each real event that was not predicted in  $P$ , we assign the event positions at the end of the vectors  $\mathbf{a}_0$  and  $\mathbf{r}$  and give them all a predicted day of  $\max(\mathbf{r}) + 1$ , which means that we predict the event to occur at the very end, after the last real event. Note that we generate these vectors *solely* for the purpose of computing the  $\ell_1$  error. Our preprocessing procedure only needs to output a sequence  $\overline{P}$  consisting of tuples where predictions in  $P$  are assigned to days such that at most two events occur on any day.

**Lemma 9.3.3** (Initial Scheduling Quality and Runtime). *Let  $\mathbf{p}$  be the vector of predictions, mapping each event to a day in  $[T]$ . Let  $\mathbf{r}$  be the (unknown to the algorithm) true vector of real events, mapping each event to the day in  $[T]$  that it actually occurs.*

*Given  $\mathbf{p}$ , we can compute an assignment vector  $\mathbf{a}_0$  such that  $\mathbf{a}_0$  assigns at most one insertion event and at most one deletion event to each day, assigns all insertions events before deletion events, and has error*

$$\mathbf{E} [||\mathbf{a}_0 - \mathbf{r}||_1] = O (||\mathbf{p} - \mathbf{r}||_1 \cdot \log(T)).$$

*Furthermore, we can compute this in time  $O(T \log^*(T))$ .*

The formal proof of this lemma is given in [Section A.1](#). At a high level, the problem of reallocating the events so that there is at most one on each day is an instance of bipartite matching over the line metric. We wish to solve this instance in time near-linear in  $T$ , otherwise this preprocessing could dominate the total runtime of the algorithm. We observe that, interestingly, the online algorithm of [Gupta and Lewi \[2012\]](#) can be used as a fast

$O(\log T)$ -approximation algorithm for this offline problem.

For predictions where deletions are listed before their corresponding insertions, we perform one additional pass of  $\mathbf{p}$  and reassign all deletion events which occur before their corresponding insertion event to the day the corresponding insertion event occurs. This produces an assignment where at most two events are assigned to the same day and requires only  $O(T)$  additional work.

After we obtain a feasible sequence of predictions,  $\bar{P}$ , we now construct our partition-tree over our  $c$ -divide-and-conquer algorithm  $\mathcal{A}$  (Definition 9.3.2). We now compute the work of constructing our partition-tree,  $\mathcal{T}$ , using  $\bar{P}$  and prove the depth of the constructed tree. We first prove the work to compute the partition-tree.

**Lemma 9.3.4** (Work to Compute Initial Divide-and-Conquer Solutions). *The full divide-and-conquer algorithm over the partition-tree can be computed in expected time*

$$O(T^c \cdot \log^3(T) \cdot \log \log(T \cdot |\mathcal{S}|)) \quad \text{when } c > 1$$

$$O(T \cdot \log^3(T) \cdot \log \log(T \cdot |\mathcal{S}|)) \quad \text{when } c \leq 1.$$

*Proof.* This is a special case of Lemma 9.3.9, with  $\text{RETRIGGER}(1, T)$ . □

We now bound the depth of the partition-tree over the predictions in  $P$ . Bounding the depth is necessary in order to minimize the work of performing recomputation when the real update sequence deviates from the predictions. As the below analysis draws inspiration from the analysis of randomized quicksort, we relegate its proof to Section A.1.

**Lemma 9.3.5** (Partition-Tree has  $O(\log(T))$  Depth). *The depth of a random partition-tree*



drawn over  $T$  days is  $O(\log(T))$  in expectation. Furthermore, for any constant  $k \geq 36$ , the depth is

$$\leq k \ln(T) \quad \text{with probability} \quad \geq 1 - T^{-\frac{k}{24}}.$$

Note that we do not need knowledge of  $T$  when computing the aforementioned bounds. We show in [Algorithm 9.4](#) how to remove the assumption on the value of  $T$ .

Finally, we give the work of maintaining the schedule after predictions are rescheduled based on the online dynamic sequence of updates. Predictions are rescheduled using `PROCESSEVENTEARLIERTHANPREDICTION` and `PROCESSEVENTLATERTHANPREDICTION`. Below (in [Lemma 9.3.6](#) with proof deferred to [Section A.1](#)) we show the maximum total number of times all predicted events are rescheduled by [Algorithm 9.1](#) which combined with the work necessary to call `RETRIGGER` which we analyze in [Sections 9.3.4](#) and [9.3.5](#) gives the total work of our algorithm.

**Lemma 9.3.6** (Work to Maintain Schedule). *[Algorithm 9.1](#) performs at most  $O(T \log T)$  reschedules to the predictions of updates and maintain the predicted schedule of events over the course of the online dynamic updates.*

### 9.3.4 Work of a Single Call to Retrigger

The main workhorse of our algorithm is the `RETRIGGER`( $t_1, t_2$ ) operation, which recomputes the subtree of the partition-tree rooted at the smallest window  $W$  that contains  $t_1$  and  $t_2$ . This operation recomputes the part of the partition-tree that can be affected by rearranging events that are scheduled between  $t_1$  and  $t_2$ . We showed above that because the partition-tree is drawn randomly, we can bound the depth of the tree with high probability. Using what we proved above, in this section, we show the additional properties that:

- The expected work to recompute the subtree associated with a  $\text{RETRIGGER}(t_1, t_2)$  call scales linearly with  $|t_2 - t_1|$ .
- The work of the divide-and-conquer algorithm is balanced over calls to  $\text{RETRIGGER}$ .
- Hence, combined with the fact that our tree has bounded  $O(\log T)$  depth, with high probability, our running time scales *linearly* with the  $\ell_1$  error.

We first show that the expected work to recompute the subtree associated with a  $\text{RETRIGGER}(t_1, t_2)$  call scales with  $|t_2 - t_1|$ . To do this, we first prove the expected size of the smallest window that contains any pair of days  $t_1$  and  $t_2$ . Such a lemma allows us to bound the amount of computation we need to perform to recompute the events between any two  $t_1, t_2 \in [T]$ .

*Remark 9.3.7.* The below lemma essentially establishes that the random-partition tree is a good metric embedding for the line. In fact, we could have used any distribution on trees that is low-depth and a good metric embedding for the random-partition tree, including a balanced tree with a random offset, or the randomized tree-embedding scheme of [Fakcharoenphol, Rao, and Talwar \[2004\]](#).

**Lemma 9.3.8** (Random Partition-Tree Preserves Lengths in Expectation). *Consider a random partition-tree drawn according to [Definition 9.3.1](#) for time sequence  $[T]$ . Then, for any  $t_1, t_2 \in [T]$  where  $t_1 \neq t_2$ , the expected size of the smallest window that strictly contains both  $t_1$  and  $t_2$  is  $O(|t_2 - t_1| \cdot \log(T))$ .*

*Proof.* We prove this lemma via a coupling argument where we show another way to generate the same distribution of binary tree partitions that will be easier to analyze. For each possible divider  $d \in [T - 1]$  of the original sequence, we associate  $d$  with a rank  $r_d$  drawn uniformly at random from the uniform distribution over  $[0, 1]$ . The ranks are drawn independently for

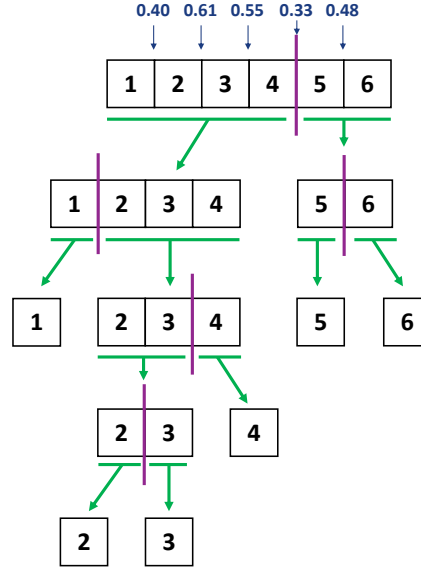


Figure 9.2: Dividers are drawn uniformly at random from  $[0, 1]$  and shown in blue above the figure. This example contains 5 dividers since  $T = 6$ .  $[2, 4]$  is a window in the tree because  $0.40, 0.33 < 0.61, 0.55$ .  $[2, 5]$  is not a window in the tree because  $0.48 \not< 0.33$ .

each  $d$ .

Using the ranks, we assign the tree structure from the top down. At the top level, the divider with the lowest rank is used to split the sequence into the left child and the right child. Iteratively, for each new window of the tree, we use the lowest ranked divider of its subsequence to split the sequence. This results in the same distribution over partition-trees as [Definition 9.3.1](#), as at each level, each of the possible dividers is equally likely to be chosen next.

Now, we see that a contiguous subsequence  $[t_{\text{start}}, t_{\text{end}}]$  of  $[T]$  is a window in the partition-tree if and only if the divider directly preceding  $t_{\text{start}}$  and the divider directly following  $t_{\text{end}}$  both have lower ranks than all of the dividers between  $t_{\text{start}}$  and  $t_{\text{end}}$ , where we can consider the endpoints of the original sequence over all days in  $[T]$  to be dividers with rank 0. This is illustrated in [Figure 9.2](#).

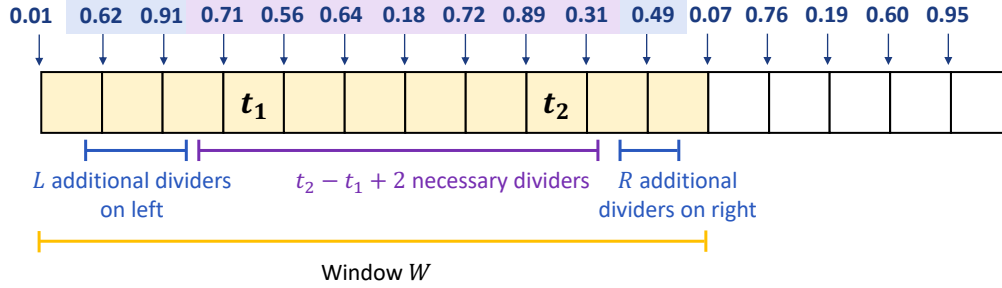


Figure 9.3:  $W$  is the smallest window containing  $[t_1, t_2]$ . The random variables  $L$  and  $R$  capture the number of additional days added to  $[t_1, t_2]$  to satisfy the property illustrated in Fig. 9.2. The dividers bordering  $W$  must have smaller value than all dividers in  $W$ . Furthermore, the dividers in  $L$  and  $R$  have *larger* values than the dividers bordering  $[t_1, t_2]$  and the dividers contained within  $[t_1, t_2]$ ; otherwise,  $W$  would not be the smallest window containing  $[t_1, t_2]$ .

Without loss of generality let  $t_1 < t_2$ . Let  $W$  be the smallest window that strictly contains  $t_1$  and  $t_2$ . We have that the size of  $W$  (in days) is

$$|W| = (t_2 - t_1 + 1) + L + R,$$

where  $t_2 - t_1 + 1$  are the number days between the dividers bordering  $t_1$  and  $t_2$ .  $L$  is a random variable representing the number of days before  $t_1$  until we reach one that is bordering a divider with strictly smaller rank than the  $t_2 - t_1 + 2$  dividers drawn between  $t_1$  and  $t_2$  and the dividers bordering  $t_1$  and  $t_2$ . Symmetrically,  $R$  is the random variable for days after  $t_2$  until we reach a divider with smaller rank. This is illustrated in Figure 9.3.  $L$  and  $R$  must each contain dividers with ranks *larger* than the dividers between  $t_1$  and  $t_2$ ; otherwise,  $W$  would not be the smallest window strictly containing  $t_1$ ,  $t_2$ , and all days in between.

First, we analyze  $\mathbf{E}[L]$ . Let

$$Z = \min\{r_d : d \in [t_1 - 1, t_2 + 1]\}$$

be the minimum rank of the dividers between  $t_1$  to  $t_2$  and including the ones bordering  $t_1$  and  $t_2$ . We can compute  $\mathbf{E}[L]$  by conditioning on different values of  $Z$ . Let  $p_Z(\cdot)$  be the p.d.f. of  $Z$ .

$$\mathbf{E}[L] = \int_{z=0}^1 \mathbf{E}[L|Z = z] \cdot p_Z(z) dz \quad (9.1)$$

$$\leq \int_{z=0}^1 \min \left\{ T, \frac{1}{z} \right\} p_Z(z) dz \quad (9.2)$$

$$\leq \int_{z=0}^{1/T} T dz + \int_{z=1/T}^1 \frac{1}{z} \cdot p_Z(z) dz \quad (9.3)$$

$$\leq 1 + \int_{z=1/T}^1 \frac{1}{z} \cdot p_Z(z) dz \quad (9.4)$$

Eq. (9.2) follows because at most  $T$  items can be added to the window before we run out of items in the sequence to add; furthermore, the  $\frac{1}{z}$  term is the mean of the geometric distribution with parameter  $z$  since the probability that we draw a rank smaller than  $z$  is at most  $z$ . We can simplify our equation in Eq. (9.3) to Eq. (9.4) since  $\int_{z=0}^{1/T} T dz = 1$ .

Now, we can find the p.d.f. of  $Z$  via its c.d.f. Let  $F_Z(\cdot)$  be the c.d.f. of  $Z$ .

$$F_Z(z) = \mathbf{P}[Z < z] = 1 - (1 - z)^{t_2 - t_1 + 2}$$

$$\begin{aligned} p_Z(z) &= \frac{d}{dz} (F_Z(z)) \\ &= (t_2 - t_1 + 2)(1 - z)^{t_2 - t_1 + 1}. \end{aligned}$$

This allows us to conclude the following since  $\int_{z=1/T}^1 \frac{1}{z} dz = \ln(T)$ :

$$\begin{aligned}
\mathbf{E}[L] &\leq 1 + \int_{z=1/T}^1 \frac{(t_2 - t_1 + 2)(1 - z)^{t_2 - t_1 + 1}}{z} dz \\
&\leq 1 + (t_2 - t_1 + 2) \int_{z=1/T}^1 \frac{1}{z} dz && \text{since } (1 - z)^{t_2 - t_1 + 1} \leq 1 \\
&= 1 + (t_2 - t_1 + 2) \ln(T).
\end{aligned}$$

Symmetrically, by the same argument,  $\mathbf{E}[R] \leq 1 + \ln(T)$  as well. So by linearity of expectation, we have

$$\begin{aligned}
\mathbf{E}[W] &= (t_2 - t_1 + 1) + \mathbf{E}[L] + \mathbf{E}[R] \\
&\leq (t_2 - t_1 + 1) + 2 \cdot (t_2 - t_1 + 2) \cdot \ln(T) \\
&= O(|t_2 - t_1| \cdot \log(T)).
\end{aligned}$$

□

Now, we are ready to bound the work of the RETRIGGER operation described in [Section 9.3.2](#). The work is defined in terms of a  $c$ -divide-and-conquer algorithm as defined in [Definition 9.3.2](#).

**Lemma 9.3.9** (Expected Work of RETRIGGER Operation). *Provided a  $c$ -divide-and-conquer algorithm ([Definition 9.3.2](#)), the expected work of  $\text{RETRIGGER}(t_1, t_2)$  is bounded by*

$$\begin{aligned}
&\tilde{O}(|t_2 - t_1| \cdot T^{c-1}) \quad \text{for } c > 1 \\
&\tilde{O}(|t_2 - t_1|) \quad \text{for } c \leq 1,
\end{aligned}$$

where  $W_{t_1, t_2}$  is the smallest window containing  $t_1$  and  $t_2$  and all days in between.

*Proof.* Consider a call to  $\text{RETRIGGER}(t_1, t_2)$ . We first show the following observation that our algorithm given in [Algorithm 9.1](#) assigns at most  $O(\log(T))$  (real or predicted) events to each day. First, by [Lemma 9.3.3](#), after preprocessing, we assign at most two predicted events to each day. Then, predicted events become reassigned by either  $\text{PROCESSEVENTEARLIERTHANPREDICTION}$  or  $\text{PROCESSEVENTLATERTHANPREDICTION}$ . An event gets processed by  $\text{PROCESSEVENTEARLIERTHANPREDICTION}$  when a real event happens on a day earlier than the prediction. By the guarantee that at most one real event occur on any day, at most one predicted event can be moved to a day  $t \in [T]$  by  $\text{PROCESSEVENTEARLIERTHANPREDICTION}$ . Then,  $\text{PROCESSEVENTLATERTHANPREDICTION}$  reassigns a predicted event to a later day when an event occurs later than predicted. By our procedure for reassigning such events, we keep track of a counter  $i$  for each predicted event, denoting the number of times it has been reassigned. Because  $\mathbf{a}'_0$  contains at most 2 events per day and  $\text{PROCESSEVENTLATERTHANPREDICTION}$  reassigns events in increments of powers of 2, each day  $t \in [T]$  gets assigned at most two events at a distance of  $2^i$  away for all  $i \in [\log(T)]$ . Hence, the number of events assigned to any day is  $O(\log(T))$ . We call this set of events, on day  $t$ , a **batch** of events and denote it as  $\mathcal{B}_t$ .

First, we fix the size of the smallest window containing  $[t_1, t_2]$  and let  $S = |W_{t_1, t_2}|$ . Now, we consider the work done in any one level  $\ell$  of the subtree rooted at  $W_{t_1, t_2}$ . Let the windows in level  $\ell$  of the subtree have sizes  $S_1, \dots, S_k$ . If  $c > 1$ , the work done at this level can be bounded by

$$\sum_{i=1}^k S_i^c \leq \sum_{i=1}^k S_i \cdot T^{c-1} = ST^{c-1},$$

since  $S_i \leq T$ . If  $c \leq 1$ , the work of the subtree rooted at  $W_{t_1, t_2}$  is dominated by the root,

and so, the work done at level  $\ell$  is bounded by

$$\sum_{i=1}^k S_i \leq S.$$

By [Lemma 9.3.5](#), if we set  $k = 24(c + 2)$ , we get that the depth of the random partition-tree is  $24(c + 2) \cdot O(\log(T))$  over  $T$  days with probability  $\geq 1 - T^{-(c+2)}$  (since  $c$  is a fixed constant).<sup>6</sup> Conditioning on this event, and accounting for the  $\log(\log(T \cdot |\mathcal{S}|))$  blowup from using persistent memory, we get that the total work is bounded by

$$O(S \cdot T^{c-1} \log(T) \log(\log(T \cdot |\mathcal{S}|))) \quad \text{when } c > 1 \tag{9.5}$$

$$O(S \log(T) \log(\log(T \cdot |\mathcal{S}|))) \quad \text{when } c \leq 1, \tag{9.6}$$

where the extra  $\log(T)$  comes from multiplying our work per level by  $O(\log(T))$  levels.

Each day corresponds to a batch of  $O(\log T)$  events, so there are at most  $|t_2 - t_1| \log T$  events between  $t_1$  and  $t_2$ . By [Lemma 9.3.8](#), we can bound the expected size of  $S = |W_{t_1, t_2}|$  to be  $\mathbf{E}[S] = O(|t_2 - t_1| \log^2(T))$ . Thus, if we condition on the event that the depth of the partition-tree is  $O(\log(T))$ , then the expected work, using [Eqs. \(9.5\) and \(9.6\)](#), denoted by  $\text{work}(W_{t_1, t_2})$ , is given by

$$\mathbf{E}[\text{work}(W_{t_1, t_2}) \mid \text{partition-tree has depth } O(\log T)] = \tag{9.7}$$

$$\begin{cases} O(|t_2 - t_1| \cdot T^{c-1} \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c > 1 \\ O(|t_2 - t_1| \cdot \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c \leq 1. \end{cases} \tag{9.8}$$

---

<sup>6</sup>We can choose to set  $k$  to be an arbitrarily large constant if we want to increase the probability of success.



In the case where the tree is not  $O(\log T)$  depth, which happens with probability at most  $T^{-(c+2)}$ , we have the trivial bound that the tree can be depth at most  $T$ . Thus, the work over the subtree can be bounded as

$$\mathbf{E}[\text{work}(W_{t_1, t_2}) \mid \text{partition-tree does not have depth } O(\log T)] = \begin{cases} O(|t_2 - t_1| \cdot T^c \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c > 1 \\ O(|t_2 - t_1| \cdot T \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c \leq 1. \end{cases}$$

We can also trivially bound that  $|t_2 - t_1| \leq T$ , allowing us to trivially bound the expected work by

$$\mathbf{E}[\text{work}(W_{t_1, t_2}) \mid \text{partition-tree does not have depth } O(\log T)] = \tag{9.9}$$

$$\begin{cases} O(T^{c+1} \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c > 1 \\ O(T^2 \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) & \text{when } c \leq 1. \end{cases} \tag{9.10}$$

Using the two cases given in [Eqs. \(9.8\)](#) and [\(9.10\)](#) allows us to bound the expected work over

this subtree by

$$\begin{aligned}
\mathbf{E}[\text{work}(W_{t_1, t_2})] &= \mathbf{E}[\text{work}(W_{t_1, t_2}) \mid \text{depth } O(\log T)] \cdot \text{Prob}[\text{depth } O(\log T)] \\
&\quad + \mathbf{E}[\text{work}(W_{t_1, t_2}) \mid \text{not depth } O(\log T)] \cdot \text{Prob}[\text{not depth } O(\log T)] \\
&= (1 - T^{-(c+2)}) \cdot O(|t_2 - t_1| \cdot \log(T)) \cdot T^{c-1} \log^3(T) \log(\log(T \cdot |\mathcal{S}|)) \\
&\quad + T^{-(c+2)} \cdot O(T^{c+1} \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) \\
&= O(|t_2 - t_1| \cdot T^{c-1} \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) \quad \text{when } c > 1 \\
&= (1 - T^{-(c+2)}) \cdot O(|t_2 - t_1| \cdot \log(T)) \cdot \log^3(T) \log(\log(T \cdot |\mathcal{S}|)) \\
&\quad + T^{-(c+2)} \cdot O(T^2 \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) \\
&= O(|t_2 - t_1| \cdot \log^3(T) \log(\log(T \cdot |\mathcal{S}|))) \quad \text{when } c \leq 1.
\end{aligned}$$

This proves our desired bounds.  $\square$

### 9.3.5 Total Work Over All Calls to Retrigger

In the previous subsection, we bounded the work done by a single call to RETRIGGER. In this section, we bound the total work done by calls to RETRIGGER over the entire run of [Algorithm 9.1](#).

**Lemma 9.3.10** (Expected Work Over All Calls to RETRIGGER). *The expected work done by [Algorithm 9.1](#) over all calls to RETRIGGER is*

$$\tilde{O}(|\mathbf{p} - \mathbf{r}|_1 \cdot T^{c-1}) \quad \text{when } c \geq 1$$

$$\tilde{O}(|\mathbf{p} - \mathbf{r}|_1) \quad \text{when } c < 1.$$

*Proof.* First, consider a single event  $e$ . Let  $\mathbf{a}_0(e)$  be the day that  $e$  is assigned to after the preprocessing step (Lemma 9.3.3). Let  $\mathbf{r}(e)$  be the true day on which  $e$  occurs.

If  $\mathbf{r}(e) \leq \mathbf{a}_0(e)$ , then  $\text{RETRIGGER}(\mathbf{r}(e), \mathbf{a}_0(e))$  is called exactly once for  $e$ . Otherwise, there is a sequence of reassignments, following a doubling search procedure, over  $\mathbf{a}_1(e), \dots, \mathbf{a}_k(e)$ , where  $\mathbf{a}_{i+1}(e) = 2^i + \mathbf{a}_i(e)$ ,  $\mathbf{a}_i(e) \leq \mathbf{r}(e)$  for  $i < k$ , and the final  $\mathbf{a}_k(e) \geq \mathbf{r}(e)$ .  $\text{RETRIGGER}$  is called for each of these reassignments:  $\text{RETRIGGER}(\mathbf{a}_i(e), \mathbf{a}_{i+1}(e))$  for  $i = 0, \dots, k-1$ , and finally called one last time as  $\text{RETRIGGER}(\mathbf{r}(e), \mathbf{a}_k(e))$ . Thus, by linearity of expectation, the total expected work done is

$$\left[ \sum_{i=0}^{k-1} \text{work}(\text{RETRIGGER}(\mathbf{a}_i(e), \mathbf{a}_{i+1}(e))) \right] + \text{work}(\text{RETRIGGER}(\mathbf{r}(e), \mathbf{a}_k(e))). \quad (9.11)$$

Assume for simplicity we are in the  $c > 1$  case. (The  $c \leq 1$  analysis simply doesn't contain the  $T^{c-1}$  term.) We use Lemma 9.3.9 to bound Equation (9.11) by

$$\begin{aligned} & \left[ \sum_{i=0}^{k-1} O(|\mathbf{a}_{i+1}(e) - \mathbf{a}_i(e)| \cdot T^{c-1} \log^3(T) \log \log(T \cdot |\mathcal{S}|)) \right] \\ & \quad + O(|\mathbf{a}_k(e) - \mathbf{r}(e)| \cdot T^{c-1} \log^3(T) \log \log(T \cdot |\mathcal{S}|)) \\ & \leq \left( \sum_{i=0}^{k-1} |\mathbf{a}_{i+1}(e) - \mathbf{a}_i(e)| + |\mathbf{a}_k(e) - \mathbf{r}(e)| \right) \cdot O(T^{c-1} \log^3(T) \log \log(T \cdot |\mathcal{S}|)) \\ & = O(|\mathbf{a}_0(e) - \mathbf{r}(e)| \cdot T^{c-1} \log^3 T \log \log(T \cdot |\mathcal{S}|)). \end{aligned}$$

The final inequality follows by first observing that

$$|\mathbf{r}(e) - \mathbf{a}_k(e)| \leq |\mathbf{r}(e) - \mathbf{a}_0(e)|, \quad (9.12)$$

by the nature of the doubling search. This is because  $\mathbf{a}_k(e)$  only exists if  $\mathbf{a}_{k-1}(e) < \mathbf{r}(e)$ , and

$$\mathbf{a}_{k-1}(e) = \frac{1}{2}(\mathbf{a}_0(e) + \mathbf{a}_k(e)).$$

This means that

$$\begin{aligned} \sum_{i=0}^{k-1} |\mathbf{a}_{i+1}(e) - \mathbf{a}_i(e)| &= |\mathbf{a}_k(e) - \mathbf{a}_0(e)| \\ &\leq |\mathbf{a}_k(e) - \mathbf{r}(e)| + |\mathbf{r}(e) - \mathbf{a}_0(e)| \\ &\leq 2|\mathbf{a}_0(e) - \mathbf{r}(e)| \quad \text{by (9.12)} \end{aligned}$$

So in total,

$$\sum_{i=0}^{k-1} |\mathbf{a}_{i+1}(e) - \mathbf{a}_i(e)| + |\mathbf{a}_k(e) - \mathbf{r}(e)| \leq 3|\mathbf{a}_0(e) - \mathbf{r}(e)| = O(|\mathbf{a}_0(e) - \mathbf{r}(e)|).$$

This bounds the expected cost of calls to RETRIGGER related to the specific event  $e$ . To bound the work over all events  $e$ , recall we denote the vector of input predictions as  $\mathbf{p}$ . Then we can bound the total work by

$$\begin{aligned} &\sum_e O(|\mathbf{a}_0(e) - \mathbf{r}(e)| \cdot T^{c-1} \log^3(T) \log \log(T \cdot |\mathcal{S}|)) \\ &= |\mathbf{a}_0 - \mathbf{r}|_1 \cdot O(T^{c-1} \log^3(T) \log \log(T \cdot |\mathcal{S}|)) \\ &= O(|\mathbf{p} - \mathbf{r}|_1 \cdot T^{c-1} \log^4(T) \log \log(T \cdot |\mathcal{S}|)), \end{aligned}$$

where the last line uses the [Lemma 9.3.3](#) scheduling quality, when  $c > 1$ . Using the same analysis and [Lemma 9.3.9](#), we can bound the total work by

$$O(|\mathbf{p} - \mathbf{r}|_1 \cdot \log^4(T) \log \log(T \cdot |\mathcal{S}|)), \quad \text{when } c < 1.$$

□

### 9.3.6 Boosting to High Probability

#### Best-of-All-Worlds Backstop

In this section we describe how to compose  $N$  dynamic algorithms  $\{A_1, \dots, A_N\}$ , that achieve amortized guarantees leading to a framework that achieves both *competitiveness* and our desired work bound *with high probability* (described in [Section 9.3.6](#)). That is, by day  $t$ , algorithm  $A_i$  has done total work  $R_{A_i}(t)$ . We show how to use  $\{A_1, \dots, A_N\}$  to design one algorithm that by day  $t$  has done total work

$$O(N \cdot \min\{A_1, \dots, A_N\}).$$

The merit of implementing such a backstop depends on the use-case of the algorithm. In some cases, we would hope that a predicted-updates algorithm can significantly outperform the fully-dynamic algorithm in some but not all regimes, in which case the backstop may be useful. In other cases, the motivation could be simplicity of implementation. The framework presented in this work does not rely on algorithmic “heavy machinery,” and in some cases, could be more practical to implement without a backstop.

Below, we define *one computation step* as one word operation in the word-RAM. The reduction is straightforward, and relies heavily on the fact that the goal is an amortized bound, and not a worst-case update time: we run the algorithms “in parallel,” and we return the output of the algorithm that terminates first. We show our desired work bounds in [Theorem 9.3.11](#) and give the formal proof in [Section A.2](#).

**Theorem 9.3.11** (Best-of-All-Worlds Backstop). *Fix  $N$  algorithms  $\{A_1, \dots, A_N\}$ , that will see the same, a priori unknown, input sequence over  $T$  days. Define  $R_{A_i}(t)$  to be the total work that  $A_i$  does over the first  $t$  days.*

We can design a meta-algorithm  $M$ , such that

$$\forall t, \quad R_M(t) = O(N \cdot \min\{R_{A_1}(t), \dots, R_{A_N}(t)\}),$$

where  $R_M(t)$  is the total work that  $M$  does over the first  $t$  days.

### Obtaining Work Bounds with High Probability

In this section, we show how to modify our partition-tree-based algorithm using our backstop meta-algorithm to obtain our running time bounds with high probability, using a simple “boosting” argument. We first make the observation that if we run  $O(\log(T))$  independent instantiations of our algorithm, at least one instantiation will have runtime close to the expectation with high probability.

We can also remove the assumption that  $T$  is known, by using an additional guess-and-double argument; such an argument comes in handy for our incremental or decremental transformation to fully dynamic in later sections.

**Lemma 9.3.12** (Boosting Argument). *Consider  $k \log(T)$  independent instantiations of the predicted-updates algorithm (Algorithm 9.1), for some constant integer  $k > 0$ . With probability at least  $1 - \frac{1}{T^k}$ , at least one of these instantiations has runtime*

$$\begin{aligned} \tilde{O}(T^{c-1} \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1)) & \quad \text{when } c > 1, \\ \tilde{O}(T + \|\mathbf{p} - \mathbf{d}\|_1) & \quad \text{when } c \leq 1. \end{aligned}$$

*Proof.* Consider a single instantiation of the predicted-updates algorithm given in Algorithm 9.1. By Lemmas 9.3.4 and 9.3.10, the total expected work of this algorithm (including

preprocessing) is

$$O\left(T^{c-1} \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1) \cdot \log^{c+3}(T) \cdot \log \log(T \cdot |\mathcal{S}|)\right) \quad \text{when } c > 1, \quad (9.13)$$

$$O\left((T + \|\mathbf{p} - \mathbf{d}\|_1) \cdot \log^4(T) \cdot \log \log(T \cdot |\mathcal{S}|)\right) \quad \text{when } c \leq 1. \quad (9.14)$$

By Markov's inequality, the probability that the runtime of a single instantiation exceeds two times the expected work is at most  $\frac{1}{2}$ . Thus, the probability that  $k \log(T)$  independent instantiations all simultaneously exceed two times the expected work is

$$\leq \left(\frac{1}{2}\right)^{k \log(T)} = \frac{1}{T^k}.$$

Thus, with probability at least  $1 - \frac{1}{T^k}$ , one of the  $k \log(T)$  instantiations has runtime at two times the expected work given in [Eqs. \(9.13\) and \(9.14\)](#).  $\square$

Now, using the backstop technique, we take  $O(\log(T))$  instantiations of the predicted updates algorithm ([Algorithm 9.1](#)) to create a composite algorithm that has runtime that scales with the minimum of the instantiations.

We observe that the guarantee from [Lemma 9.3.3](#) and [Lemma 9.3.10](#), and therefore [Lemma 9.3.12](#) as well, requires the algorithm to have access to the size of the time horizon  $T$ . We show how to set this via a guess-and-double procedure assuming we receive sets of updates  $P_1, P_2, \dots, P_T$  of successively larger sizes where  $P_i \subseteq P_{i+1}$  and  $|P_{i+1}| = 2 \cdot |P_i|$ . Together, this gives us [Algorithm 9.4](#) and our final theorem below.

### 9.3.7 Putting Everything Together: the Final Theorem

We use [Algorithm 9.4](#) and all our prior proofs to show our final reduction. We use a  $c$ -divide-and-conquer algorithm as well as a partition-tree which are given in [Definition 9.3.2](#)

and [Definition 9.3.1](#), respectively.

**Theorem 9.3.13** (Offline Divide-and-Conquer to Fully Dynamic with Predictions Reduction). *Given a  $c$ -divide-and-conquer algorithm  $\mathcal{A}$  that computes the solution to a problem  $\mathcal{P}$ , predictions  $P$ , and online sequence of events  $U = [E_1, \dots, E_T]$ , [Algorithm 9.4](#) (using [Algorithm 9.1](#)) correctly outputs the solution to  $f(U_t)$  after each  $t \in [T]$  and uses total work, with high probability,*

$$\begin{aligned} \tilde{O}(T^{c-1} \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1)) & \quad \text{when } c > 1, \\ \tilde{O}(T + \|\mathbf{p} - \mathbf{d}\|_1) & \quad \text{when } c \leq 1. \end{aligned}$$

Furthermore, our algorithm never performs worse than the best-known fully dynamic algorithm for problem  $\mathcal{P}$ .

*Proof.* Given our inputs, we construct our set of random partition-trees using [Algorithm 9.1](#). We first prove that our algorithm correctly outputs  $f(U_t)$  after every event  $E_t$  in the sequence of online events. [Algorithm 9.4](#) runs  $L$  instantiations of [Algorithm 9.1](#) and outputs the answer from the first instantiation that finishes processing event  $E_t$ . By our procedures in [Algorithm 9.1](#), every real update is processed in every window that contains it, and no windows in the union of all windows which do not contain days after  $t$  contain any predicted events that have *not* occurred after event  $E_t$  is processed. Furthermore, by [Definition 9.3.2](#), all windows process all events irrespective of the order of the events. Hence, [Algorithm 9.1](#) correctly returns  $f(U_t)$  for every  $t \in [T]$ .

We now give our high probability bound for the total amount of work performed over all  $T$  real events.

Consider a day  $t = 2^i$  on which [Algorithm 9.4](#) is triggered, and  $\hat{T}$  is doubled.  $\hat{T}$  is newly



set to be  $2^{i+1}$ . Consider the total work done by the new instantiation of  $M$ . Specifically, the work between the event at  $t = 2^i$  and when  $\widehat{T}$  is doubled again at event  $t' = 2^{i+1}$ . By [Theorem 9.3.11](#), we can bound the work of  $M$  by  $L$  times the work of the minimum  $A_i \in \mathcal{A}$  that is drawn independently in this iteration. [Lemma 9.3.12](#) bounds the work of that minimum  $A_i$  by

$$\begin{aligned} \tilde{O}(T^{c-1} \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1)) & \quad \text{when } c > 1, \\ \tilde{O}(T + \|\mathbf{p} - \mathbf{d}\|_1) & \quad \text{when } c \leq 1, \end{aligned}$$

with probability  $\geq \max(1 - \frac{1}{t^k}$  if  $\log(t) > \log(|\mathcal{S}|)$  and  $\geq 1 - \frac{1}{t^{k \cdot \log(|\mathcal{S}|)}}$ , otherwise. The probability is due to the fact that we create  $L = \max(k \cdot \log(T), k \cdot \log(|\mathcal{S}|))$  independent instantiations.

Now, can use a union bound to conclude that our work bounds hold for all integers  $t \in [T]$  with probability

$$\begin{aligned} & \geq 1 - \left( \sum_{i=0}^{\infty} \frac{1}{2^{k(i+1)}} + \frac{\log(T)}{2^{k \cdot \log |\mathcal{S}|}} \right) \\ & \geq 1 - \frac{2 \log(T)}{|\mathcal{S}|^k}, \end{aligned}$$

which is high probability for sufficiently large constant  $k > 0$ .

This condition allows us to, for a day  $t$ , bound the total work done by [Algorithm 9.3](#) up through day  $t$ . Define  $i$  such that  $2^i \leq t < 2^{i+1}$ . We can bound the work done by [Algorithm 9.3](#) up through day  $t$  as

$$\sum_{i=0}^{\lfloor \log_2(t) \rfloor} O(\|\mathbf{p} - \mathbf{r}\|_1 \cdot (2^i)^{c-1} \log^{c+3}(2^i) \log \log(2^i)) = O(\|\mathbf{p} - \mathbf{r}\|_1 \cdot (t)^{c-1} \log^{c+3}(t) \log \log(t \cdot |\mathcal{S}|)),$$

when  $c > 1$ , and  $O(|\mathbf{p} - \mathbf{r}|_1 \cdot \log^4(t) \log \log(t))$  when  $c \leq 1$ . Substituting  $T$  for  $t$  into the equations results in our desired work.

Finally, by composing [Algorithm 9.4](#) with the best-known algorithm for  $\mathcal{P}$ , once again using [Theorem 9.3.11](#), we never perform worse than the best-known algorithm for the problem.  $\square$

Finally, we observe that this framework gives a stronger in the update/query model, when we reduce to an algorithm that has worst-case query time. That is, an algorithm that handles updates *without* information about queries, can continue to do so in this model. This is in contrast to some offline divide-and-conquer algorithms that use information about what elements will be queried to make decisions.

**Corollary 9.3.14** (Bound for Update/Query problems). *Given a  $c$ -divide-and-conquer algorithm  $\mathcal{A}$  for  $f(\cdot)$  with worst-case query time  $\text{query}(A)$ , we can construct an algorithm, such that with high probability with respect to  $T$ , has total work*

$$\begin{aligned} \tilde{O}(T^{c-1} \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1)) & \quad \text{when } c > 1, \\ \tilde{O}(T + \|\mathbf{p} - \mathbf{d}\|_1) & \quad \text{when } c \leq 1, \end{aligned}$$

*and the worst-case query time is always bounded by  $\text{query}(A)$ .*

*Furthermore, our algorithm never does more total work (over updates and queries) than the best-known fully dynamic algorithm for problem  $\mathcal{P}$ .*

*Proof.* For the first part of the corollary, we can think of an update/query problem as simply being an update problem, where after each update, the algorithm must return a data structure that is compatible with the query algorithm. Using [Theorem 9.3.13](#) on this update problem, we get a predicted-dynamic algorithm that always returns a data structure that is

compatible with the original query algorithm. Thus, the query algorithm is unchanged.

For the second part of the corollary, we use the backstop procedure of [Theorem 9.3.11](#) to backstop the predicted-deletion algorithm with the fully-dynamic algorithm.  $\square$

## 9.4 Incremental to Fully-Dynamic Transformation

In this section we show how our framework that lifts offline algorithms to the fully-dynamic setting, given predictions of all update times, can be adapted to lift an incremental algorithm to the fully-dynamic setting, given predictions of only the deletion times. Formally, we consider the following model.

**Definition 9.4.1** (Predicted-Deletion Dynamic Model). *In the predicted-deletion dynamic model, we consider a ground set  $\mathcal{S}$ . On each day exactly one of the following occurs:*

1. *An element  $e \in \mathcal{S}$  is inserted, and reports a prediction of the day on which it will be deleted.*
2. *A previously inserted element is deleted.*

*An algorithm computes a function  $f(\cdot)$  in the predicted-deletion dynamic model, if on every day  $t$ , the algorithm outputs  $f(S)$ , where  $S \subseteq \mathcal{S}$  is the working subset induced by the true (not predicted) sequence of element insertions and deletions that occur in time-steps  $1, \dots, t$ .*

This model is stronger than the general predicted-updates model that was presented in the last section. In the previous section, we charged the runtime of the fully-dynamic algorithm to the prediction error in insertions and deletions. In this model, the runtime of the fully-dynamic algorithm should only depend on the predictions of deletion times. This corre-

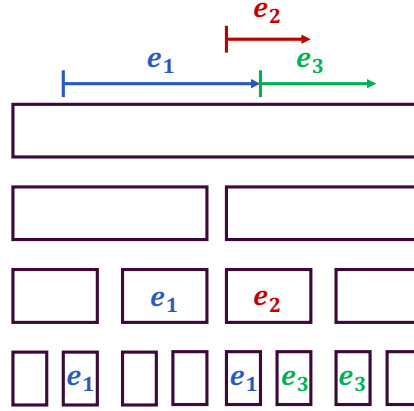


Figure 9.4: An illustration of which windows consider the elements  $e_1$ ,  $e_2$ , and  $e_3$  permanent

sponds to the fact that we are starting with an incremental algorithm rather than an offline algorithm. Thus, the algorithm we start with can already handle arbitrary insertions, and our reduction adds functionality only in handling deletions.

First, in what may seem like a step in the wrong direction, we show how to build an *offline dynamic* divide-and-conquer algorithm, using an incremental algorithm with a worst-case guarantee. To do this, we introduce the concept of permanent elements.

**Definition 9.4.2** (Permanent elements). *An element is permanent with respect to a window  $W$  of a partition-tree if*

1. *the element is inserted on or before the first day of  $W$ ,*
2. *the item is deleted after the last day of  $W$ ,*
3. *and the element is not permanent for any ancestor of  $W$ .*

This is illustrated in [Figure 9.4](#). Note that for a day  $t$ , the permanent elements of the windows containing  $t$  perfectly partition the elements that are active on day  $t$ . Conversely, we also have that the windows for which an element  $e$  is permanent perfectly partition the lifetime of  $e$ . This suggests a natural way to convert an (online) incremental algorithm,  $\mathcal{A}$ ,

with worst-case update time  $\text{update}(\mathcal{A})$  into a divide-and-conquer offline algorithm.

**Lemma 9.4.3** (Offline-dynamic divide-and-conquer from incremental). *An incremental algorithm,  $\mathcal{A}$ , with worst-case update time  $\text{update}(\mathcal{A})$ , can be converted into a offline-dynamic divide-and-conquer algorithm satisfying [Definition 9.3.2](#), where the work at each window,  $W$ , is*

$$O(\text{update}(\mathcal{A}) \cdot |W|).$$

*Proof.* At the root node, we initialize the state of the incremental algorithm. Then, at each window, we insert the permanent elements associated with that window into the state of the incremental algorithm. The set of permanent elements can be computed from the set of all elements that are active at any time during the parent window, in a linear-time scan. Thus, the work done at a window scales with  $\text{update}(\mathcal{A})$  times the number of permanent edges associated with that window.

Note that any element that is permanent for a window  $W$ , must have either been inserted or deleted during  $W$ 's sibling window. Otherwise, this element would actually be permanent for  $W$ 's parent window. Thus, the number of permanent elements of  $W$  is bounded by the size of  $W$ 's sibling. For a random partition-tree, this is the same as the size of  $W$  in expectation. So this meets the third criterion of [Definition 9.3.2](#), where the expected work at a node is linear in the size of the node, with a multiplicative factor of  $\text{update}(\mathcal{A})$ .

With respect to the second criterion of [Definition 9.3.2](#), we need that the computation of the window depends only on the state of the parent window, and the *unordered set* of updates occurring in this window. At first glance, this does not appear to be the case, as calculating the set of permanent elements requires the algorithm to compare the set of updates of its

sibling window. However, we can consider the set of updates of the parent window to be included in the state of the parent window. Then, since the number of updates of the parent is, in expectation, a constant factor more than the size of the child window, the child window can reconstruct the necessary information from the state of the parent and its own set of updates.

Thus, this algorithm meets the criteria of [Definition 9.3.2](#), where the expected work at each window  $W$  is

$$O(\text{update}(\mathcal{A}) \cdot |W|).$$

□

Now, if we directly apply [Theorem 9.3.13](#), we recover an algorithm that does total work

$$O((T + \|\mathbf{p} - \mathbf{d}\|_1) \cdot \text{update}(\mathcal{A}) \cdot \log^3(T) \log \log(T \cdot |\mathcal{S}|)),$$

when given predictions of *all updates* at the outset of the algorithm. However, we wish to consider a setting in which the elements arrive arbitrarily, and an element's predicted deletion time is only provided when the element is inserted.

This leads to two main issues. The first is that we cannot create a tentative schedule of events during preprocessing, because we have no information about the events that are going to occur. The second is that we cannot compute a preliminary version of the divide-and-conquer algorithm during preprocessing, because we do not have a tentative schedule of events to refer to. To address the first issue, we make the following observation.

**Observation 9.4.4** (Online scheduling). *Even in the setting where the initial schedule of events is computed statically in the preprocessing step, we use an online algorithm to assign events to days.*

*Thus, as predicted events arrive over time, we can assign each event irrevocably to an initial tentative schedule using this online algorithm, while maintaining the runtime and approximation guarantees of the original algorithm.*

Thus, we can schedule events as they arrive, without having to reschedule events to account for later arrivals.

To address the second issue, we show that, while we don't know enough about the schedule to run the entire divide-and-conquer algorithm during preprocessing, we actually can utilize a “just in time” approach.

**Lemma 9.4.5** (Windows can be computed “just in time”). *Consider a offline-dynamic divide-and-conquer algorithm of the form constructed by [Lemma 9.4.3](#). Then, the fully-dynamic algorithms with predictions resulting from [Theorem 9.3.13](#) can be run in a “just in time” fashion where*

- 1. the computation associated with a window  $W$  beginning on day  $t$  is only run on or after day  $t$ , and*
- 2. on day  $t$ , we have enough information to run the computation associated with all windows starting on day  $t$ , and*
- 3. RETRIGGER is never called for an insertion event.*

*Proof.* First, we note that the output of the fully-dynamic algorithm resulting from [Theorem 9.3.13](#) on day  $t$  only depends on the computations associated with windows that contain  $t$ . Thus, the first point of the lemma is true for *any* algorithm of this form. That is, on a day  $t$ , we can consider all windows starting after day  $t$  to be “inactive,” and ignore them when they are part of a RETRIGGER operation. Then, on day  $t$ , we run the computations

of any windows beginning exactly on day  $t$ , for the first time.<sup>7</sup>

For the second point, consider a window  $W$  beginning on some day  $t$ . It is possible that on day  $t$ , there are some events that will occur during  $W$ , for which we do not have predicted times. For example, insertions occur entirely arbitrarily, so we do not have any information about the insertions that will occur during  $W$ . However, any element that could be permanent for  $W$ , must *already* be inserted. This is by definition: for an element to be permanent for  $W$ , it must be inserted on or before the first day of  $W$ . Thus, on day  $t$ , we have enough information to compute all of the permanent elements of  $W$ , which is enough to run the divide-and-conquer algorithms constructed by [Lemma 9.4.3](#).

Finally, for the third point, note that any window that depends on an insertion event  $e$  is only computed for the first time after  $e$  occurs. Thus, at this point,  $e$  is fully known, and will not be rescheduled, so it will never call RETRIGGER.  $\square$

Altogether, [Lemma 9.4.3](#), [Observation 9.4.4](#), and [Lemma 9.4.5](#), along with [Theorem 9.3.13](#) give us the following theorem.

**Theorem 9.4.6** (Incremental to Fully-Dynamic). *Given an incremental algorithm,  $\mathcal{A}$ , with expected worst-case update time  $\text{update}(\mathcal{A})$ , we can construct an algorithm in the predicted-deletion model ([Definition 9.4.1](#)), such that the total expected work done by the algorithm is*

$$O\left(\text{update}(\mathcal{A}) \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1) \cdot \log^4 T \log \log(T \cdot |\mathcal{S}|)\right),$$

where  $\|\mathbf{p} - \mathbf{d}\|_1$  is the  $\ell_1$  error of the deletion-time predictions.

Using [Theorem 9.3.13](#), we obtain the following work, with high probability.

---

<sup>7</sup>In fact, while we don't get a better asymptotic runtime bound, running any algorithm of this form in a "just in time" way can only reduce the total work done, as we do not have to retrigger windows that are not yet active.



**Corollary 9.4.7.** *Given an incremental algorithm,  $\mathcal{A}$ , with expected worst-case update time  $\text{update}(\mathcal{A})$ , we can construct an algorithm in the predicted-deletion model (Definition 9.4.1), such that the total work done by the algorithm is  $\tilde{O}(\text{update}(\mathcal{A}) \cdot (T + \|\mathbf{p} - \mathbf{d}\|_1))$  with high probability, where  $\|\mathbf{p} - \mathbf{d}\|_1$  is the  $\ell_1$  error of the deletion-time predictions.*

### 9.4.1 Comparison to Independent Concurrent Work of Peng and Rubinstein [2023], van den Brand et al. [2023]

The independent and concurrent work of van den Brand, Forster, Nazari, and Polak [2023] also provides a reduction that lifts an incremental algorithm to the fully dynamic setting given predictions of deletion times. In this section, we overview their approach and provide a comparison to our reduction.

The reduction of van den Brand, Forster, Nazari, and Polak [2023] is based on a simple and elegant observation over the previous work of Peng and Rubinstein [2023]. Peng and Rubinstein [2023] consider a model that they call the *deletion look-ahead model*. In this model, on every day  $t$ , the relative deletion order of all elements that are active in the system is known. Another way to think of this model is that, when an element is inserted, it announces which of the existing elements it will be deleted after. This is similar to our predicted-deletion model in the regime where the prediction error is 0. However, it is actually more general, because it only requires *ordinal* information rather than *cardinal* information about the deletion times.

In the language of our work, we can interpret the reduction of Peng and Rubinstein [2023] as doing something similar to constructing the offline divide-and-conquer algorithm from Lemma 9.4.3, and running it with a just-in-time approach (in the manner of Lemma 9.4.5). This is all with respect to a fixed deterministic partition tree (i.e. a perfectly balanced binary

tree). Indeed, if we had access to cardinal information (i.e. the actual day on which each edge is predicted to be deleted), this would suffice to recover an amortized guarantee for this problem.

To adapt this to the ordinal setting, [Peng and Rubinstein \[2023\]](#) make the following observation. Over the course of any window  $W$ , at most  $|W|$  elements can be deleted. Thus, at the start of a window  $W$ , any element that is *not* in the  $|W|$  earliest-to-be-deleted elements will certainly be permanent over  $W$ ,<sup>8</sup> and can safely be inserted. At the end of the computation associated with  $W$ , there are still up to  $|W|$  elements that we did not insert, because we were unsure of whether or not they would be permanent. These elements are passed to the child windows, which process these elements, along with any elements that were inserted in their sibling windows. In all, the total number of elements that any window has to process is at most a constant factor more than the size of the window.

This adaptation to the ordinal setting leaves the reduction of [Peng and Rubinstein \[2023\]](#) with a very useful property: the state of the incremental algorithm at each leaf of the partition tree has elements inserted in *approximately reverse deletion order*. That is, any element that will be deleted in the next  $d$  deletions, was one of the last  $c \cdot d$  elements to be inserted, where  $c$  is a constant.

Now, in the case where the predicted deletion order is subject to error, [van den Brand et al. \[2023\]](#) observe that this property allows us to fix the data structure with low overhead. In particular, consider an element  $e$  that is deleted earlier than predicted. That is, on day  $t$ ,  $e$  was predicted to be the  $d$ -th next element to be deleted, but instead it was the first element to be deleted. The approximate reverse deletion order property tells us that  $e$  was in the last  $c \cdot d$  elements to be inserted into this leaf. This means that it was inserted in some window

---

<sup>8</sup>Here, we are abusing our notation of permanent elements from earlier, and referring to an element as permanent if it is inserted on or before the first day of  $W$ , and deleted on or after the last day of  $W$ .

$W$  relatively close to leaf-level in the partition-tree. Thus, to fix the data structure, they only need to retrigger these few windows that are descendants of  $W$ .

Essentially, this approximate reverse deletion order property allows the data structure to be fixed on the fly, with low overhead. Furthermore, their strategy does *not* require a random partition-tree: indeed it is fully deterministic, and can even be adapted to provide a worst-case per update guarantee (as opposed to our work which provides an amortized guarantee).

In comparison to our work, it is not clear if it is possible to extend the [Peng and Rubinstein, 2023, van den Brand, Forster, Nazari, and Polak, 2023] approach to lift offline algorithms to the fully dynamic setting. In particular, we provide a simple analogue of their approach that can convert a decremental algorithm to the fully-dynamic setting, when given predictions of insertion times. We explore this connection formally in the next section (Section 9.5). With respect to the specific reduction of van den Brand, Forster, Nazari, and Polak [2023], it essentially maintains state of a decremental algorithm, where elements are deleted in *approximately reverse insertion order*.

While this approach can handle predicted deletion times given an incremental algorithm, and predicted insertion times given a decremental algorithm, it is not clear if and how these two guarantees can be combined. In particular, this runs into two issues.

1. It is not clear what kind of algorithm to reduce to, as one direction requires an incremental algorithm, and the other direction requires a decremental algorithm.
2. It is not clear what the analogue of maintaining elements in both reverse insertion *and* reverse deletion order should be.

Our work addresses the first point by making the key observation that we can reduce to an offline divide-and-conquer algorithm. As to the second point, we resolve the issue by using

a random partition-tree. This does not require us to maintain any ordering of the elements that we consider.

Thus, the reduction of [van den Brand, Forster, Nazari, and Polak \[2023\]](#) for the predicted deletion model is based on a simple and elegant extension of the previous framework of [Peng and Rubinstein \[2023\]](#), which lifts incremental algorithms into the deletion look-ahead setting. The reduction that we present in this work for the predicted deletion model, is instead based on a simple extension from our earlier framework that lifts offline divide-and-conquer algorithms into the predicted update setting. Under a single framework, we generalize all three settings: offline, incremental, and decremental.

## 9.5 Decremental to Fully-Dynamic Transformation

In this section, we show how to adapt our framework to convert a decremental algorithm to the fully-dynamic setting, given predictions of only the insertion times. Formally, we consider the following model.

**Definition 9.5.1** (Predicted-Insertion Dynamic Model). *In the predicted-insertion dynamic model, we consider a set  $S$  of all the elements that are predicted to ever appear in the system. At the outset, the algorithm is given  $S$ , and predictions for when each of the elements in  $S$  will be inserted. Then, on each day, exactly one of the following occurs:*

1. *An element  $e$ , either in  $S$  or not, is inserted,*
2. *A previously inserted element is deleted, and provides a prediction of when it will be reinserted (if ever).*

*An algorithm computes a function  $f(\cdot)$  in the predicted-insertion dynamic model, if on every day  $t$ , the algorithm outputs  $f(S_t)$ , where  $S_t$  is the working subset induced by the true (not*

*predicted*) sequence of element insertions and deletions that occur in time-steps  $1, \dots, t$ .

A number of works in the past have performed fully dynamic to decremental reductions for specific problems, starting with the seminal work of [Henzinger and King \[1997\]](#). We show that, with the help of predictions, we can generalize such ideas to *any* worst-case decremental algorithm. Now, we show that, given a decremental algorithm, we can design algorithms for the predicted insertion model. We do this via a simple reduction to the predicted insertion case.

**Theorem 9.5.2** (Decremental to Fully-Dynamic). *Consider a decremental algorithm,  $\mathcal{A}$ , which takes time  $\text{initialize}_{\mathcal{A}}(T)$  to initialize a state containing up to  $T$  elements, and then has worst-case update time  $\text{update}(\mathcal{A})$ . Given such an algorithm, we can construct an algorithm for the predicted-insertion model ([Definition 9.5.1](#)), that does expected total work*

$$O\left(\text{initialize}_{\mathcal{A}}(T) \cdot K + \text{update}(\mathcal{A}) \cdot (T + \|\mathbf{p} - \mathbf{i}\|_1 + TK) \log^4(T) \log \log(T \cdot |\mathcal{S}|)\right),$$

where  $\|\mathbf{p} - \mathbf{i}\|_1$  is the  $\ell_1$  error of the insertion-time predictions for elements in  $S$ , and  $K$  is the number of elements that are not in  $S$  that are ever inserted.

*Proof.* We reinterpret  $\mathcal{A}$  as an incremental algorithm on “anti-elements.” That is, suppose we initialize our algorithm to contain the elements in a set  $S$ . Then, as a decremental algorithm, the algorithm can handle updates that delete elements of  $S$ . We can also think of this as an incremental algorithm, that can handle the insertion of anti-elements, corresponding to elements in  $S$ .

Now, consider a series of updates in the predicted-insertion model ([Definition 9.5.1](#)). From the view of our decremental algorithm, each element starts as not being in the system, then is inserted at some time for which we are given a prediction, then is deleted at an unknown

arbitrary time, when it predicts its next insertion.

From the view of the incremental algorithm, each anti-element starts *in* the system, then is deleted at some time for which we have a prediction, and then is reinserted again at an unknown arbitrary time, at which point it predicts its next deletion.

Thus, viewing the decremental algorithm as an incremental algorithm on anti-elements, makes it fit directly into our earlier framework for incremental algorithms, because it converts the decremental algorithm into an incremental algorithm, and it converts insertion events into deletion events and vice versa.

The only remaining issue is that this strategy cannot handle the case where an element that was never part of the predicted set  $S$  is inserted. In this case, we must add the new element to  $S$ , and run the initialization of the algorithm again from scratch. This contributes total work

$$O(\text{initialize}_{\mathcal{A}}(T) \cdot K),$$

where  $K$  is the number of such elements outside  $S$  that are ever inserted. Then, we must retrigger the entire tree, which is equivalent to this new element contributing  $\ell_1$ -error of  $T$ . Over all such elements this contributes work

$$O(TK \cdot \log^4 T \log \log(T \cdot |\mathcal{S}|)).$$

Adding these contributions to the work bound from the incremental reduction, gives us our final bound of

$$O(\text{initialize}_{\mathcal{A}}(T) \cdot K + \text{update}(\mathcal{A}) \cdot (T + \|\mathbf{p} - \mathbf{i}\|_1 + TK) \log^4 T \log \log(T \cdot |\mathcal{S}|)).$$

□

Using [Theorem 9.3.13](#), we obtain the following work, with high probability.

**Corollary 9.5.3.** *Consider a decremental algorithm,  $\mathcal{A}$ , which takes time  $\text{initialize}_{\mathcal{A}}(T)$  to initialize a state containing up to  $T$  elements, and then has worst-case update time  $\text{update}(\mathcal{A})$ . Given such an algorithm, we can construct an algorithm for the predicted-insertion model ([Definition 9.5.1](#)), that does total work*

$$\tilde{O}(\text{initialize}_{\mathcal{A}}(T) \cdot K + \text{update}(\mathcal{A}) \cdot (T + \|\mathbf{p} - \mathbf{i}\|_1 + TK)),$$

with high probability, where  $\|\mathbf{p} - \mathbf{i}\|_1$  is the  $\ell_1$  error of the insertion-time predictions for elements in  $S$ , and  $K$  is the number of elements that are not in  $S$  that are ever inserted.

*Remark 9.5.4.* The above reduction is not specific to our particular implementation of the incremental to fully dynamic reduction. In particular, our interpretation of anti-elements can be applied to the incremental to fully-dynamic reduction in the concurrent independent work of [van den Brand, Forster, Nazari, and Polak \[2023\]](#) and achieve their deterministic worst-case per update work bound also in the decremental setting.

## 9.6 Applications: Offline, Incremental, and Decremental to Fully Dynamic

We apply our framework to the following problems to obtain fully dynamic algorithms in the predicted-deletion model. A summary of our runtimes compared to the best-known fully dynamic algorithms is given in [Fig. 9.1](#). We define the problems we study below. Unless specified, the input to the following problems is an input graph  $G = (V, E)$ .

- **All-Pairs Shortest Paths (APSP):** For any pair of queried vertices  $u$  and  $v$ , determine the distance from  $u$  to  $v$ . In the planar diagraph APSP problem, we are given a planar, directed and weighted graph. A  $c$ -approximate APSP algorithm returns an approximation that is within  $c$ -factor of the distance between the pair.
- **Triconnectivity:** For any pair of queried vertices, determine whether the pair is *3-vertex connected*. Two pairs of vertices  $u$  and  $v$  are 3-vertex connected if and only if  $u$  and  $v$  remain connected whenever fewer than two vertices are removed.
- **Dynamic Depth-First Search (DFS) Tree:** A dynamic DFS tree is a DFS tree that is reported after each edge insertion or deletion and is a valid DFS tree for the current graph.
- **All-Pairs Maximum Flow:** For any pair of queried vertices,  $s$  and  $t$ , return the maximum flow between  $s$  and  $t$ . A  $c$ -approximate maxflow algorithm returns a flow value that is a  $c$ -factor approximation of the actual maxflow.
- **All-Pairs Minimum Cut:** For any pair of queried vertices,  $s$  and  $t$ , return the minimum cut between  $s$  and  $t$ . A  $c$ -approximate maxflow algorithm returns a cut value that is a  $c$ -factor approximation of the actual min-cut.
- **Multi-Commodity Concurrent Flow:** For a set of triples  $\{(s_1, t_1, d_1), \dots, (s_P, t_P, d_P)\}$ , return the maximum value  $\alpha$  where, concurrently for all  $i \in [P]$ ,  $s_i$  can send  $\alpha \cdot d_i$  units of flow to  $t_i$ . A  $c$ -approximate multi-commodity concurrent flow algorithm returns a value that is at least  $\alpha/c$ .
- **Uniform Sparsest Cut:** For a given graph  $G = (V, E, \mathbf{w})$ , return

$$\Phi_G = \min_{S \subset V} \frac{\sum_{(u,v) \in E, u \in S, v \notin S} \mathbf{w}(\mathbf{u}, \mathbf{v})}{|S| \cdot |V \setminus S|}.$$



- **Monotone Submodular Maximization:** Given a ground set  $N = [n]$  and a set function  $f : N \rightarrow \mathbb{R}^+$ , function  $f$  is monotone if  $f(A) \geq f(B)$  for any  $B \subseteq A \subseteq N$ . Function  $f$  is submodular if  $f(A \cup \{u\}) - f(A) \leq f(B \cup \{u\}) - f(B)$  for any  $B \subseteq A \subseteq N$  and element  $u$ . Under cardinality constraint  $k$ , the problem maximizes  $\max_{S \subseteq [n], |S|=k} (f(S))$  for some parameter  $1 \leq k \leq n$  and under a matroid constraint  $\mathcal{M}$ , the problem maximizes  $\max_{S \subseteq \mathcal{M}} (f(S))$ . In the dynamic setting, elements can be inserted and deleted from the ground set, and the goal is to maintain  $\max_{S \subseteq \mathcal{M}} (f(S))$ . A  $c$ -approximate algorithm returns a set  $S$  that has value at least  $c \cdot \max_{S \subseteq \mathcal{M}} (f(S))$ .
- **$k$ -Edge Connectivity:** For any pair of queried vertices  $\{u, v\}$ , the pair  $\{u, v\}$  is  $k$ -edge connected if and only if  $u$  and  $v$  remain connected whenever any set of  $k - 1$  edges are removed.
- **Minimum Spanning Forest (MST):** A dynamic minimum spanning forest is an minimum weight spanning forest that is maintained under edge insertions and deletions.
- **Strongly Connected Components (SCC)/Topological Sort:** Given a directed input graph  $G = (V, E)$ , maintain the strongly connected components and topological sort, respectively, under edge insertions/deletions.

**Offline to Fully Dynamic** In this section, we instantiate several offline-to-fully dynamic algorithms using our [Theorem 9.3.13](#). Specifically, we show transformations for triconnectivity,  $k$ -edge connectivity, and minimum spanning forest using the divide-and-conquer algorithms of [Chalermsook, Das, Kook, Laekhanukit, Liu, Peng, Sellke, and Vaz \[2021\]](#), [Eppstein \[1994\]](#), [Peng, Sandlund, and Sleator \[2017\]](#), achieving exponential time improvements on the running times of the best-known fully dynamic algorithms when  $\|\mathbf{p} - \mathbf{r}\|_1 = \tilde{O}(T)$  where  $\mathbf{p}$  is the predicted sequence of update times and  $\mathbf{r}$  are the real update times.

We first prove that the offline algorithms given in [Chalermsook et al. \[2021\]](#), [Eppstein \[1994\]](#), [Peng et al. \[2017\]](#) falls under our definition of  $c$ -divide-and-conquer algorithms and specify the parameters for each algorithm. After proving these lemmas, our framework directly gives fully dynamic algorithms for these problems given predicted-updates.

**Lemma 9.6.1.** *There exists a  $c$ -divide-and-conquer algorithm for offline triconnectivity [[Peng, Sandlund, and Sleator, 2017](#)] where  $c = 1$ .*

*Proof.* [Peng et al. \[2017\]](#) provides a divide-and-conquer algorithm on the sequence of events where each subproblem is half of its parent. In each subproblem, they produce a 3-vertex sparsifier with size equal to the number of non-permanent edges in the subproblem. The permanent edges in each subproblem are used to contract the graph to smaller sizes. For a subproblem  $S$  of size  $|S|$  (where  $S$  contains the non-permanent edges), the sparsification can be done in time  $O(|S|)$ . Thus, [Peng et al. \[2017\]](#) gives a  $c$ -divide-and-conquer algorithm where  $c = 1$ . □

**Lemma 9.6.2.** *There exists a  $c$ -divide-and-conquer algorithm for offline  $k$ -edge connectivity [[Chalermsook, Das, Kook, Laekhanukit, Liu, Peng, Sellke, and Vaz, 2021](#)] for any constant  $k \geq 1$  where  $c = 1$ .*

*Proof.* [Chalermsook et al. \[2021\]](#) provides a divide-and-conquer algorithm that divides the events in approximately equal sizes. Then, in each subproblem  $S$  of size  $|S|$  (where  $S$  contains the non-permanent edges), they construct a vertex sparsifier of size  $\tilde{O}(|S|)$  with the terminals being the set of queried vertices falling within the subproblem. Each sparsifier can be constructed in  $\tilde{O}(|S|)$  time; hence, [Chalermsook et al. \[2021\]](#) gives a  $c$ -divide-and-conquer algorithm where  $c = 1$ . □

**Lemma 9.6.3.** *There exists a  $c$ -divide-and-conquer algorithm for offline minimum spanning forest [[Eppstein, 1994](#)] where  $c = 1$ .*

*Proof.* Eppstein [1994] provides a divide-and-conquer algorithm that recursively divides the sequence into halves. Then, in each subproblem  $S$  where  $|S|$  denotes the number of non-permanent edges, they sparsify using the permanent and non-permanent edges in the following way. They run the MST algorithm by successively picking edges in non-decreasing weight using a standard MST algorithm like Kruskal's in  $O(|S| \log(|S|))$  time, *breaking ties by giving preference to non-permanent edges*. All non-permanent edges that are not picked are deleted and every picked permanent edge is contracted. This results in a sparsifier that has size  $O(|S|)$  that is passed to children. Hence, Eppstein [1994] gives a  $c$ -divide-and-conquer algorithm where  $c = 1$ .  $\square$

Lemmas 9.6.1 to 9.6.3 combined with Theorem 9.3.13 directly give the following theorem.

**Theorem 9.6.4** (Predicted-Updates Algorithms). *Using our framework given in Algorithm 9.4, we obtain the following fully dynamic algorithms in the predicted-updates model, assuming  $\|\mathbf{p} - \mathbf{r}\|_1 = O(T)$ , where  $T$  is the total number of updates,  $\mathbf{p}$  is a vector of predicted event times, and  $\mathbf{r}$  is a vector of real event times:*

1. *An algorithm for triconnectivity in  $\tilde{O}(1)$  amortized update and query times;*
2. *An algorithm for  $k$ -edge connectivity in  $\tilde{O}(1)$  amortized update time and query time for any constant  $k \geq 1$ ; and*
3. *A dynamic minimum spanning forest maintained in  $\tilde{O}(1)$  amortized update time.*

**Incremental to Fully Dynamic** To obtain our fully dynamic algorithms, we use the worst-case incremental algorithms given in the following works Baswana, Chaudhury, Choudhary, and Khan [2019], Chen, Duan, Wang, Zhang, and Zhang [2018], Chen, Goranci, Henzinger, Peng, and Saranurak [2020], Das, Gutenberg, and Wulff-Nilsen [2022], Feldman, Liu, Norouzi-Fard, Svensson, and Zenklusen [2022], Holm and Rotenberg [2020], Goranci [2019],

Goranci, Henzinger, and Saranurak [2019] combined with Corollary 9.3.14 to obtain Theorem 9.6.5.

**Theorem 9.6.5** (Predicted-Deletion Algorithms). *Using our framework, we obtain the following fully dynamic algorithms in the predicted-deletion model, assuming  $\|\mathbf{p} - \mathbf{d}\|_1 = O(T)$ , where  $T$  is the total number of updates,  $\mathbf{p}$  is a vector of predicted deletion times, and  $\mathbf{d}$  is a vector of real deletion times:*

1. *An exact all-pairs shortest path algorithm for planar directed graphs with  $\tilde{O}(\sqrt{n})$  amortized update time and  $\tilde{O}(\sqrt{n})$  worst-case query time;*
2. *An algorithm for triconnectivity in  $\tilde{O}(1)$  amortized update time and  $O(\log^3(n))$  worst-case query time;*
3. *A dynamic DFS tree reported in  $\tilde{O}(n)$  amortized update time;*
4. *A  $O(\log^{8k}(n))$ -approximate maxflow/min-cut algorithm with  $\tilde{O}(n^{2/(k+1)})$  amortized update time and reports the maxflow between any pair of vertices  $s$  and  $t$  in  $\tilde{O}(n^{2/(k+1)})$  worst-case query time;*
5. *A  $O(\log^{8k}(n))$ -approximate multi-commodity concurrent flow algorithm with  $\tilde{O}(n^{2/(k+1)})$  amortized update time and  $\tilde{O}(P^2)$  worst-case query time where  $P$  is the number of queried pairs;*
6. *A  $O(\log^{8k}(n))$ -approximate uniform sparsest cut algorithm with  $\tilde{O}(n^{2/(k+1)})$  amortized update time.*
7. *A 0.3178-approximate monotone submodular maximization algorithm under a matroid constraint of rank  $k$  and makes  $\tilde{O}(\text{poly}(k, \log n))$  function evaluations per update for any  $n, k > 0$ .*

**Decremental to Fully Dynamic** To obtain our fully dynamic algorithms, we use the worst-case decremental algorithms for strongly connected components of [Roditty \[2013\]](#) and topological sort (trivial) [Corollary 9.3.14](#) to obtain [Theorem 9.5.2](#) when  $\|\mathbf{p} - \mathbf{i}\|_1 = O(T)$  where  $\mathbf{p}$  are the predicted times for insertions,  $\mathbf{i}$  is the vector of real insertion times, and  $T$  is the total number of updates.

**Theorem 9.6.6** (Predicted-Deletion Algorithms). *Using our framework, we obtain the following fully dynamic algorithms in the predicted-deletion model, assuming  $\|\mathbf{p} - \mathbf{i}\|_1 = O(T)$ , where  $T$  is the total number of updates,  $\mathbf{p}$  is a vector of predicted insertion times, and  $\mathbf{i}$  is a vector of real insertion times:*

1. *An algorithm for maintaining strongly connected components with  $\tilde{O}(m)$  amortized update time (matching fine-grained lower bounds of [Abboud and Williams \[2014\]](#)); and*
2. *An algorithm for maintaining a topological sort in  $\tilde{O}(1)$  amortized update time.*

## 9.7 Future Directions

**Extension to other problems.** One follow-up direction to our work is to find more problems to which this framework can be applied. This could help design algorithms that are more efficient and/or simpler than the state-of-the-art fully-dynamic solution. Alternatively, it would be interesting to see algorithms designed for this model that look very different from this framework. A particularly interesting follow-up question is: are there problems for which the a predicted-deletion dynamic algorithm can circumvent a fully-dynamic lower bound when the prediction error is sufficiently low?

**Derandomization and/or robustness to adaptive adversary.** The framework presented in this work has a key reliance on randomization and is not robust to an adaptive adversary. It is an interesting question whether the framework can be derandomized, or otherwise made robust to adaptive adversaries.

**Removing amortization.** The framework presented in this work relies heavily on amortization to achieve the desired runtime bounds. It is an interesting question of whether problems in this model can meet a per-update runtime bound. It is also not immediately clear what role prediction error should play in such a bound. For example, one could hope to design an algorithm that, on each day  $t$ , has the update time of an incremental algorithm, with an additive factor that scales with the “error at day  $t$ .” This would require an interesting definition of error in a local sense, in contrast to the global measure of error that we consider in this work ( $\ell_1$  error).

**Lower bounds.** It would be interesting to get fine-grained lower bounds for problems in this model. In particular, this framework accrues logarithmic factors in various places, including a competitive ratio for online metric matching on a line, and the expected size of the smallest node of the partition tree fully containing a subsequence  $[a, b]$ . For the approach in this work, some of the logarithmic factors are necessary side effects of lower bounds for these quantities. It would be interesting to see if there are indeed necessary for any approach to solve dynamic problems in this model, or if they can be circumvented by a different approach.

**Implementation.** A main motivation of this model is that it could potentially allow practitioners to take advantage of known structure in real-world data that classical fully-dynamic algorithms are oblivious to. It would be interesting to see how algorithms designed in this framework compare with the best algorithms and heuristics in practice.

## **9.8 Chapter Acknowledgements**

This chapter is based on joint work with Quanquan C. Liu. We thank Gramoz Goranci and anonymous reviewers for providing very helpful references for application problems addressed in this chapter. We thank Pat Sukprasert for pointing us to references on online bipartite matching.

---

**Algorithm 9.1** Fully Dynamic Algorithms with Predictions from Offline Divide-and-Conquer Algorithm (continued on next page)

---

**Require:** *Offline (during preprocessing):* partition-tree  $\mathcal{T}$ , divide-and-conquer algorithm  $\mathcal{A}$  that computes  $f(\cdot)$ , and predicted sequence of dynamic updates  $P$ . *Online:* Online sequence of dynamic updates  $U = [E_1, \dots, E_T]$  where each event  $E = (e, \text{type})$  is a *real* event.

**Ensure:** After each day  $t \in [T]$ , output  $f(U_t)$ .

```

1:  $\bar{P} \leftarrow \text{PREPROCESSPREDICTIONS}(P)$ .
2:   // Converts  $P$  into a feasible set of predictions
3: Insert all events in  $\bar{P}$  as predicted events on their corresponding days.
4:
5: procedure RETRIGGER( $t_1, t_2$ )
6:   Find  $W$ , the smallest window in  $\mathcal{T}$  that contains both  $t_1$  and  $t_2$ .
7:    $S \leftarrow \{W\}$ 
8:   // Set of windows to process
9:   while  $S \neq \emptyset$  do
10:    Remove a window  $W'$  from  $S$ .
11:    for each child  $C$  of  $W'$  do
12:      PROCESSEVENTS( $C$ ).
13:       $S \leftarrow S \cup \{C\}$ .
14:    end for
15:  end while
16: end procedure
17:
18: procedure PROCESSEVENTEARLIERTHANPREDICTION( $E, t, t_{\text{predict}}$ )
19:   Remove predicted event  $E$  from day  $t_{\text{predict}}$ .
20:   Add  $E$  to day  $t$  as a real event.
21:   RETRIGGER( $t, t_{\text{predict}}$ ).
22: end procedure
23:
24: procedure PROCESSEVENTLATERTHANPREDICTION( $E, i, t$ )
25:   Remove predicted event  $E$  from day  $t$ .
26:   Add  $E$  to day  $(t + 2^i)$  as a predicted event.
27:   Change the prediction for  $E = (e, T)$  in  $\bar{P}$  to  $(e, T, t + 2^i, i)$ .
28:   if  $e$  has a corresponding predicted deletion event that occurs earlier than  $(t + 2^i)$ 
29:   then
30:     Remove  $e$ 's predicted deletion event from the day  $t'$  that it was scheduled for.
31:     Add  $e$ 's predicted deletion event to day  $(t + 2^i)$ .
32:     Let  $(e, \text{deletion}, t', i_{\text{del}})$  be the entry for  $e$ 's deletion event in  $P$ .
33:     Change  $e$ 's deletion event prediction in  $\bar{P}$  to  $(e, \text{deletion}, (t + 2^i), i_{\text{del}} + 1)$ .
34:   end if
35:   RETRIGGER( $t, t + 2^i$ ).
36: end procedure

```

---



---

**Algorithm 9.2** Fully Dynamic Algorithms with Predictions from Offline Divide-and-Conquer Algorithm (continued from previous page)

---

```

36: for day  $t \in [T]$  do
37:   for each real event  $E = (e, T)$  on day  $t$  do
38:     Suppose event  $E$  is an event of type  $T$  on element  $e$ .
39:     Find the corresponding prediction  $(e, T, t_{\text{predict}}, i) \in \overline{P}$  for event  $E$ .
40:     if  $t < t_{\text{predict}}$  then
41:       PROCESSEVENTEARLIERTHANPREDICTION( $E, t, t_{\text{predict}}$ ).
42:     end if
43:   end for
44:   for each predicted event  $E = (e, T)$  on day  $t$  do
45:     if event  $E$  wasn't a real event on day  $t$  then
46:       PROCESSEVENTLATERTHANPREDICTION( $E, i + 1, t$ ).
47:     end if
48:   end for
49:   Return OUTPUT( $t$ ).
50: end for

```

---



---

**Algorithm 9.3** Backstop Meta-Algorithm for  $N$  Algorithms

---

```

1: Initialize algorithms  $\{A_1, \dots, A_N\}$ .
2: Initialize event buffers for algorithms  $\{A_1, \dots, A_N\}$ .
3:
4: for day  $t$ , event  $e_t$  occurs do
5:   // occurs online
6:   Add  $e_t$  to event buffers for  $\{A_1, \dots, A_N\}$ .
7:   Iteratively perform one computation step of each algorithm in  $\{A_1, \dots, A_N\}$  until one
   has completed the computation for all events in its buffer.
8:   Output the completed computation.
9: end for

```

---

---

**Algorithm 9.4** Work Bounds with High Probability

---

**Require:** *Offline (during preprocessing):* partition-tree  $\mathcal{T}$ , divide-and-conquer algorithm  $\mathcal{A}$  that computes  $f(\cdot)$ , predicted sequence of dynamic updates  $P$ , and ground set  $\mathcal{S}$  where each event is on an element  $e \in \mathcal{S}$ . *Online:* Online sequence of dynamic updates  $U = [E_1, \dots, E_T]$  where each event  $E = (e, \text{type})$  is a *real* event; prediction sets  $P_1, \dots, P_{\log_2(T)}$ .

**Ensure:** After each day  $t \in [T]$ , output  $f(U_t)$ .

```

1:  $\hat{T} \leftarrow 1$  // Initialization
2:  $L \leftarrow k \cdot \log(\hat{T})$  for some fixed constant  $k > 0$ .
3: for real event  $E_t$  do
4:   if  $t \geq \hat{T}$  then // Guess-and-double
5:     Obtain prediction set  $P_{\max(1, \log_2(\hat{T}))}$ .
6:      $\hat{T} \leftarrow 2 \cdot \hat{T}$ .
7:      $L \leftarrow \max(k \cdot \log(\hat{T}), \log(|\mathcal{S}|))$  for sufficiently large constant  $k > 0$ .
8:     Create  $\mathcal{A} := [A_1, \dots, A_L]$ ,  $L$  independent instantiations of Algorithm 9.1 using  $\hat{T}$ 
       and  $P_{\max(1, \log_2(\hat{T}))}$ .
9:     // Instantiations use independent sources of randomness.
10:     $M \leftarrow$  Initialize backstop meta-algorithm (Algorithm 9.3) over  $\mathcal{A}$ .
11:    Use  $M$  to process all previously seen events  $E_1, \dots, E_{t-1}$  in  $B$ .
12:  end if
13:  Add  $E_t$  to  $B$ .
14:  Pass  $E_t$  to  $M$ , return output of  $M$ .
15: end for

```

---

## Chapter 10

# Competitive Strategies to Use “Warm Start” Algorithms with Predictions

This chapter is based on *Competitive strategies to use “warm start” algorithms with predictions*, which is joint work with Avrim Blum, and appeared at the Symposium on Discrete Algorithms (SODA) 2025 [Blum and Srinivas, 2025].

### 10.1 Introduction

*Warm start* algorithms are a practically motivated paradigm of algorithms that make use of *predictions* to improve runtime. It is the most common framework in which *algorithms with predictions* (also *learning-augmented algorithms*, see Mitzenmacher and Vassilvitskii [2021]) are studied for static problems.<sup>1</sup> A warm start algorithm solves an instance  $I$  from an instance space  $\mathbf{I}$  to find a solution  $S$  in a solution space  $\mathbf{S}$ . In addition to  $I$ , the algorithm is

---

<sup>1</sup>By “static,” we mean problems for which we are given all inputs up front, and we wish to minimize runtime. This is as opposed to work in online, streaming, dynamic, and approximation algorithms.

given a prediction  $P \in \mathbf{S}$ , which can be thought of as a “guess” of the solution. The runtime of the algorithm depends on how far  $S$  is from  $P$ . This has applications in settings in which one must solve a series of related instances of a problem, and can learn information about “typical” solutions. An example is a network routing problem that must be solved daily, for which the user knows that “today’s network traffic is not too different than yesterday’s.”

In this setting, the solution space  $\mathbf{S}$  is a metric space with distance  $\mathbf{d}(\cdot, \cdot)$ , and the runtime of an algorithm on instance  $I$  with prediction  $P$  can be bounded by  $\mathbf{d}(P, S)$ , where  $S$  is the true solution of instance  $I$ . (We wrap factors that depend on the size of the instance and other details into the distance function  $\mathbf{d}$ .) Examples that fit in this framework include algorithms for *bipartite matching* [Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2021, Chen, Silwal, Vakilian, and Zhang, 2022] and *network flows* [Davies, Moseley, Vassilvitskii, and Wang, 2023]. Additionally, many algorithms that follow the paradigm of *local search* on the solution space can also be cast in this framework.

Given warm starts as an algorithmic primitive, the question remains how to obtain and use high quality predictions to achieve good performance. Previous work on this problem has focused on competing with the best fixed prediction in hindsight. In the distributional setting, Dinitz, Im, Lavastida, Moseley, and Vassilvitskii [2021] and Davies, Moseley, Vassilvitskii, and Wang [2023] show that it is possible to PAC-learn a good fixed prediction to use with future instances. In the online setting, Khodak, Balcan, Talwalkar, and Vassilvitskii [2022] show that it is possible to compete with the best fixed prediction in hindsight, and their work extends to algorithms with predictions in settings beyond warm starts.

These results demonstrate settings in which warm starts can give us an advantage. However, to see a large advantage, we must be in a setting where the best fixed prediction leads to much better performance than a generic default prediction. This means that the solutions

of most instances coming from our distribution must be very close to the prediction, and therefore very similar to each other, which is a restrictive assumption. Our work explores less restrictive structure that we can take advantage of to achieve good performance.

We observe that warm start algorithms have many properties that we can take advantage of, that make them powerful and flexible. They are packaged local search routines that allow you to search the solution space by conceptually growing a ball around a prediction, where the radius of a ball corresponds to the runtime of the warm start algorithm. It is also possible to run multiple instantiations of the algorithm in parallel, and simultaneously search the space from multiple points. Another useful property of warm starts is that every time we solve an instance, we learn the optimal prediction for that instance, i.e. the solution. These observations motivate our main question:

**Question 10.1.1.** *Are there settings in which we can use warm-start algorithms to achieve significantly better performance than any single fixed prediction?*

In this work, we leverage these properties of warm starts to compete against a variety of stronger baselines, answering this question in the affirmative. In particular, we consider competing with a set of  $k$  predictions  $\mathbf{P} \in \mathbf{S}^k$ . That is, the cost of  $\mathbf{P}$  with respect to an instance  $I$  with solution  $S$  is the distance from  $S$  to the closest prediction:  $\min_{P \in \mathbf{P}} \mathbf{d}(S, P)$ . This is analogous to the  $k$ -medians objective. We also consider competing with the best function  $h$  from some given class, mapping each instance to one of  $k$  predictions.

It is somewhat surprising that it is possible to achieve any non-trivial guarantees against multiple points, especially in the online setting. Previous work approaches this problem by reducing to online convex optimization [Khodak, Balcan, Talwalkar, and Vassilvitskii \[2022\]](#). We note that in the standard online convex optimization setting, we cannot expect to compete with a collection of multiple points and achieve vanishing regret. In particular,

even going from 1 trajectory to 2 trajectories makes the offline problem of choosing the best trajectories go from being convex to being non-convex. We observe that vanishing regret is actually a stronger property than what we need. By accruing constant factors in runtime, and taking advantage of structural properties such as running algorithms in parallel, we show that we can compete with far stronger baselines.

We note that there is a line of work that studies competing against multiple predictions for algorithms with predictions for *online* problems (see e.g. [Anand, Ge, Kumar, and Panigrahi, 2022, Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2022, Antoniadis, Coester, Elias, Polak, and Simon, 2023]), which has a different set of challenges and techniques from the warm start setting, which are discussed more in Section 10.1.1. Our work is, to the best of our knowledge, the first to consider this problem for warm starts.

**Competing against multiple points offline.** Our first setting is simple, but we find it beneficial to provide a formal treatment as it motivates our other two settings. In this setting, we consider instance-solution pairs drawn from a distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ . Here, we can use a simple strategy that learns  $k$  fixed predictions for  $\mathcal{D}$  from samples. Then, for a subsequent instance  $(I, S) \sim \mathcal{D}$ , for each prediction we can run an instantiation of our warm start algorithm “in parallel,”<sup>2</sup> and output the solution of the thread that completes first.

**Informal Theorem 10.1.2** (Competing against  $k$  fixed points offline, Formally Theorem 10.3.5). *For a distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$  it is possible to learn a set of  $k$  predictions  $\hat{\mathbf{P}}$  that is an  $O(1)$  approximation to*

$$\operatorname{argmin}_{\mathbf{P} \in \mathbf{S}^k} \mathbb{E}_{(I, S) \sim \mathcal{D}} \left[ \min_{P \in \mathbf{P}} \mathbf{d}(S, P) \right].$$

---

<sup>2</sup>By “in parallel,” we refer to interleaving the  $k$  threads of computation, resulting in a sequential algorithm.

Furthermore, given a set of predictions  $\hat{\mathbf{P}}$ , we can design a procedure such that the time to solve a new instance  $I \in \mathbf{I}$  with (unknown) true solution  $S \in \mathbf{S}$  is

$$O(k) \cdot \min_{P \in \hat{\mathbf{P}}} \mathbf{d}(S, P).$$

We note that a  $\Omega(k)$  approximation factor in the runtime is necessary without assuming more structure in the problem ([Remark 10.3.6](#)).

**Learning “coarse information.”** This motivates our second setting, in which we avoid this extra  $\Omega(k)$  factor in the runtime. Suppose we had access to extra “coarse information” about the mapping from instances and solutions. A warm start algorithm, or indeed any algorithm, already encodes the perfect mapping between instances and solutions. However, this comes at a high cost to uncover. We model coarse information as a  $k$ -wise partition of the instance space  $h : \mathbf{I} \rightarrow [k]$ , where we expect that two instances  $I_1$  and  $I_2$  that map to the same partition have similar solutions. In addition, we expect  $h(I)$  to be significantly faster to compute than solving the instance  $I$ . Given an  $h$  that satisfies these properties, we can learn a fixed prediction  $P^{(i)}$  for each partition  $i$  of  $h$  with respect to a distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ . Then, for a new instance  $(I, S) \sim \mathcal{D}$ , we can use the single prediction  $P^{h(I)}$ . This allows us to potentially avoid the  $O(k)$  factor in the runtime of the previous theorem.

Thus the question is whether we can learn a partition  $h$  of  $\mathbf{I}$  that is informative and well-aligned with  $\mathcal{D}$ . We show that given a class of efficiently-computable candidate partitions  $\mathcal{H}$ , it is possible to learn an  $h \in \mathcal{H}$  that is approximately best-aligned with the distribution  $\mathcal{D}$ .

**Informal Theorem 10.1.3** (Competing with a hypothesis class of  $k$ -wise partitions offline, Formally [Theorem 10.4.10](#)). *For a distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ , and a learnable hypothesis*

class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{I}$ , given access to an appropriate ERM oracle over  $\mathcal{H}$ , it is possible to learn an  $h : \mathbf{I} \rightarrow [k]$  and set of predictions  $\mathbf{P} = (P^{(1)}, \dots, P^{(k)})$  that are an  $O(1)$  approximation to

$$\min_{h \in \mathcal{H}} \min_{\mathbf{P} \in \mathbf{S}^k} \mathbb{E}_{(I, S) \sim \mathcal{D}} [\mathbf{d}(S, P^{(h(I))})].$$

Furthermore, given an  $h : \mathbf{I} \rightarrow [k]$  and  $\mathbf{P} = (P^{(1)}, \dots, P^{(k)})$ , we can design a procedure such that the expected time to solve a new instance is

$$\mathbb{E}_{(I, S) \sim \mathcal{D}} [\text{runtime}(h(I)) + \mathbf{d}(S, P^{(h(I))})].$$

We achieve this via a two step procedure, that first learns an approximately good set of  $k$  centers for  $\mathcal{D}$ , then uses an ERM oracle to find an  $h \in \mathcal{H}$  that aligns well with this set of centers. While there may be no reason to expect that the  $k$  centers chosen in the first step are similar to the best centers for the best  $h \in \mathcal{H}$ , we show that this strategy is still approximately optimal.

**Competitive algorithms to solve sequences of instances.** Finally, we consider an online setting where instances arrive sequentially. One of the main motivations of warm start algorithms is settings in which instances that arrive close in time are likely to be similar. One example is the setting explored in experiments by [Davies, Moseley, Vassilvitskii, and Wang \[2023\]](#), in which they use warm starts to speed up solving a series of flow instances that correspond to image segmentation tasks on frames of a video.

We formulate the *online ball search* problem to model this setting. In this problem, on each day there is an hidden point  $S \in \mathbf{S}$  (corresponding to the true solution of that day's instance) that the algorithm must find. The algorithm is allowed to search the space  $\mathbf{S}$  by growing



balls around points of its choosing in  $\mathbf{S}$ . The algorithm succeeds when one of the balls first contains  $S$ . Each of these balls corresponds to running an instance of an algorithm  $\mathcal{A}(I, P)$  using a particular prediction  $P$ . The radius of the ball corresponds to the number of steps for which the algorithm was run.

In this setting, we design competitive algorithms that compete against the best strategy in hindsight. The strategies that we compete against are collections of  $k$  predictions that are allowed to move over time. We call these moving predictions *trajectories*. The cost of a collection of  $k$  trajectories is the total *hit cost* plus the total *movement cost*. The hit cost on a given day is the distance from that day's solution to the closest of the trajectories on that day. The total movement cost is the total distance that each of the  $k$  trajectories moves over time. This is a generalization of competing with  $k$  fixed points, that includes adaptive strategies.

Note that while the baseline strategies that we compete against are constrained to using  $k$  predictions, and must pay to move the predictions, the online algorithm has no constraint on how many predictions it can use (i.e., points from which it can search), nor does it pay to move predictions. We note that the restriction on the offline strategies is necessary, because if the offline strategy was allowed to move arbitrarily, it would move perfectly to the solution on each day and achieve cost 0, making it impossible to compete with.

Our objective is to minimize the total runtime to solve a sequence of instances. The runtime to solve the instances corresponds to the sum of radii that the algorithm searches, which is the number of steps it runs of the subroutine  $\mathcal{A}$ , plus any additional overhead in running the algorithm.

**Informal Theorem 10.1.4** (Competing with trajectories online efficiently, Formally [Theorem 10.5.16](#)). *We have an algorithm for online ball search that is  $O(k^4 \ln^2 k)$ -competitive*

*with any set of  $k$  trajectories, where the total runtime of the algorithm is bounded by  $O(1)$  times the number of steps it runs of the algorithms-with-predictions subroutine. That is, this algorithm does at most  $O(k^4 \ln^2 k)$  times as much work as any set of  $k$  trajectories.*

*Furthermore, the algorithm is deterministic and therefore robust to an adaptive adversary, and oblivious to the setting of  $k$ , so the guarantee holds for all  $k$  simultaneously.*

This guarantee states that our online algorithm has runtime comparable, within an  $O(k^4 \ln^2 k)$  factor, to any offline strategy that is allowed to maintain  $k$  moving predictions, and pay the cost on each day from that day's solution to the *closest* of the predictions. In some sense, this algorithm does well when the solutions to the instances we see fall into  $k$  (moving) clusters, and our objective is related to the  $k$ -medians objective. Thus, it is particularly nice that the algorithm does not need to know  $k$ , as in practice we often do not know the number of clusters in our data in advance.

To approach the online problem, it is illustrative to first consider competing with only one trajectory. In this setting, an algorithm that always uses the previous day's solution as the next day's prediction is 2-competitive (via the triangle inequality). Note that this simple guarantee already generalizes what was previously known, as it competes not only with the best fixed prediction in hindsight, but also against adaptive strategies that move over time. For multiple trajectories, this simple strategy is no longer enough, as the previous day's solution could come from some other trajectory, and be arbitrarily far away from today's solution.

Our algorithm for multiple trajectories addresses this by searching from *all* previous solutions in parallel. These “threads” are run at quadratically decaying rates, i.e. the thread of the  $i$ th most recent solution is run at rate  $\frac{1}{i^2 \ln^2 i}$ . This leaves the issue that the previous solution from the same trajectory as today could be arbitrarily far in the past, and be run at an

extremely slow rate. This is addressed by “pruning” threads that are no longer fruitful. That is, when the ball searched around a solution  $i$  fully contains the ball searched around a previous solution  $j$ , the algorithm can stop running thread  $j$ , as that work is redundant with thread  $i$ . Then, we can increase the rates of the threads that are lower priority than  $j$ , and maintain that there is at most one thread being run at each rate  $\frac{1}{i^2 \ln^2 i}$  for each integer  $i$ . In the analysis we show that either the algorithm runs long enough for the previous solution from the same trajectory as today to be elevated to rate  $\geq \frac{1}{k^2 \ln^2 k}$  and eventually solve the instance, or the algorithm terminates more quickly than that, which is even better. This allows us to bound the competitive ratio.

To bound the runtime, we note that the amount of work done by each thread is actually dominated by the work done to check if it was been pruned yet. This blows up the work done by the  $i$ th fastest thread by a factor  $O(i)$ . Since the rate at which the  $i$ th thread takes steps of  $\mathcal{A}$  is  $\frac{1}{i^2 \ln^2 i}$ , the rate at which it does work is  $O(i) \cdot \frac{1}{i^2 \ln^2 i} = O(\frac{1}{i \ln^2 i})$ . Thus, summing the work done over all of the threads results in a convergent series, and we are able to bound the total work done by the algorithm by the work done by the fastest thread.

We remark that if we relax our objective to only be competitive in sum of radii searched, and not worry about the runtime of the algorithm, the problem is still interesting. This could correspond to settings where steps of the subroutine are much more costly than steps of the online scheduling algorithm. In this setting, we can design an algorithm with an improved competitive ratio of  $O(k^2)$  based on a reduction to the  $k$ -server problem ([Corollary 10.5.7](#)).

In this work, we demonstrate that warm starts are a powerful algorithmic primitive that can, in many settings, be used in ways that are much stronger than competing with one fixed point in hindsight. This opens new research directions in more effective ways to learn and use warm starts. We also observe that the properties of warm starts that we use may hold

in many settings that use local-search type algorithms, and we hope that our techniques can be extended to broad-ranging applications.

### 10.1.1 Related work

**Algorithms with predictions.** We refer the reader to [Section 8.2](#) for a primer on *algorithms with predictions*.

**Data-driven algorithms.** We refer the reader to [Section 8.3](#) for an introduction to *data-driven algorithm design*. Data-driven algorithms are often studied in online settings where one must learn a good parameter setting over time. This is usually studied in the framework of regret, where an algorithm must compete with the best fixed parameter setting in hindsight. There is a line of work that considers “mixture” settings, in which the performance of the algorithm competes against multiple points in hindsight. [Sharma, Balcan, and Dick \[2020\]](#) give guarantees for *shifting regret*, where the algorithm must compete against offline strategies that can switch parameter settings some fixed number of times. [Balcan, Khodak, Sharma, and Talwalkar \[2021\]](#) study *meta-learning*, in which an algorithm sees samples from multiple distributions one at a time, and uses information from previous distributions to learn good initializations for future distributions. These settings are different from ours, as the set of points that they compete with are considered sequentially, whereas we compete with a  $k$ -medians style objective. [Khodak, Chow, Balcan, and Talwalkar \[2024\]](#) use a *contextual bandit* framework to choose different parameter settings for different types of instances. This is related to the setting that we study in [Section 10.4](#), in that we are learning information about the instance space, to provide parameter settings that are more fit to each instance.

**Warm starts.** *Warm starts* are a popular heuristic that are used in practice to speed up the computation of various problems, e.g. for linear and mixed integer programs [Gurobi Optimization, LLC, 2023]. Due to their success in practice, a line of work has sought to provide rigorous theoretical guarantees for problems such as bipartite matching [Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2021, Chen, Silwal, Vakilian, and Zhang, 2022] and max flow [Davies, Moseley, Vassilvitskii, and Wang, 2023]. Previous work has studied the problem of solving a sequence of instances using a learning-augmented algorithm to compete with the performance of the best fixed prediction in hindsight [Khodak, Balcan, Talwalkar, and Vassilvitskii, 2022]. Our work is, to the best of our knowledge, the first to consider the problem of competing with multiple predictions in the warm start setting.

**Dynamic algorithms.** We note that the online setting of our problem, in which we solve a sequence of related instances, is related to the setting of *dynamic algorithms*. In dynamic algorithms, the input can change in small, structured ways from one day to the next (e.g. for a graph, one edge is inserted or deleted per day). A dynamic algorithm must update its solution based on the change in the input, and seeks to minimize its *update time*, or runtime to perform this update on a given day. In dynamic algorithms, we typically hope to achieve update time that is sublinear, or even much smaller, in the size of the input. This justifies that a dynamic algorithm is much more efficient than solving each day’s instance from scratch.

In the warm start setting on the other hand, the input instance can change arbitrarily from day to day. Thus, we cannot hope for sublinear update times, as the algorithm must at least read the input and verify the predicted solution on each day. However, as we show in this work, in this less structured setting, it is possible to compete against stronger baselines. For example, consider a setting where on each day we receive an input corresponding to one of

several slowly changing graphs. This falls outside the standard dynamic model, but can be addressed by our guarantee that competes with multiple trajectories. There is also a line of work that considers *dynamic algorithms with predictions* including [Liu and Srinivas, 2024] which is the focus of Chapter 9, but this setting is quite different than the one we consider in this chapter.

**Multiple predictions for online algorithms.** There is a line of work that studies competing with multiple predictions for *online algorithms with predictions*. Online algorithms usually have some notion of commitment, where on each day the algorithm must make an irrevocable decision that affects future performance. Typically, algorithms with predictions in this setting solicit information about future events. Given multiple such predictions, the challenge is combining them into *one* decision that can be made on a given time step. Because the decisions that the algorithm makes vary from problem to problem, techniques to combine predictions often have to be problem specific. Work in this area includes strategies for scheduling problems [Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2022], set cover and facility location [Anand, Ge, Kumar, and Panigrahi, 2022], and metric algorithms [Antoniadis, Coester, Elias, Polak, and Simon, 2023]. This setting is significantly different from ours. In our setting, we can asymptotically compete with multiple predictions by running multiple instantiations of the learning-augmented algorithm in parallel. This leads to a different set of strategies and techniques than in the online setting.

**Online search and server problems.** The *online ball search* problem that we formulate in this work has connections to other well-studied online problems. Previous work has approached this problem through the lens of *online convex optimization* [Khodak, Balcan, Talwalkar, and Vassilvitskii, 2022], to provide competitive guarantees against a single fixed point in hindsight. We note that in going from competing against one trajectory to competing

against multiple trajectories, our problem becomes non-convex, and we require a different set of tools to approach it.

For competing against  $k$ -trajectories, the closest connection is to the  $k$ -server problem [Manasse, McGeoch, and Sleator, 1988, 1990], in which we must service a sequence of *requests* in a metric space, using  $k$  *servers*, while moving the servers as little as possible over the course of the algorithm. In our setting, we can think predictions as servers, and the solutions of arriving instances as being requests, a connection that we explore in Section 10.5.2. While we can solve online ball search with a reduction to the  $k$ -server problem, it is not clear if they are equivalent problems, and another approach we propose for online ball search (Section 10.5.3) does not go through this connection.

A related online search problem is the *oil searching problem* [McGregor, Onak, and Panigrahy, 2009], in which an algorithm can put in work to search  $n$  locations to certain depths to try to find a resource. This is related to our problem, in which the algorithm can put in work at various predictions to try to find a solution near that prediction. However, our problem allows the algorithm more freedom in choosing the starting location, and has a different objective function. Other related search problems are the *cow path problem* [Kao, Reif, and Tate, 1993], and the problem of *searching in the plane* [Baeza-Yates, Culberson, and Rawlins, 1993]. In these problems the algorithm is allowed to traverse the space in different ways than in online ball search.

The objective of the online ball search problem is analogous to the  $k$ -medians objective function in clustering, in some sense. Thus, the online problem also has connections to online and dynamic  $k$ -medians clustering [Bhattacharya, Costa, Lattanzi, and Parotsidis, 2023], and *online steiner tree* [Imase and Waxman, 1991, Alon and Azar, 1992, Gu, Gupta, and Kumar, 2016] though the objectives of these problems are somewhat different from ours.

We note that the online steiner tree problem has also been studied in the learning-augmented setting [Xu and Moseley, 2021], a setting which is quite different from this work.

**Solving sequences of correlated instances.** Another related line of work studies the problem of solving *correlated* instances of a problem. Dinur, Goldwasser, and Lin [2015] show that for certain NP-hard *constraint satisfaction problems*, it is possible to get improved algorithms when solving a series of instances where the solutions are small perturbations of each other. Goldwasser and Holden [2017] study a similar model, for problems in  $\mathbf{P}$ , where they establish fine-grained complexity bounds. This line of work is spiritually related to our problem, but is a significantly different model from the one that we study.

## 10.2 Preliminaries

We model an algorithm as solving instances from an instance space  $\mathbf{I}$  to produce solutions from a solution space  $\mathbf{S}$ . That is, a (standard) algorithm  $\mathcal{A}$  takes  $I \in \mathbf{I}$  as input to produce  $\mathcal{A}(I) = S \in \mathbf{S}$ . To model a warm start algorithm, we assume a metric distance  $\mathbf{d}$  on the solution space  $\mathbf{S}$ .

**Definition 10.2.1** (Warm start algorithm). *A warm start algorithm  $\mathcal{A}$  is one that takes a problem instance  $I \in \mathbf{I}$  and a predicted solution  $P \in \mathbf{S}$  as input.  $\mathcal{A}(I, P)$  outputs the true solution  $S \in \mathbf{S}$  of instance  $I$  in time  $\leq \mathbf{d}(P, S)$ .*

To justify modelling the runtime of an algorithm with predictions as a metric distance over the solution space  $\mathbf{S}$ , consider the following examples. For bipartite matching, Dinitz, Im, Lavastida, Moseley, and Vassilvitskii [2021] give an algorithm with runtime  $\tilde{O}(m\sqrt{n} \cdot (1 + \|y^* - \hat{y}\|_1))$ , where  $\hat{y}$  is the predicted (dual) solution, and  $y^*$  is the optimal (dual) solution. Chen, Silwal, Vakilian, and Zhang [2022] improve the guarantee to  $O(m\sqrt{n} + (m + n \log n)\|y^* - \hat{y}\|_0)$ . Both of these runtimes can be interpreted as a metric



distance that is essentially the distance between the true solution and the predicted solution in the relevant norm, scaled by a factor that depends on the input size. Another example is the algorithm of [Davies, Moseley, Vassilvitskii, and Wang \[2023\]](#) which solves instances of max-flow in time  $O(|E| \cdot (1 + \|\hat{f} - f^*\|_1))$ , where  $|E|$  is the number of edges in the flow network,  $\hat{f}$  is the predicted flow (as a vector), and  $f^*$  is the optimal flow closest to  $\hat{f}$ .

Note that we consider copies of the same solution to be distinct entities at some fixed distance from each other that depends on the input size. This is because even when the predicted solution is exactly the true solution, a warm-start algorithm will still take some amount of time to verify the solution. This is reflected in the time-bounds that are given above, which are lower bounded by a constant that depends on the input size. This is consistent with our characterization of the runtime as a metric on the solution space. For example, in many settings, we can take the distance between a prediction  $P$  and a solution  $S$  to be

$$\mathbf{d}(P, S) = |I| \cdot (1 + \|P - S\|)$$

where  $|I|$  is the size of the relevant instances, and  $\|P - S\|$  is the distance between  $P$  and  $S$  as vectors taken in the relevant norm.

In the remainder of the chapter, we will assume that the runtime of a warm start algorithm corresponds to some metric distance  $\mathbf{d}$  that is known and easily computable. In particular, we make the following assumption.

**Assumption 10.2.2** (Metric distance is easily computable). *It is possible to compute the distance,  $\mathbf{d}(S_1, S_2)$ , between any two points,  $S_1, S_2 \in \mathbf{S}$ , in time at most  $\mathbf{d}_{\min}$ , where  $\mathbf{d}_{\min}$  is the minimum distance in  $\mathbf{S}$ . Recall that the distance  $\mathbf{S}$  is lower bounded, as we consider two copies of the same point to be distinct entities at some fixed distance away from each other.*

*For ease of notation, we will consider the distance  $\mathbf{d}$  to be scaled by a factor such that  $\mathbf{d}_{\min} = 1$  and the time to compute the distance between points is  $O(1)$ . This is without loss of generality, as we give multiplicative runtime guarantees in terms of  $\mathbf{d}$ .*

This assumption is reasonable, because in our applications of interest, the points in  $\mathbf{S}$  are represented by vectors of dimension that scales with the input size, and the distance function is distance taken in an appropriate norm. The runtime of the warm-start algorithms also has a multiplicative factor that scales with the input size, so the time to calculate the distance is on the same order of magnitude as the time to perform “one iteration” of the warm-start algorithm.

### 10.3 Competing Against $k$ Fixed Points Offline

As a warm up and motivating example, we consider a distribution  $\mathcal{D}$  over instance-solution pairs  $\mathbf{I} \times \mathbf{S}$ . We assume that there is a underlying ground-truth mapping from instances  $I \in \mathbf{I}$  to solutions  $S \in \mathbf{S}$  that is uncovered by our algorithm-with-predictions  $\mathcal{A}$ . Thus it is sufficient to assume a distribution only over  $\mathbf{I}$ , and we include the solution in the distribution for ease of notation. In many places it will be useful to consider the marginal distribution of solutions.

We show that, with access to i.i.d. samples from  $\mathcal{D}$ , a simple strategy of running the predictions in parallel can compete with the best  $k$  fixed predictions for the distribution, with an  $O(k)$  approximation factor. This is a generalization of the results in [Dinitz, Im, Lavastida, Moseley, and Vassilvitskii, 2021] and [Davies, Moseley, Vassilvitskii, and Wang, 2023], which show that it is possible to PAC learn the single best fixed prediction with respect to  $\mathcal{D}$ .

Our main observation is that a warm start algorithm-with-predictions allows us to run mul-

**Algorithm 10.1** Using  $k$  Predictions in Parallel

- 
- 1: Run  $\mathcal{A}(I, P_j)$  for all  $j \in [k]$  “in parallel”<sup>3</sup> until one of them completes
  - 2: Output the solution of the  $\mathcal{A}(I, P_j)$  that completed
- 

multiple threads in parallel, allowing us to compete with the performance of the best thread.

**Lemma 10.3.1** (Using  $k$  predictions). *Given an algorithm-with-predictions  $\mathcal{A}$  for instances in  $\mathbf{I}$ , and a set of  $k$  predictions  $\mathbf{P} = (P_1, \dots, P_k), P_i \in \mathbf{S}$ , for an instance  $I$  with (unknown) true solution  $S$ , we can solve  $I$  in time*

$$O(k \cdot \mathbf{d}(S, \mathbf{P}(S))),$$

where  $\mathbf{P}(S) = \operatorname{argmin}_{P_j \in \mathbf{P}} \mathbf{d}(S, P_j)$ .

*Proof.* We run the algorithm-with-predictions in parallel with each of the  $k$  predictions, and output the solution of the thread that completes first (Algorithm 10.1). Let  $\mathbf{P}(S) = P_{j^*}$  be the prediction that minimizes  $\mathbf{d}(S, P_{j^*})$ . We know that the time that Algorithm 10.1 spends on thread  $j^*$ , i.e. the runtime of  $\mathcal{A}(I, P_{j^*})$ , is  $O(\mathbf{d}(S, P_{j^*}))$ . Since the threads are run in parallel at the same rate, this means that the time that Algorithm 10.1 spends on any thread  $j$  is  $O(\mathbf{d}(S, P_{j^*}))$ , and therefore the total runtime of the algorithm is bounded by  $O(k \cdot \mathbf{d}(S, P_{j^*}))$ .  $\square$

We observe that it is possible to learn an approximately good set of predictions with respect to a distribution  $\mathcal{D}$  over  $\mathbf{S}$  from samples, by noting that this is the  $k$ -medians clustering problem for the distribution  $\mathcal{D}$ . We provide a standard sample compression argument that assumes little structure on  $\mathbf{S}$  for completeness, and note that better bounds are known for specific settings of interest.

---

<sup>3</sup>Here, we use “in parallel” to mean alternately running a constant number of steps of each of the  $\mathcal{A}(I, P_j)$ , resulting in a sequential algorithm.

**Definition 10.3.2** (Clustering cost of a set). Let  $\mathbf{X} = \{X_1, \dots, X_m\}$  be a set of  $m$  points from  $\mathbf{S}$ , and  $\mathbf{C} \in \mathbf{S}^k$  be an arbitrary set of  $k$  centers. We define the cost of  $\mathbf{C}$  over  $\mathbf{X}$  by

$$\text{cost}(\mathbf{C}, \mathbf{X}) = \frac{1}{m} \sum_{i=1}^m [\mathbf{d}(\mathbf{C}(X_i), X_i)],$$

where  $\mathbf{C}(X_i)$  is the closest center in  $\mathbf{C}$  to  $X_i$ .

**Definition 10.3.3** (Clustering cost of a distribution). Let  $\mathcal{D}$  be a distribution over  $\mathbf{I} \times \mathbf{S}$ , and  $\mathbf{C} \in \mathbf{S}^k$  be an arbitrary set of  $k$  centers. We define the cost of  $\mathbf{C}$  over  $\mathcal{D}$  by

$$\text{cost}(\mathbf{C}, \mathcal{D}) = \mathbb{E}_{(I, S) \sim \mathcal{D}} [\mathbf{d}(\mathbf{C}(S), S)],$$

where  $\mathbf{C}(S)$  is the closest center in  $\mathbf{C}$  to  $S$ .

**Lemma 10.3.4** (Learning  $k$  fixed points). For a distribution  $\mathcal{D}$  over instance-solution pairs, it is possible to learn an  $O(1)$ -approximate  $k$ -medians clustering of  $\mathcal{D}$  from  $m \geq \frac{12 \cdot \mathbf{d}_{\max} \cdot k \cdot \log(1/\delta)}{\text{cost}(\mathbf{C}^*, \mathcal{D})}$  samples drawn i.i.d. from  $\mathcal{D}$ , with probability  $\geq 1 - 2\delta$ , where  $\mathbf{d}_{\max}$  is the width of  $\mathbf{S}$ .

*Proof.* Let  $\mathbf{C}^* = (\mathbf{C}_{(1)}^*, \dots, \mathbf{C}_{(k)}^*)$  be a best set of  $k$  centers for distribution  $\mathcal{D}$ . Let  $\mathbf{d}_{\max}$  be the width (largest distance) of the metric space  $\mathbf{S}$ . Consider  $\mathbf{X} = \{X_1, \dots, X_m\}$ , a set of  $m$  samples drawn i.i.d. from  $\mathcal{D}$ . First, we bound the probability that the average loss of  $\mathbf{C}^*$  over  $\mathbf{X}$  is far from the average loss of  $\mathbf{C}^*$  over  $\mathcal{D}$ . Let

$$\text{cost}(\mathbf{C}^*, \mathcal{D}) = \mathbb{E}_{X \sim \mathcal{D}} [\mathbf{d}(\mathbf{C}^*(X), X)],$$

where  $\mathbf{C}^*(X)$  is the closest  $C \in \mathbf{C}^*$  to  $X$ . Similarly, define the empirical cost of  $\mathbf{C}^*$  over  $\mathbf{X}$

as

$$\text{cost}(\mathbf{C}^*, \mathbf{X}) = \frac{1}{m} \sum_{i=1}^m \mathbf{d}(\mathbf{C}^*(X_i), X_i).$$

Now we can use a Chernoff-Hoeffding bound to see that

$$\mathbb{P}[\text{cost}(\mathbf{C}^*, X) \geq 2 \cdot \text{cost}(\mathbf{C}^*, \mathcal{D})] = e^{-\frac{m \cdot \text{cost}(\mathbf{C}^*, \mathcal{D})}{3\mathbf{d}_{\max}}}.$$

Thus, by setting  $m \geq \log(1/\delta) \cdot \frac{3\mathbf{d}_{\max}}{\text{cost}(\mathbf{C}^*, \mathcal{D})}$ , for some  $0 < \delta < 1$  we can achieve that the empirical loss of  $\mathbf{C}^*$  over the samples  $X$  is at most twice the true loss over the distribution, with probability  $\geq 1 - \delta$ .

Now, we show that it is possible to learn an approximately good clustering for  $\mathcal{D}$  from  $\mathbf{X}$ . Our learning algorithm proceeds as follows. It considers all  $\binom{m}{k}$  subsets of  $\mathbf{X}$  as possible centers. Of the possible centers, it chooses the  $\hat{\mathbf{C}}$  with the lowest empirical loss.

By a sample compression argument, see e.g. Theorem 30.2 in [Shalev-Shwartz and Ben-David, 2014], we have that with probability at least  $1 - \delta$

$$\text{cost}(\hat{\mathbf{C}}, \mathcal{D}) \leq \text{cost}(\hat{\mathbf{C}}, \mathbf{X} \setminus \hat{\mathbf{C}}) + \sqrt{\text{cost}(\hat{\mathbf{C}}, \mathbf{X} \setminus \hat{\mathbf{C}}) \frac{4k \log(m/\delta)}{m}} + \frac{8k \log(m/\delta)}{m}.$$

Setting  $m$  to be sufficiently larger than  $4k \log(1/\delta) \mathbf{d}_{\max}$  allows us to conclude that

$$\text{cost}(\hat{\mathbf{C}}, \mathcal{D}) \leq 2 \cdot \text{cost}(\hat{\mathbf{C}}, \mathbf{X} \setminus \hat{\mathbf{C}}).$$

Now, we note that there is a clustering of  $\mathbf{X}$  that uses points from  $\mathbf{X}$  as centers, that has cost at most 2 times that of  $\mathbf{C}^*$  on  $\mathbf{X}$ . To see this, consider mapping each center in  $\mathbf{C}^*$  to

its nearest point in  $\mathbf{X}$ , to get  $\mathbf{C}'$ ,

$$\mathbf{C}'_{(i)} = \operatorname{argmin}_{X \in \mathbf{X}} \mathbf{d}(\mathbf{C}^*_{(i)}, X).$$

We have that for each  $X \in \mathbf{X}$

$$\begin{aligned} \mathbf{d}(\mathbf{C}'(X), X) &\leq \mathbf{d}(\mathbf{C}^*(X), X) + \min_{X' \in \mathbf{X}} \mathbf{d}(\mathbf{C}^*(X), X') \\ &\leq 2 \cdot \mathbf{d}(\mathbf{C}^*(X), X). \end{aligned}$$

Since  $\widehat{\mathbf{C}}$  is the cost minimizer over all sets of centers that are subsets of  $\mathbf{X}$ , this means that  $\operatorname{cost}(\widehat{\mathbf{C}}, \mathbf{X}) \leq 2 \cdot \operatorname{cost}(\mathbf{C}^*, \mathbf{X})$ . Furthermore, since  $2k \leq m$ , we have that  $\operatorname{cost}(\widehat{\mathbf{C}}, \mathbf{X} \setminus \widehat{\mathbf{C}}) \leq 2 \cdot \operatorname{cost}(\widehat{\mathbf{C}}, \mathbf{X})$ . Finally, we get that if  $m \geq \frac{12 \cdot \mathbf{d}_{\max} \cdot k \cdot \log(1/\delta)}{\operatorname{cost}(\mathbf{C}^*, \mathcal{D})}$ , then with probability  $\geq 1 - 2\delta$

$$\begin{aligned} \operatorname{cost}(\widehat{\mathbf{C}}, \mathcal{D}) &\leq 2 \cdot \operatorname{cost}(\widehat{\mathbf{C}}, \mathbf{X} \setminus \widehat{\mathbf{C}}) \\ &\leq 4 \cdot \operatorname{cost}(\widehat{\mathbf{C}}, \mathbf{X}) \\ &\leq 8 \cdot \operatorname{cost}(\mathbf{C}^*, \mathbf{X}) \\ &\leq 16 \cdot \operatorname{cost}(\mathbf{C}^*, \mathcal{D}), \end{aligned}$$

so  $\widehat{\mathbf{C}}$  is an  $O(1)$ -approximation to the minimum cost  $k$ -clustering of  $\mathcal{D}$ . □

Using the above lemmas, we can conclude the following theorem.

**Theorem 10.3.5** (Competing against  $k$  fixed points offline). *With access to  $m \geq \frac{12 \cdot \mathbf{d}_{\max} \cdot k \cdot \log(1/\delta)}{\operatorname{cost}(\mathbf{C}^*, \mathcal{D})}$  i.i.d. samples from  $\mathcal{D}$ , it is possible to design an algorithm that has expected runtime*

$$O(k) \cdot \operatorname{cost}(\mathbf{C}^*, \mathcal{D})$$

*on future instances drawn from  $\mathcal{D}$ , where  $\mathbf{C}^*$  is the set of  $k$  centers in  $\mathbf{S}$  that minimizes the*

cost with respect to  $\mathcal{D}$ .

*Proof.* [Lemma 10.3.4](#) tells us that from  $m \geq \frac{12 \cdot \mathbf{d}_{\max} \cdot k \cdot \log(1/\delta)}{\text{cost}(\mathbf{C}^*, \mathcal{D})}$  samples, with probability  $\geq 1 - 2\delta$  we can learn a set of centers  $\hat{\mathbf{C}}$  such that

$$\text{cost}(\hat{\mathbf{C}}, \mathcal{D}) = O(1) \cdot \text{cost}(\mathbf{C}^*, \mathcal{D}).$$

Once we have  $\hat{\mathbf{C}}$ , on a subsequent instance-solution pair  $(I, S) \sim \mathcal{D}$ , we can use the algorithm from [Lemma 10.3.1](#) to solve  $I$  in expected time

$$\begin{aligned} \mathbb{E}_{(I, S) \sim \mathcal{D}} \left[ O(k \cdot \mathbf{d}(S, \hat{\mathbf{C}}(S))) \right] &= O(k) \cdot \text{cost}(\hat{\mathbf{C}}, \mathcal{D}) \\ &= O(k) \cdot \text{cost}(\mathbf{C}^*, \mathcal{D}). \end{aligned}$$

□

*Remark 10.3.6.* In the generality that we have modeled the problem in this section, we cannot hope to achieve an approximation factor that is  $o(k)$ . Consider the following bad example. There are  $k$  planted solutions that are arbitrarily far apart. Our distribution serves instances such that there is a uniform probability that any of these planted solutions is the true solution. Thus, an algorithm must either explore a constant fraction of the  $k$  planted solutions in expectation, incurring a  $O(k)$ -approximation, or pay the large distance between the solutions, resulting in an unbounded approximation.

It is potentially possible to get around this lower bound by taking advantage of additional structure in the problem. We explore one such approach in [Section 10.4](#).

## 10.4 Competing with a Hypothesis Class of $k$ -wise Partitions Offline

In the previous section, we showed that in the offline setting, it is possible to construct an algorithm that competes with the best  $k$ -wise clustering cost for the distribution, with a factor  $O(k)$  blow-up. We also show that, without additional assumptions, this is the best approximation factor achievable for this model. In this section, we investigate ways to leverage extra information to remove the multiplicative  $O(k)$  factor. In particular, to achieve better performance than the lower bound, we must have some additional way to extract information about the location of  $S$  from  $I$ .

A learning-augmented algorithm, or indeed any algorithm to solve instances  $I$ , already encodes the mapping from instances to solutions. However, it encodes the exact mapping, and the cost of uncovering the mapping is high. It is reasonable that for many distributions of interest over  $\mathbf{I} \times \mathbf{S}$ , we can learn a *coarse* mapping between instances and solutions, that can help us search for the solution faster.

A natural way to model a coarse mapping between instances and solutions is as a  $k$ -wise partition over the instance space  $\mathbf{I}$ , i.e.  $h : \mathbf{I} \rightarrow [k]$ , where we expect that instances in a particular partition have solutions that are similar. With this additional assumption of access to a partition  $h : \mathbf{I} \rightarrow [k]$ , we can learn the best fixed prediction for each partition of  $h$ . Once we have the  $k$  predictions fixed, for a new instance  $I$ , we can evaluate  $h(I)$  and use the corresponding prediction to solve  $I$ . This avoids the factor  $k$  blow up in runtime that we had to incur in the previous section.

We note that this is somewhat different from the approach in the previous section ([Section 10.3](#)). In the previous section, we can think of the  $k$  predictions  $\mathbf{C}$  that we choose as



defining an implicit partition of the solution space, where each partition corresponds to a section in the Voronoi diagram of  $\mathbf{S}$  defined by  $\mathbf{C}$ . Because the partition is over the solution space, given a fresh instance  $I$ , it is not easy to see which partition  $I$ 's solution belongs to. Thus we must run all of the predictions, and accrue an  $\Omega(k)$  approximation factor. In this section, we are considering partitions  $h$  over *instance* space, so that given a new instance  $I$ , we can easily compute the partition that  $I$  belongs to. This also implicitly defines subsets of the solution space  $\mathbf{S}$ , where the  $i$ th (potentially overlapping) subset consists of solutions corresponding to instances in the  $i$ th partition of  $h$ . We note that these subsets may not have any particular structure.

The question remains whether it is possible to learn a good partition  $h : \mathbf{I} \rightarrow [k]$  for a distribution  $\mathcal{D}$ . We consider the setting where we must select a partition from a hypothesis class  $\mathcal{H}$  of potential efficiently-computable partitions of  $\mathbf{I}$ . We aim to minimize the expected runtime on future instance-solution pairs from  $\mathcal{D}$ . We show that when the hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{I}$  is learnable, it is indeed possible to learn an approximately optimal  $h \in \mathcal{H}$  and set of predictions, or “centers,”  $\mathbf{C} \in \mathbf{S}^k$  for the partitions of  $h$ .

We do this by constructing a loss function,  $\mathbf{C}$ -loss that can be used with an ERM oracle. In [Section 10.4.1](#), we give the main result of this section, which is showing that the  $\mathbf{C}$ -loss of a partition  $h : \mathbf{I} \rightarrow [k]$  approximates the cost of  $h$  over a distribution  $\mathcal{D}$ . In [Section 10.4.2](#) we show that if the hypothesis class  $\mathcal{H}$  of partitions is learnable, then we can learn an approximately optimal hypothesis  $h \in \mathcal{H}$ . In [Section 10.4.3](#), we conclude an algorithmic guarantee.

**Definition 10.4.1** (Clustering cost of  $k$ -wise partition). *Let  $\mathcal{D}$  be a distribution over  $\mathbf{I} \times \mathbf{S}$ , and  $h$  be a  $k$ -wise partition of  $\mathbf{S}$ . We define the cost of  $h$  over  $\mathcal{D}$  by*

$$\text{cost}(h, \mathcal{D}) = \mathbb{E}_{(I, S) \sim \mathcal{D}} \left[ \mathbf{d}(\mathbf{C}_h^{(h(I))}, S) \right],$$

where  $\mathbf{C}_h$  is the collection of the best centers for each partition of  $h$ , with respect to  $\mathcal{D}$ . That is,

$$\mathbf{C}_h^{(i)} = \operatorname{argmin}_{C \in \mathbf{S}} \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(C, S) \mid h(I) = i].$$

**Definition 10.4.2 (C-loss).** Let  $\mathcal{D}$  be a distribution over  $\mathbf{I} \times \mathbf{S}$ , and  $h$  be a  $k$ -wise partition of  $\mathbf{S}$ , and  $\mathbf{C} \in \mathbf{S}^k$  be an arbitrary set of  $k$  centers. We define the **C-loss** of  $h$  on a point  $(I, S) \in \mathbf{I} \times \mathbf{S}$  as

$$\ell_{\mathbf{C}}(h, (I, S)) = \mathbf{d}(S, \mathbf{C}^{(h(I))}).$$

We denote the expected **C-loss** of  $h$  over  $\mathcal{D}$  by

$$\ell_{\mathbf{C}}(h, \mathcal{D}) = \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}^{(h(I))})].$$

**Definition 10.4.3 (Rotation).** For  $h \in \mathcal{H}$ , where  $\mathcal{H}$  is a hypothesis class of  $k$ -wise partitions, we say that the rotation of  $h$  by some  $\varphi : [k] \rightarrow [k]$  is  $\varphi \circ h$ . (Note that  $\varphi$  does not have to be bijective.)

**Definition 10.4.4 (Rotational completion).** We say that a hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions is rotationally complete if for every  $h \in \mathcal{H}$  and  $\varphi : [k] \rightarrow [k]$ ,

$$\varphi \circ h \in \mathcal{H}.$$

For any hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions, we denote the rotational completion of  $\mathcal{H}$  by  $\operatorname{rc}(\mathcal{H})$ , where we define

$$\operatorname{rc}(\mathcal{H}) = \{\varphi \circ h \mid h \in \mathcal{H}, \varphi : [k] \rightarrow [k]\}.$$

### 10.4.1 Approximation

The main technical component of this section is designing the  $\mathbf{C}$ -loss (Definition 10.4.2), that approximates the clustering cost of a  $k$ -wise partition, while also being decomposable and therefore usable with an ERM oracle.

**Lemma 10.4.5** ( $\ell_{\mathbf{C}}(h, \mathcal{D})$  approximates clustering cost). *Consider a distribution  $\mathcal{D}$  over pairs  $(I, S) \in (\mathbf{I}, \mathbf{S})$ , an arbitrary set of  $k$  centers  $\mathbf{C} = (\mathbf{C}^{(1)}, \dots, \mathbf{C}^{(k)}) \in \mathbf{S}^k$ , and a hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{S}$ . For every  $h \in \mathcal{H}$ , there exists a rotation  $\varphi : [k] \rightarrow [k]$ , such that*

$$\ell_{\mathbf{C}}(\varphi \circ h, \mathcal{D}) \leq O(1) \cdot (\text{cost}(h, \mathcal{D}) + \text{cost}(\mathbf{C}, \mathcal{D})).$$

*Proof.* First, we consider the cost associated with a particular partition  $i$  of  $h$ . We show how to choose a center  $j$  from  $\mathbf{C}$  to assign this partition. Let  $\mathbf{C}_h^{(i)}$  be the best possible center in  $\mathbf{S}$  for partition  $i$  of  $h$  for  $\mathcal{D}$ . That is,

$$\mathbf{C}_h^{(i)} = \operatorname{argmin}_{C \in \mathbf{S}} \mathbb{E}_{(I, S) \sim \mathcal{D}} [\mathbf{d}(S, C) \mid h(I) = i].$$

Let  $\mathbf{C}(S)$  be the best center from  $\mathbf{C}$  for a solution  $S \in \mathbf{S}$ . That is,

$$\mathbf{C}(S) = \operatorname{argmin}_{\mathbf{C}^{(j)}} \mathbf{d}(S, \mathbf{C}^{(j)}).$$

Now, we can associate  $\mathbf{C}_h^{(i)}$  with its closest center in  $\mathbf{C}$ . Formally, let

$$\varphi(i) = \operatorname{argmin}_{j \in [k]} \mathbf{d}(\mathbf{C}^{(j)}, \mathbf{C}_h^{(i)}).$$

Then we have

$$\begin{aligned}
& \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}^{(\varphi(i))}) \mid h(I) = i] \\
& \leq \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) \mid h(I) = i] + \mathbf{d}(\mathbf{C}^{(\varphi(i))}, \mathbf{C}_h^{(i)}) \quad \text{triangle inequality} \\
& \leq \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) \mid h(I) = i] + \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(\mathbf{C}(S), \mathbf{C}_h^{(i)}) \mid h(I) = i] \\
& \leq \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) \mid h(I) = i] + \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) + \mathbf{d}(S, \mathbf{C}(S)) \mid h(I) = i] \quad \text{triangle inequality} \\
& = 2 \cdot \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) \mid h(I) = i] + \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}(S)) \mid h(I) = i]
\end{aligned}$$

Taking the appropriate combination over the partitions  $i$ , we get

$$\begin{aligned}
& \ell_{\mathbf{C}}(\varphi \circ h, \mathcal{D}) \\
& = \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}^{(\varphi(h(I)))})] \\
& = \sum_{i \in [k]} \mathbb{P}_{(I,S) \sim \mathcal{D}} [h(I) = i] \cdot \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}^{(\varphi(i))}) \mid h(I) = i] \\
& \leq \sum_{i \in [k]} \mathbb{P}_{(I,S) \sim \mathcal{D}} [h(I) = i] \cdot \left( 2 \cdot \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(i)}) \mid h(I) = i] + \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}(S)) \mid h(I) = i] \right) \\
& = 2 \cdot \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}_h^{(h(I))})] + \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, \mathbf{C}(S))] \\
& = 2 \cdot \text{cost}(h, \mathcal{D}) + \text{cost}(\mathbf{C}, \mathcal{D}).
\end{aligned}$$

□

### 10.4.2 Learning the Hypothesis Class

In this section, we show a series of lemmas that imply that learnability of the hypothesis class  $\mathcal{H}$  is enough to imply that the minimum loss  $h \in \text{rc}(\mathcal{H})$  is learnable. The proofs of these lemmas are largely straightforward sample complexity arguments, which we include

for completeness.

To characterize the learnability of hypothesis classes, we find it simplest to go through  $\Psi_B$ -dimension, as defined by Ben-David, Cesa-Bianchi, and Long [1992].

**Lemma 10.4.6** ( $\Psi_B$ -dimension of  $\text{rc}(\mathcal{H})$ ). *For a hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions over  $\mathbf{I}$ , the  $\Psi_B$ -dimension of  $\text{rc}(\mathcal{H})$  can only be an order  $k$  factor (up to logarithmic factors) larger than the  $\Psi_B$ -dimension of  $\mathcal{H}$ . That is,*

$$\Psi_B(\text{rc}(\mathcal{H})) \in O\left(k\Psi_B(\mathcal{H})\log(k\Psi_B(\mathcal{H}))\right).$$

*Proof.* Suppose that  $\text{rc}(\mathcal{H})$  can  $\Psi_B$  shatter  $n$  points  $x_1, \dots, x_n \in \mathbf{I}$ . This means that there exist functions  $\psi_1, \dots, \psi_n : [k] \rightarrow \{0, 1\}$  such that for every labeling  $y \in \{0, 1\}^n$ , there exists an  $h' \in \text{rc}(\mathcal{H})$  such that

$$\psi_i(h'(x_i)) = y_i, \quad \forall i.$$

Let  $S$  be a set containing one  $h'$  achieving each labeling  $y$ . Thus,  $|S| = 2^n$ . Associate each  $h' \in S$  with a choice of  $\varphi$  and  $h \in \mathcal{H}$  such that  $h' = \varphi \circ h$ .

There are at most  $k^k$  possible values of  $\varphi$ . Thus, there must be some  $\varphi^* : [k] \rightarrow [k]$  such that at least  $2^n/k^k$  elements of  $S$  are associated with  $\varphi^*$ . Let  $S'$  be the subset of  $S$  containing  $h'$  that are associated with  $\varphi^*$ . We have that each  $h' \in S'$  maps to a distinct value of

$$(\psi_1(h'(x_1)), \dots, \psi_n(h'(x_n))) = ((\psi_1 \circ \varphi^*)(h(x_1)), \dots, (\psi_n \circ \varphi^*)(h(x_n))).$$

Let  $\mathcal{H}'$  be the set of  $h$  that are associated with  $h' \in S'$ . The above tells us that for functions  $(\psi_1 \circ \varphi^*), \dots, (\psi_n \circ \varphi^*)$ , the hypothesis classes in  $\mathcal{H}'$  span at least  $2^n/k^k = n^{\frac{n-k \log k}{\log n}}$  labelings

of  $x_1, \dots, x_n$ .

By the Sauer-Shelah lemma, this means that there must be a subset  $X' \subseteq (x_1, \dots, x_n)$  of size  $\Omega((n - k \log k)/\log n)$  that is  $\Psi_B$ -shattered by  $S'$  with respect to the functions  $(\psi_1 \circ \varphi), \dots, (\psi_n \circ \varphi)$ .

Let  $q$  be the  $\Psi_B$ -dimension of  $\mathcal{H}$ . Since  $q \in \Omega(\Omega((n - k \log k)/\log n))$ , we have that  $n \in O(kq \log kq)$ . Since this applies when  $n$  is  $\Psi_B$ -dimension of  $\text{rc}(\mathcal{H})$ , we have that

$$\Psi_B(\text{rc}(\mathcal{H})) \in O\left(k\Psi_B(\mathcal{H}) \log(k\Psi_B(\mathcal{H}))\right).$$

□

A simple reduction can construct an ERM oracle for  $\text{rc}(\mathcal{H})$  using calls to an ERM oracle for  $\mathcal{H}$ , where the number of calls depends only on  $k$ . However, we note that it may be possible to get a much more efficient ERM oracle, for example when  $\mathcal{H}$  is rotationally complete to begin with.

**Lemma 10.4.7** (Converting ERM for  $\mathcal{H}$  to ERM for  $\text{rc}(\mathcal{H})$ ). *For a hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{I}$  and an empirical distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ , given access to an oracle  $\mathcal{O}$  that can answer queries of the form*

$$\underset{h \in \mathcal{H}}{\text{argmin}} \ell_{\mathbf{C}}(h, \mathcal{D})$$

*for any set of centers  $\mathbf{C} \in \mathbf{S}^k$ , we can construct an oracle  $\mathcal{O}_{\text{rc}}$  that can answer queries of the form*

$$\underset{h' \in \text{rc}(\mathcal{H})}{\text{argmin}} \ell_{\mathbf{C}}(h', \mathcal{D})$$

*for any set of centers  $\mathbf{C} \in \mathbf{S}^k$  using  $k^k$  calls to  $\mathcal{O}$ .*

*Proof.* There are exactly  $k^k$  possible choices for  $\varphi : [k] \rightarrow [k]$ . Define

$$\varphi(\mathbf{C}) = (\mathbf{C}^{(\varphi(1))}, \dots, \mathbf{C}^{(\varphi(k))}).$$

Then we have that

$$\ell_{\mathbf{C}}(\varphi \circ h, \mathcal{D}) = \ell_{\varphi(\mathbf{C})}(h, \mathcal{D}).$$

Therefore,

$$\begin{aligned} \operatorname{argmin}_{h' \in \operatorname{rc}(\mathcal{H})} \ell_{\mathbf{C}}(h', \mathcal{D}) &= \min_{\varphi: [k] \rightarrow [k]} \ell_{\mathbf{C}} \left( \varphi \circ \left( \operatorname{argmin}_{h \in \mathcal{H}} \ell_{\mathbf{C}}(\varphi \circ h, \mathcal{D}) \right), \mathcal{D} \right) \\ &= \min_{\varphi: [k] \rightarrow [k]} \ell_{\mathbf{C}} \left( \varphi \circ \left( \operatorname{argmin}_{h \in \mathcal{H}} \ell_{\varphi(\mathbf{C})}(h, \mathcal{D}) \right), \mathcal{D} \right). \end{aligned}$$

Thus, we can evaluate this by enumerating over all choices of  $\varphi$ , and making one call to  $\mathcal{O}$  for each  $\varphi$ . Then, we can evaluate the empirical loss of  $\varphi$  with the minimizing  $h \in \mathcal{H}$ , and choose the  $\varphi$  that achieves the lowest loss.  $\square$

**Lemma 10.4.8** (Pseudo-dimension of loss functions). *For a fixed  $\mathbf{C}$ , and hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions over  $\mathbf{I}$ , the pseudo-dimension of the set of loss functions  $\ell_{\mathbf{C}}(h, \cdot)$  for  $h \in \mathcal{H}$  is bounded by the  $\Psi_B$ -dimension of  $\mathcal{H}$ .*

*Proof.* Let  $d$  be the pseudo-dimension of the set of  $\ell_{\mathbf{C}}(h, \cdot)$  for  $h \in \mathcal{H}$ . This means that there exist points  $x_1, \dots, x_d \in \mathbf{I}$  and thresholds  $t_1, \dots, t_d \in \mathbb{R}$  such that for every labeling  $y \in \{0, 1\}^n$ , there exists an  $h \in \mathcal{H}$  such that for all  $i \in [d]$

$$\begin{cases} \ell_{\mathbf{C}}(h, x_i) \leq t_i & \text{if } y_i = 0 \\ \ell_{\mathbf{C}}(h, x_i) > t_i & \text{if } y_i = 1 \end{cases}.$$

For each  $i$ , we construct a function  $\psi_i : [k] \rightarrow \{0, 1\}$  via

$$\psi_i(z) = \begin{cases} 0 & \text{if } \mathbf{d}(\mathbf{C}^{(i)}, x_i) \leq t_i \\ 1 & \text{if } \mathbf{d}(\mathbf{C}^{(i)}, x_i) > t_i \end{cases}.$$

This ensures that for a hypothesis  $h \in \mathcal{H}$ ,  $\psi_i(h(x_i)) = 0$  if and only if  $\ell_{\mathbf{C}}(h, x_i) \leq t_i$ . Thus,  $\psi_1, \dots, \psi_d$  witness the  $\Psi_B$ -shattering of  $x_1, \dots, x_d$ , and the pseudo-dimension of the set of  $\ell_{\mathbf{C}}(h, \cdot)$  for  $h \in \mathcal{H}$  is at most the  $\Psi_B$ -dimension of  $\mathcal{H}$ .  $\square$

**Lemma 10.4.9** (Learning guarantee). *Given an arbitrary set of centers  $\mathbf{C}$  over a bounded space  $\mathbf{S}$ , a learnable (finite  $\Psi_B$ -dimension) rotationally complete hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{I}$ , an ERM oracle  $\mathcal{O}$  that can compute  $\arg\min_{h \in \mathcal{H}} \ell_{\mathbf{C}}(h, \mathcal{D}')$  for empirical distributions  $\mathcal{D}'$ , and sample access to a distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ , we can learn an  $h \in \mathcal{H}$  and a set of centers  $\mathbf{C}_h \in \mathbf{S}^k$  such that*

$$\mathbb{E}_{(I, S) \sim \mathcal{D}} [\mathbf{d}(\mathbf{C}_h^{h(I)}, S)] \leq O(1) \left( 1 + \text{cost}(\mathbf{C}, \mathcal{D}) + \min_{h' \in \mathcal{H}} \text{cost}(h', \mathcal{D}) \right).$$

*In particular, if  $d$  is the  $\Psi_B$ -dimension of  $\mathcal{H}$ , and  $\mathbf{d}_{\max}$  is the largest distance in  $\mathbf{S}$ , with probability  $1 - \delta$  we can learn the above in*

$$O(\mathbf{d}_{\max}^2 (d + \ln \frac{1}{\delta}))$$

*samples, where  $\mathbf{d}_{\max}$  is the largest distance in  $\mathbf{S}$ .*

*Proof.* Since  $\mathcal{H}$  has  $\Psi_B$  dimension  $d$ , by [Lemma 10.4.8](#) the set of loss functions  $\mathcal{L} = \{\ell_{\mathbf{C}}(h, \cdot) \mid h \in \mathcal{H}\}$  has pseudo-dimension at most  $d$ .

By uniform convergence bounds for  $\Psi_B$ -dimension [[Ben-David, Cesa-Bianchi, and Long](#),



1992], we have that for any  $\delta \in (0, 1)$ , any distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ , since all of the loss functions in  $\mathcal{L}$  have value bounded in  $[0, D]$ ,  $m = O(\mathbf{d}_{\max}^2(d + \ln \frac{1}{\delta}))$  samples are sufficient to ensure that with probability  $1 - \frac{\delta}{4}$  over the draw of  $\{(I, S)_1, \dots, (I, S)_m\} \sim \mathcal{D}^m$ , for all  $h \in \mathcal{H}$ , the difference between the average empirical loss over the samples and the expected loss over  $\mathcal{D}$  is at most a constant, i.e.:

$$\left| \frac{1}{m} \sum_{j=1}^m \ell_{\mathbf{C}}(h, (I, S)_j) - \ell_{\mathbf{C}}(h, \mathcal{D}) \right| \leq 1. \quad (10.1)$$

Let  $\mathcal{D}_m$  be the empirical distribution of  $m$  samples drawn i.i.d. from  $\mathcal{D}$ . The ERM oracle  $\mathcal{O}$  then gives us  $\hat{h}$  such that

$$\hat{h} = \operatorname{argmin}_{h \in \mathcal{H}} \ell_{\mathbf{C}}(h, \mathcal{D}_m).$$

Lemma 10.4.5 implies that, for a rotationally complete  $\mathcal{H}$ , and

$$h^* = \operatorname{argmin}_{h \in \mathcal{H}} \operatorname{cost}(h, \mathcal{D}),$$

the ERM minimizer  $\hat{h}$  achieves

$$\ell_{\mathbf{C}}(\hat{h}, \mathcal{D}_m) \leq O(1) \cdot (\operatorname{cost}(h^*, \mathcal{D}_m) + \operatorname{cost}(\mathbf{C}, \mathcal{D}_m)),$$

Combining this with Equation (10.1), we get that

$$\ell_{\mathbf{C}}(\hat{h}, \mathcal{D}) \leq O(1) \cdot (1 + \operatorname{cost}(h^*, \mathcal{D}_m) + \operatorname{cost}(\mathbf{C}, \mathcal{D}_m)).$$

We bound the deviation of  $\operatorname{cost}(h^*, \mathcal{D}_m)$  and  $\operatorname{cost}(\mathbf{C}, \mathcal{D}_m)$  from  $\operatorname{cost}(h^*, \mathcal{D})$  and  $\operatorname{cost}(\mathbf{C}, \mathcal{D})$ .

Specifically, let  $\mathbf{C}_{h^*}$  be the optimal set of centers for each partition of  $h^*$  with respect to  $\mathcal{D}$ .

We can write  $\text{cost}(h^*, \mathcal{D}_m) \leq \frac{1}{m} \sum_{i=1}^m X_i$ , where  $X_i = \mathbf{d}\left(C_{h^*}^{(h^*(I_i))}, S_i\right)$ , where  $(I_i, S_i)$  are the independent samples drawn from  $\mathcal{D}$ . Let  $X = \sum_{i=1}^m X_i$ . By a Chernoff-Hoeffding bound, we have that

$$\begin{aligned} \mathbb{P}[X \geq \mathbb{E}[X] + m] &\leq \exp\left(-\frac{(\mathbb{E}[X] + m)^2}{\mathbb{E}[X] + m}\right) \\ &\leq \exp\left(-\frac{1}{2}(\mathbb{E}[X] + m)\right). \end{aligned}$$

Thus, we have that as long as  $m \geq O(\log(\frac{1}{\delta}))$ , we have

$$\text{cost}(h^*, \mathcal{D}_m) \leq 2 \cdot \text{cost}(h^*, \mathcal{D}) + 1,$$

with probability  $\geq 1 - \frac{\delta}{4}$ .

Similarly, let  $Y = Y_1 + \dots + Y_m$ , where  $Y_i = \mathbf{d}(\mathbf{C}(S_i), S_i)$  where the  $S_i$  are the  $m$  independent samples we drew from  $\mathcal{D}$ . Then  $\text{cost}(\mathbf{C}, \mathcal{D}_m) = \frac{1}{m} \sum_{i=1}^m Y_i$ . By a Chernoff-Hoeffding bound, we have

$$\mathbb{P}[Y \geq \mathbb{E}[Y] + m] \leq \exp\left(-\frac{1}{2}(\mathbb{E}[Y] + m)\right).$$

Thus, as long as  $m \geq O(\log(\frac{1}{\delta}))$ , we have

$$\text{cost}(\mathbf{C}, \mathcal{D}_m) \leq \text{cost}(\mathbf{C}, \mathcal{D}) + 1,$$

with probability  $\geq 1 - \frac{\delta}{4}$ .

Taking a union bound over these events, we have that with  $m = O(\mathbf{d}_{\max}^2(d + \ln \frac{1}{\delta}))$  samples,

$$\mathbb{E}_{(I,S) \sim \mathcal{D}} \left[ \mathbf{d}(\mathbf{C}^{\hat{h}(I)}, S) \right] \leq O(1) \left( 1 + \text{cost}(\mathbf{C}, \mathcal{D}) + \min_{h' \in \mathcal{H}} \text{cost}(h', \mathcal{D}) \right),$$

with probability  $\geq 1 - \delta$ . □

### 10.4.3 Algorithmic Guarantee

Finally, we can use the above procedure to design an algorithm that can take advantage of course information in the form of partitions of  $\mathbf{S}$ , to achieve a potentially improved runtime guarantee for future instances.

**Theorem 10.4.10** (Competing with a hypothesis class of  $k$ -wise partitions offline). *Given an algorithm-with-predictions  $\mathcal{A}$ , a learnable (finite  $\Psi_B$ -dimension) rotationally complete hypothesis class  $\mathcal{H}$  of  $k$ -wise partitions of  $\mathbf{I}$ , an ERM oracle  $\mathcal{O}$  that can compute  $\operatorname{argmin}_{h \in \mathcal{H}} \ell_{\mathbf{C}}(h, \mathcal{D}')$  for empirical distributions  $\mathcal{D}'$ , and  $O(\mathbf{d}_{\max}^2(d + \ln \frac{1}{\delta}))$  samples from distribution  $\mathcal{D}$  over  $\mathbf{I} \times \mathbf{S}$ , it is possible to learn an  $h : \mathbf{I} \rightarrow [k]$  and set of centers  $\mathbf{C} = (C^{(1)}, \dots, C^{(k)})$  that are an  $O(1)$  approximation to*

$$\min_{h \in \mathcal{H}} \min_{\mathbf{C} \in \mathbf{S}^k} \mathbb{E}_{(I, S) \sim \mathcal{D}} [\mathbf{d}(S, C^{(h(I))})].$$

Furthermore, given an  $h : \mathbf{I} \rightarrow [k]$  and  $\mathbf{C} = (C^{(1)}, \dots, C^{(k)})$ , we can design a procedure such that the expected time to solve a new instance is

$$\mathbb{E}_{(I, S) \sim \mathcal{D}} [\text{runtime}(h(I)) + \mathbf{d}(S, C^{(h(I))})].$$

*Proof.* By [Lemma 10.3.4](#) we know that we can use samples from  $\mathcal{D}$  to find a set of centers  $\hat{\mathbf{C}}$  that are an  $O(1)$ -approximation to

$$\operatorname{argmin}_{\mathbf{C}} \text{cost}(\mathbf{C}, \mathcal{D}).$$

By [Lemma 10.4.9](#) we have that, using this  $\widehat{\mathbf{C}}$ , and additional samples from  $\mathcal{D}$ , we can find a hypothesis  $\widehat{h} \in \mathcal{H}$  such that

$$\begin{aligned}
\mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(\widehat{\mathbf{C}}^{\widehat{h}(I)}, S)] &\leq O(1) \left( 1 + \text{cost}(\widehat{\mathbf{C}}, \mathcal{D}) + \min_{h' \in \mathcal{H}} \text{cost}(h', \mathcal{D}) \right) \\
&\leq O(1) \left( 1 + \min_{\mathbf{C} \in \mathbf{S}^k} \text{cost}(\mathbf{C}, \mathcal{D}) + \min_{h' \in \mathcal{H}} \text{cost}(h', \mathcal{D}) \right) \\
&\leq O(1) \cdot \min_{h' \in \mathcal{H}} \text{cost}(h', \mathcal{D}) \\
&\leq O(1) \cdot \min_{h \in \mathcal{H}} \min_{\mathbf{C} \in \mathbf{S}^k} \mathbb{E}_{(I,S) \sim \mathcal{D}} [\mathbf{d}(S, C^{(h(I))})] ,
\end{aligned} \tag{10.2}$$

where the [Equation \(10.2\)](#) follows because the cost of the best hypothesis is lower bounded by the cost of the best clustering, and because we assume all distances in  $\mathbf{S}$  are lower bounded by 1 (see preliminaries, [Section 10.2](#)).

Finally, once we have a hypothesis  $h \in \mathcal{H}$  and the centers  $\mathbf{C}_h$ , on subsequent instances  $I$  drawn from  $\mathcal{D}$ , we can compute  $\mathcal{A}(I, \mathbf{C}_h^{(h(I))})$ , in expected time

$$\mathbb{E}_{(I,S) \sim \mathcal{D}} [\text{runtime}(h(I)) + \mathbf{d}(S, C^{(h(I))})] .$$

□

## 10.5 Competing with Trajectories Online

In this section, we consider the problem of solving a series of instance that arrive online. Consider an instance  $I \in \mathbf{I}$  with (unknown) true solution  $S \in \mathbf{S}$ . As before, we assume that the solution space  $\mathbf{S}$  is equipped with a metric distance  $\mathbf{d}$ , which we can calculate efficiently (see [Assumption 10.2.2](#)). We can think of an algorithm with predictions as searching the solution space  $\mathbf{S}$  by growing a ball around a prediction  $P \in \mathbf{S}$  of our choice. The cost of

the algorithm is the radius of the ball necessary to find  $S$ . Furthermore, we can search from multiple points, and the total cost will be the sum of the radii searched from each of the points.

**Definition 10.5.1** (Online Ball Search). *In the online setting, we consider instances arriving over  $T$  days. On each day  $t \in [T]$  an instance  $I_t \in \mathbf{I}$  arrives, with (unknown) solution  $S_t \in \mathbf{S}$ . We are given access to an algorithm with predictions  $\mathcal{A}$  such that for any prediction  $P \in \mathbf{S}$ , the runtime of  $\mathcal{A}(I_t, P)$  is bounded by  $\mathbf{d}(S_t, P)$ . On each day  $t \in [T]$ , the algorithm must output  $S_t$ .*

In the online setting, we want to minimize the total work that the algorithm does over a sequence of  $T$  instances. We make *no distributional assumptions* about the instance-solution pairs that arrive. Instead, we approach the problem through the lens of *competitive analysis*, and provide algorithms that compete with the best strategies in hindsight.

In [Section 10.5.1](#), we define the offline strategies that we are competing against. This requires careful consideration of various modeling constraints. In [Section 10.5.2](#), we give an algorithm that is  $O(k^2)$ -competitive against these baselines in the total radius searched, via a reduction to  $k$ -server. We note that this does not immediately solve our problem of interest, as we wish to bound the *total work* done by our algorithm, which is only lower bounded by the total radius searched. In [Section 10.5.3](#), we give an algorithm that is  $O(k^4 \ln^2 k)$ -competitive in the total radius searched, and which has total runtime bounded by an  $O(1)$  factor times the total radius searched. To approach this, we use significantly different techniques than in the reduction to  $k$ -server.

### 10.5.1 Offline Baselines

We will analyze algorithms for this problem under the paradigm of competitive analysis, where we compare the performance of our algorithm to the performance of the best offline strategy in hindsight. To do this, we must define the set of offline strategies that we are competing with.

In our problem setting, on each day, the algorithm is allowed to search outward from any set of points, and pays for the total radius that is searched before that day's point is found. A natural idea is to allow the offline strategies to have the same form. However, the offline strategy has full knowledge of all of the requests that arrive in the sequence. So, on each day, it could search exactly from that day's request, achieving zero cost. Thus, this set of offline strategies is not particularly meaningful.

This motivates us to consider a restricted set of offline strategies, that captures some structure in the input that we could hope to take advantage of. This is reasonable, as we must leverage some structure to get a fast algorithm for sequences of inputs. Otherwise, if we had a fast algorithm for arbitrary sequences of inputs, this would imply a faster algorithm for one arbitrary input.

Previous work imposes structure by providing a guarantee that competes with the best fixed prediction in hindsight [Khodak, Balcan, Talwalkar, and Vassilvitskii, 2022]. This can be interpreted as imposing the structure that the data is well-clustered, and therefore the best fixed point performs well for the distribution of inputs. This objective function is analogous to a 1-medians objective, as our cost is the sum of the distances from the requests to the center (best fixed prediction in hindsight).

The first extension is to allow the offline strategy to move over time. We must be careful

about how we do this, as allowing the offline point to move arbitrarily on each day results in the zero cost issue that we discussed earlier. Thus, we allow the point to move, but we charge the offline strategy for the total distance that the point moves. That is, the cost to the offline strategy on a day  $t$  is the sum of the *movement cost*, the distance that the strategy moved its prediction on day  $t$ , and the *hit cost*, the distance from the request to the prediction after the prediction moves. We call this strategy a *trajectory*.

**Definition 10.5.2** (Trajectory). *A trajectory, with respect to a sequence of instances  $\{I_t : t \in [T]\}$ , is a sequence of predictions  $P_t \in \mathbf{S}$ , for  $t \in [T]$ . The cost of a trajectory is the sum of the prediction costs for each day's prediction ("hit cost"), plus the total distance that the trajectory moves ("movement cost"),*

$$\text{cost}(\{P_t : t \in [T]\}) = \left[ \sum_{i=1}^T \mathbf{d}(P_t, S_t) \right] + \left[ \sum_{i=1}^T \mathbf{d}(P_{t-1}, P_t) \right],$$

where  $S_t$  is the solution associated with instance  $I_t$ , and we define the initial  $P_0 = \mathbf{0}$ , is a fixed arbitrary starting prediction.

The second extension is that we compete against offline strategies that consist of  $k$  points. That is, on each day, the hit cost of the offline strategy is the distance from the request to the nearest of the  $k$  points. This setting is analogous to a  $k$ -medians objective, as the cost of each request is the distance from the request to the closest of  $k$  centers. We note that the extension from one center to  $k$  centers makes the offline problem of choosing the best strategy go from being convex, to being non-convex.

**Definition 10.5.3** (Multiple trajectories). *A collection of  $k$  trajectories, with respect to a sequence of instances  $\{I_t : t \in [T]\}$ , is a sequence of predictions  $\{P_t \in \mathbf{S} : t \in [T]\}$ , and associates each day  $t$  with one of the trajectories  $\in [k]$ . We use  $\mathcal{T}^{(i)} \subseteq [T]$  to refer the days that are associated with trajectory  $i$ , and we denote the collection of trajectories by*

$$(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}).$$

For a day  $t \in [T]$ , we use  $\text{prev}^{(i)}(t)$  to refer to the closest day previous to  $t$  that belongs to trajectory  $i$ , and  $\text{next}^{(i)}(t)$  to refer to the closest day subsequent to  $t$  that belongs to trajectory  $i$ . If  $t$  is the first day in its trajectory, then we define  $\text{prev}^{(i)}(t) = P_0 = \mathbf{0}$ .

The cost of the collection of trajectories is the sum of the costs of the  $k$  trajectories. The cost of trajectory  $i$  is the single trajectory cost of  $\{P_t : t \in \mathcal{T}^{(i)}\}$  with respect to  $\{I_t : t \in \mathcal{T}^{(i)}\}$ . That is,

$$\text{cost}(\{P_t : t \in \mathcal{T}^{(i)}\}) = \left[ \sum_{t \in \mathcal{T}^{(i)}} \mathbf{d}(P_t, S_t) \right] + \left[ \sum_{t \in \mathcal{T}^{(i)}} \mathbf{d}(P_{\text{prev}^{(i)}(t)}, P_t) \right],$$

$$\text{cost}(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}) = \sum_{i=1}^k \text{cost}(\{P_t : t \in \mathcal{T}^{(i)}\}).$$

We note that while the offline strategy is limited to maintaining a palette of  $k$  predictions, and must pay to move the predictions, we do not make this requirement of the online algorithm.

Another way to interpret a  $k$ -trajectory baseline for a sequence of requests, is that it is essentially the offline  $k$ -server cost for those requests. In the  $k$ -server problem, the algorithm must maintain a set of  $k$ -servers in a metric space  $\mathbf{S}$ . On each day, a request arrives at some point  $S \in \mathbf{S}$ , and the algorithm must move one of its servers to  $S$ . The cost to the algorithm is the total distance it moves its servers. The offline baseline is the best way to move the servers, when the sequence of requests is known ahead of time.

**Lemma 10.5.4** (Multiple trajectories are approximately  $k$ -server baselines).

Let  $\mathcal{S} = (S_1, \dots, S_T)$  be a sequence of  $T$  requests (solutions).

Let  $\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}$  be the offline optimal collection of  $k$  trajectories for  $\mathcal{S}$ . Let  $\text{serveropt}_k(\mathcal{S})$  be the cost of the offline optimal  $k$ -server solution serving  $\mathcal{S}$ . We have



that

$$\begin{aligned}
& \text{cost}(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}) \\
& \leq \text{serveropt}_k(\mathcal{S}) \\
& \leq 2 \cdot \text{cost}(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}).
\end{aligned}$$

*Proof.* The first inequality follows from observing that a  $k$ -server solution that serves the requests  $\mathcal{S}$  is a valid collection of  $k$  trajectories, that has movement cost equal to the cost of the  $k$ -server solution, and zero hit cost.

For the second inequality, we convert the collection of  $k$  trajectories into an offline  $k$ -server solution as follows. We assign each of trajectory one server, that services the requests that correspond to that trajectory. We can bound the cost to server  $i$  by the cost to trajectory  $i$ . We can bound the cost of this  $k$ -server solution by

$$\begin{aligned}
\sum_{t \in \mathcal{T}^{(i)}} \mathbf{d}(S_{\text{prev}^{(i)}(t)}, S_t) & \leq \sum_{t \in \mathcal{T}^{(i)}} [\mathbf{d}(S_{\text{prev}^{(i)}(t)}, P_{\text{prev}^{(i)}(t)}) + \mathbf{d}(P_{\text{prev}^{(i)}(t)}, P_t) + \mathbf{d}(P_t, S_t)] \\
& \leq 2 \left[ \sum_{t \in \mathcal{T}^{(i)}} \mathbf{d}(S_t, P_t) \right] + \left[ \sum_{t \in \mathcal{T}^{(i)}} \mathbf{d}(P_{\text{prev}^{(i)}(t)}, P_t) \right] \\
& \leq 2 \cdot \text{cost}(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\}).
\end{aligned}$$

□

This observation already gives the following algorithm that is competitive against one trajectory.

**Corollary 10.5.5** (“Predict yesterday’s solution” competes with any single trajectory). *The “predict yesterday’s solution” strategy (Algorithm 10.2) is  $O(1)$ -competitive with the best*

---

**Algorithm 10.2** “Predict Yesterday’s Solution”

---

```

1:  $P = \mathbf{0}$ 
2: for day  $t$ , instance  $I_t$  arrives do
3:    $S_t = \mathcal{A}(I_t, P)$ 
4:    $P = S_t$ 
5:   Output solution  $S_t$ 
6: end for

```

---

single trajectory in hindsight. Furthermore, the total runtime of [Algorithm 10.2](#) is bounded by  $O(1)$  times the total radius searched.

*Proof.* [Lemma 10.5.4](#) tells us that the best offline trajectory has cost within a factor 2 of the optimal offline 1-server strategy. For any sequence of requests, the optimal 1-server strategy is simply the one that moves the single server to each request on each day, and pays cost

$$\sum_{t=1}^T \mathbf{d}(S_{t-1}, S_t).$$

This is also the cost of [Algorithm 10.2](#). Thus, [Algorithm 10.2](#) is 2-competitive against any fixed strategy.

Furthermore, the runtime of [Algorithm 10.2](#) is dominated by the time to run the algorithms-with-predictions subroutine, so the total runtime of [Algorithm 10.2](#) is bounded by  $O(1)$  times the total radius searched.  $\square$

*Remark.* The guarantee given by this strategy is already stronger than what is shown in previous work. [Khodak, Balcan, Talwalkar, and Vassilvitskii \[2022\]](#) give a strategy that competes with the best *fixed* prediction in hindsight. This strategy allows us to compete with trajectories that are adaptive, in that they move over time.

### 10.5.2 Reduction to $k$ -server

We consider the setting of competing against a collection of  $k$  trajectories. In this setting, we can think of the offline optimum as maintaining  $k$  prediction points. On each day, the cost is the distance from today's solution to the minimum of these points. Like the single trajectory setting, these points can move over time, and the offline cost is charged for the total distance that the  $k$  points move, capturing an adaptive strategy.

In this section, we show how get a competitive algorithm for the online ball search problem using a competitive algorithm for the  $k$ -server problem, with an  $k$  factor blow-up in the competitive ratio. We then discuss why this does not immediately imply a fast algorithm for solving sequences of instances.

In the  $k$ -server problem, an algorithm must maintain the positions of  $k$  servers in a metric space  $\mathbf{S}$ . On each day  $t \in [T]$ , a request  $r_t \in \mathbf{S}$  is made, and the algorithm must move at least one server to  $r_t$ . The cost to the algorithm is the total distance that all of the servers move. The competitive ratio of the algorithm is given with respect to the best offline solution, i.e. the best way to service the requests in order with  $k$  servers in hindsight. For an introduction to the  $k$ -server problem, the reader is referred to the survey of [Koutsoupias \[2009\]](#), with the note that there has been a good deal of work on this problem since the survey was published.

**Theorem 10.5.6** (Online ball search to  $k$ -server reduction). *Given an  $\alpha$ -competitive algorithm for the  $k$ -server problem on metric space  $\mathbf{S}$ , we can construct a  $O(k\alpha)$ -competitive algorithm for the online ball search problem on  $\mathbf{S}$ .*

*Proof.* We maintain  $k$  predictions  $P_1, \dots, P_k \in \mathbf{S}$ , which can also be thought of as the position of  $k$  servers.

On each day  $t \in [T]$ , instance  $I_t$  arrives. Our algorithm runs  $\mathcal{A}(I_t, P_i)$  for all  $i \in [k]$  in

parallel<sup>4</sup> at equal rates, until one of them terminates and finds  $S_t$ . Then, we take  $S_t$  to be the request for the  $k$ -server problem for day  $t$ , and run our competitive  $k$ -server algorithm to update the positions of the predictions/servers.

The cost to the online ball search problem is  $k$  times the distance from the closest prediction (at the beginning of day  $t$ ) to  $S_t$ . Since the  $k$ -server algorithm must service  $S_t$ , the cost to the  $k$ -server algorithm on day  $t$  is at least the distance from the closest server/prediction (at the beginning of day  $t$ ) to  $S_t$ . Thus the cost to the online ball search algorithm is bounded by at most  $k$  times the cost to the  $k$ -server algorithm.

Finally, the  $k$ -server algorithm is  $\alpha$ -competitive against the best offline  $k$ -server solution that visits all of the requests  $S_t$ . By [Lemma 10.5.4](#), we know that this has cost at most  $O(1)$  times more than the  $k$  best trajectories for the sequence of  $S_t$ s. Thus, this algorithm for online ball search is  $O(k\alpha)$ -competitive against  $k$  trajectories in the number of steps it takes of the subroutine  $\mathcal{A}$ . □

**Corollary 10.5.7** ( $O(k^2)$ -competitive algorithm for online ball search). *There exists an  $O(k^2)$ -competitive algorithm for the online ball search problem.*

This follows from the fact that the work function algorithm is  $O(k)$ -competitive for the  $k$ -server problem [[Koutsoupias and Papadimitriou, 1995](#)].

The strategy of reducing online ball search to the  $k$ -server problem is both promising and presents some challenges. On the one hand, the  $k$ -server problem is a natural and very well-studied problem in online algorithms. It is promising that improvements to algorithms for the  $k$ -server problem can naturally translate to the online ball search problem.

---

<sup>4</sup>Here, by “in parallel” we refer to interleaving the steps of the various threads, resulting in a sequential algorithm.

On the other hand, the online ball search problem appears to have some advantages that are lost in the reduction to  $k$ -server. For example, an online ball search algorithm is not limited to maintaining a palette of exactly  $k$  predictions. This allows us in [Section 10.5.3](#) to design an algorithm that achieves competitive guarantees for all values of  $k$  simultaneously. Another issue with a  $k$ -server strategy, is that existing algorithms for the  $k$  server problem can be computationally intensive. The work function algorithm, for example, requires solving a max-flow instance on each day  $t$  with  $O(t)$  vertices. This causes the work to scale badly as the number of days grows large. Finally, some algorithms for the  $k$ -server problem work well in the setting where the number of points  $n$  in the metric space  $\mathbf{S}$  is finite. A notable example is the algorithm of [Bansal, Buchbinder, Madry, and Naor \[2015\]](#) which gives a randomized algorithm that achieves competitive ratio of  $\tilde{O}(\log^2 k \log^3 n)$  against an oblivious adversary. However, this is not ideal for our applications of interest, in which the solution space  $\mathbf{S}$  is often a space of vectors  $\subseteq \mathbb{R}^d$  with distance  $\mathbf{d}(u, v) = \|u - v\|_q$  corresponding to a relevant norm  $q$ .

### 10.5.3 Algorithm with Improved Runtime

In this section we show an algorithm that is  $O(k^4 \ln^2 k)$ -competitive against any set of  $k$  trajectories. Furthermore, the algorithm is deterministic (and thus resistant to an adaptive adversary), and is oblivious to the choice of  $k$ . That is, the competitive ratio holds for all  $k$  simultaneously. Importantly, this algorithm also has total runtime bounded that is  $O(1)$  times the sum of radii searched. We note that the techniques we use in this algorithm are significantly different than those in the reduction to  $k$ -server ([Section 10.5.2](#)).

[Algorithm 10.3](#) generalizes the single trajectory algorithm. For the single trajectory case, on each day the algorithm searched from “yesterday’s solution.” For multiple trajectories, this is no longer enough, as the previous day’s solution could come from some other trajectory,

and be arbitrarily far away from today's solution.

[Algorithm 10.3](#) addresses this by searching from *all* previous solutions in parallel. These “threads” are run at approximately harmonic rates, i.e. the thread of the  $i$ th most recent solution is run at rate  $\frac{1}{i^2 \ln^2 i}$ . This leaves the issue that the previous solution from the same trajectory as today could be arbitrarily far in the past, and be run at an extremely slow rate. This is addressed by “pruning” threads that are no longer fruitful. That is, when the ball searched around a solution  $i$  fully contains the ball searched around a previous solution  $j$ , the algorithm can stop running thread  $j$ , as that work is redundant with thread  $i$ . In this case, we say thread  $i$  “subsumes” thread  $j$ . Then, we can increase the rates of the threads that are lower priority than  $j$ , and maintain that there is at most one thread being run at each rate  $\frac{1}{i^2 \ln^2 i}$  for each integer  $i$ . In the analysis we show that either the algorithm runs long enough for the previous solution from the same trajectory as today to be elevated to rate  $\geq \frac{1}{k^2 \ln^2 k}$  and eventually solve the instance, or the algorithm terminates more quickly than that, which is even better. This allows us to bound the competitive ratio by  $O(k^4 \ln^2 k)$ .

The main runtime overhead in [Algorithm 10.3](#) comes from checking when to prune the slower threads. For each step that the algorithm takes of a thread, it must check whether the thread has been subsumed. For a thread running at rate  $\frac{1}{i^2 \ln^2 i}$  it must check the  $O(i)$  threads that are faster than it on each step. Thus, even though the thread is running steps of the subroutine  $\mathcal{A}$  at rate  $\frac{1}{i^2 \ln^2 i}$ , it is doing total work at rate  $O(i) \cdot \frac{1}{i^2 \ln^2 i} = O(\frac{1}{i \ln^2 i})$ . The rates are chosen so that this series converges, and the total work of the algorithm can be bounded by  $O(1)$  times the work it does for the fastest thread.

**Definition 10.5.8** (Subsumes). *If thread  $i$  kills thread  $j$  ([Algorithm 10.3](#), line 18), then we say that thread  $i$  subsumes thread  $j$ . If thread  $j$  previously subsumed thread  $h$ , then thread  $i$  now also subsumes thread  $h$ .*

**Definition 10.5.9** (Radius of a thread). *For a thread  $i$  at some given time,  $\text{radius}(i)$  is the*

**Algorithm 10.3** Quadratic Decay

---

```

1: for day  $t$ , instance  $I_t$  arrives do
2:
3:   Initialize linked-list of active threads
4:   for  $t' : t \geq t' \geq 1$  do
5:     Create thread  $t'$  to run  $\mathcal{A}(I_t, S_{t'})$ 
6:      $\text{radius}(t') := 0$ 
7:     Append thread  $t'$  to list of active threads
8:   end for
9:
10:  Run active threads in parallel,
11:    with  $i$ th active thread in list running at rate  $\frac{1}{i^2 \ln^2 i}$  (rate 1 for  $i = 1$ )
12:    track radius of each thread
13:  while no thread completed do
14:    for step taken of thread running at  $i$ th rate do
15:       $t_1 :=$  thread associated with this rate,
16:        found by walking down linked list of active threads
17:      for active thread  $t_2$  with faster rate than  $t_1$  do
18:        if  $d(S_{t_1}, S_{t_2}) \leq \text{radius}(t_2) - \text{radius}(t_1)$  then
19:          Kill thread  $t_1$ 
20:          Remove thread  $t_1$  from linked list of active threads
21:        end if
22:      end for
23:    end for
24:  end while
25:
26:   $S_t =$  solution of thread that completed
27:  Output  $S_t$ 
28: end for

```

---

number of steps that thread  $i$  has been run so far. If thread  $i$  subsumes thread  $j$ , for analysis we will set  $\text{rate}(j) = \text{rate}(i)$ . We also continue updating the radius of  $j$  i.e.,  $\text{radius}(j)$  continues to increase at  $\text{rate}(j)$ .

**Definition 10.5.10** (Virtual radius). The virtual radius of [Algorithm 10.3](#) is the radius of the highest rate thread.

We will show that the virtual radius of the algorithm is within  $O(1)$  both of the total radius searched ([Lemma 10.5.13](#)) and the total runtime  $O(1)$  ([Lemma 10.5.14](#)), making it a useful

abstraction.

We define the subsuming condition so that thread  $i$  subsumes thread  $j$ , exactly when the ball of radius  $\text{radius}(i)$  around  $S_i$  fully contains the ball of radius  $\text{radius}(j)$  around  $S_j$ . Thus the part of  $\mathbf{S}$  that has been searched by thread  $j$ , has also been searched by thread  $i$ , and the work of thread  $j$  is redundant. The following lemma shows that this invariant is maintained over the run of the algorithm.

**Lemma 10.5.11** (Subsuming identity). *If thread  $i$  subsumes thread  $j$ , then*

$$\mathbf{d}(S_i, S_j) \leq \text{radius}(i) - \text{radius}(j).$$

*Proof.* We show this via induction on the number of times the “kill condition” (Line 18) is triggered in Algorithm 10.3. In the base case, at the outset of the algorithm, no thread subsumes any other thread, so the statement holds vacuously.

Now, consider the  $s$ th kill event. Consider a thread  $i$  that subsumes a thread  $j$  at this point (not necessarily the threads that just triggered the kill condition). There are two cases.

*Case 1.* Thread  $i$  subsumed thread  $j$  previously, at the  $(s - 1)$ th kill event. This means that  $\text{rate}(j) = \text{rate}(i)$ , so  $\text{radius}(i)$  and  $\text{radius}(j)$  increased by the same amount since the previous kill event. Thus,

$$\mathbf{d}(S_i, S_j) \leq \text{radius}(i) - \text{radius}(j)$$

continues to hold from the induction hypothesis.

*Case 2.* Thread  $i$  killed thread  $j$ , or thread  $i$  killed a thread  $h$  that subsumes thread  $j$ , at the



sth kill event.

If thread  $i$  killed thread  $j$ , then by the kill condition (Algorithm 10.3, line 18), we have that

$$\mathbf{d}(S_i, S_j) \leq \text{radius}(i) - \text{radius}(j).$$

If thread  $i$  killed a thread  $h$  that subsumes thread  $j$ , then the kill condition gives us that

$$\mathbf{d}(S_i, S_h) \leq \text{radius}(i) - \text{radius}(h).$$

Since thread  $h$  already subsumes thread  $j$ , Case 1 gives us that

$$\mathbf{d}(S_h, S_j) \leq \text{radius}(h) - \text{radius}(j).$$

Therefore, we have

$$\begin{aligned} \mathbf{d}(S_i, S_j) &\leq \mathbf{d}(S_i, S_h) + \mathbf{d}(S_h, S_j) && \text{triangle inequality} \\ &\leq \text{radius}(i) - \text{radius}(h) + \text{radius}(h) - \text{radius}(j) \\ &\leq \text{radius}(i) - \text{radius}(j). \end{aligned}$$

□

The previous lemma implies that if a thread  $j$  is subsumed by a thread  $i$ , then thread  $i$  is at least as effective at finding that day's solution as thread  $j$ .

**Lemma 10.5.12.** *If a subsumed thread  $i$  would have completed, i.e.,*

$$\mathbf{d}(S_i, S_t) \leq \text{radius}(i),$$

then there must be some active thread of [Algorithm 10.3](#) that has completed.

*Proof.* Let  $i^*$  be the thread that subsumes thread  $i$ . We have that

$$\begin{aligned}
 \mathbf{d}(S_{i^*}, S_t) &\leq \mathbf{d}(S_{i^*}, S_i) + \mathbf{d}(S_i, S_t) && \text{triangle inequality} \\
 &\leq \mathbf{d}(S_{i^*}, S_i) + \text{radius}(i) \\
 &\leq \text{radius}(i^*) && i^* \text{ subsumes } i, \text{ [Lemma 10.5.11](#),}
 \end{aligned}$$

and therefore, thread  $i^*$  completed. □

Because we choose the rates of the threads to form a converging series, it is sufficient to bound the virtual radius of the algorithm, as it is within  $O(1)$  of the total radius searched by the algorithm.

**Lemma 10.5.13** (Total radius in terms of virtual radius). *The sum of the radii searched by [Algorithm 10.3](#) is at most  $O(1)$  times the virtual radius at the end of [Algorithm 10.3](#).*

*Proof.* Denote the virtual radius at the end of [Algorithm 10.3](#) by  $v$ . The virtual radius is the radius of the highest rate thread (rate 1). Thus, the highest rate thread contributes  $v$  to the total work.

Over the run of [Algorithm 10.3](#), at any given point, there is at most one thread per  $i \geq 2, i \in \mathbb{N}$  with rate equal to  $\frac{1}{i^2 \ln^2 i}$ . Consider the amount the thread(s) with rate  $\frac{1}{i^2 \ln^2 i}$  increase the total radius searched over time. The ratio of the radius searched by this thread to the highest rate thread is  $\frac{1}{i^2 \ln^2 i}$ . So, over the time that the highest rate thread accrued radius  $v$ , this thread has increased the total radius searched by  $\frac{v}{i^2 \ln^2 i}$ .

Summing over all threads, we can bound the total radius searched by

$$v + \sum_{i=2}^{\infty} \frac{v}{i^2 \ln^2 i} < 2v = O(1) \cdot v.$$

□

Now, we account for the overhead of running the algorithm. We note that it is not only the total radius searched that contributes to the work done by the algorithm. We must be careful to account for the work that it takes to check for when threads are subsumed.

**Lemma 10.5.14** (Runtime in terms of total radius). *On a given day  $t$ , the runtime of Algorithm 10.3 is at most  $O(1)$  times the total radius searched at the end of Algorithm 10.3.*

*Proof.* We assume that each operation of running a thread for “one step” (Algorithm 10.3, Line 11) can be done in  $O(1)$  time. In particular we note that the interleaving schedule of the threads (based on rate) is fixed and does not depend on the input. Thus it can be computed in advance.

For each step that the  $i$ th fastest thread is run (Algorithm 10.3, Line 11), the algorithm performs the following overhead:

- The algorithm steps through the linked list to find the associated thread in  $O(i)$  time.
- The algorithm checks the thread against all faster threads to see if it has been subsumed for each of the  $i - 1$  faster threads. This requires one distance query for each of the faster threads. By Assumption 10.2.2, we have that this takes  $O(1)$  time.
- If the thread has been subsumed, the algorithm removes it from the linked-list of active

threads in  $O(1)$  time.

Thus, each step of this thread is accompanied by  $O(i)$  work.

Denote the virtual radius at the end of [Algorithm 10.3](#) by  $v$ . Over the run of [Algorithm 10.3](#), the total work done by the thread(s) that run at the  $i$ th fastest rate is

$$v \cdot \frac{1}{i^2 \ln^2 i} \cdot O(i) = v \cdot O\left(\frac{1}{i \ln^2 i}\right).$$

Summing over all threads, we have that the total work can be bounded by

$$v + \sum_{i=2}^{\infty} v \cdot O\left(\frac{1}{i \ln^2 i}\right) = O(1) \cdot v.$$

By [Lemma 10.5.13](#), we have that  $v$  is within  $O(1)$  of the sum of the radii searched by [Algorithm 10.3](#). Thus, the total work done by [Algorithm 10.3](#) is at most  $O(1)$  times the sum of the radii that it searches.  $\square$

A key part of our analysis is showing that even if all solutions close to today's solution  $S_t$  were seen very far in the past, our algorithm will eventually raise the rates of these threads to a reasonable amount. We do this by showing that enough other threads will be subsumed, clearing the way for threads lower down the chain.

**Lemma 10.5.15** (Slow threads have bounded lifetime). *Let  $i$  be a thread that is running at rate  $\geq \frac{1}{(k-1)^2 \ln^2(k-1)}$  in [Algorithm 10.3](#) at some point. Let  $j$  be another thread running at a slower rate than  $i$ . Thread  $j$  is subsumed in at most  $k^3 \ln^2 k \cdot \mathbf{d}(S_i, S_j)$  additional units of virtual radius.*

*Proof.* Assume for the sake of contradiction that  $j$  is not subsumed in the next  $k^3 \ln^2 k \cdot$

$\mathbf{d}(S_i, S_j)$  units of virtual radius. For  $i$  faster than  $j$ , and thread  $j$  not subsumed, [Algorithm 10.3](#) maintains that  $\text{rate}(i) > \text{rate}(j)$ . Thus,  $(\text{radius}(i) - \text{radius}(j))$  is always nonnegative, and monotonically nondecreasing with virtual radius. Over the next  $k^3 \ln^2 k \cdot \mathbf{d}(S_i, S_j)$  units of virtual radius, we have that

$$\Delta(\text{radius}(i) - \text{radius}(j)) \geq k^3 \ln^2 k \cdot \mathbf{d}(S_i, S_j) (\text{rate}(i) - \text{rate}(j)),$$

where  $\Delta(\text{radius}(i) - \text{radius}(j))$  is the change in  $\text{radius}(i) - \text{radius}(j)$ . Since  $\text{rate}(i) \geq \frac{1}{(k-1)^2 \ln^2(k-1)}$  and  $i$  is faster than  $j$ , we have that

$$\begin{aligned} \text{rate}(i) - \text{rate}(j) &\geq \frac{1}{(k-1)^2 \ln^2(k-1)} - \frac{1}{k^2 \ln^2 k} \\ &\geq \frac{1}{(k-1)^2 \ln^2 k} - \frac{1}{k^2 \ln^2 k} \\ &= \frac{2k+1}{k^2(k-1)^2 \ln^2 k} \\ &\geq \frac{1}{k^3 \ln^2 k}. \end{aligned}$$

Thus, after  $k^3 \ln^2 k \cdot \mathbf{d}(S_i, S_j)$  units of virtual radius, we have that

$$\Delta(\text{radius}(i) - \text{radius}(j)) \geq \mathbf{d}(S_i, S_j)$$

$$\text{radius}(i) - \text{radius}(j) \geq \mathbf{d}(S_i, S_j).$$

If thread  $i$  is still active at the end of these iterations, this implies that thread  $i$  kills thread  $j$  ([Algorithm 10.3](#), line 18). Otherwise let thread  $i^*$  be the active thread that subsumes thread  $i$ . By [Lemma 10.5.11](#), we have that

$$\mathbf{d}(S_i, S_{i^*}) \leq \text{radius}(i^*) - \text{radius}(i).$$

Thus, we get that

$$\begin{aligned}
\mathbf{d}(S_{i^*}, S_j) &\leq \mathbf{d}(S_{i^*}, S_i) + \mathbf{d}(S_i, S_j) \\
&\leq \text{radius}(i^*) - \text{radius}(i) + \text{radius}(i) - \text{radius}(j) \\
&\leq \text{radius}(i^*) - \text{radius}(j),
\end{aligned}$$

which implies that thread  $i^*$  will kill thread  $j$ . Thus, thread  $j$  must be subsumed, and we reach a contradiction.  $\square$

**Theorem 10.5.16** (Competing with  $k$  trajectories online in radius *and* runtime). *Algorithm 10.3 is  $O(k^4 \ln^2 k)$ -competitive with any set of  $k$  trajectories in the total radius that it searches. Furthermore, the total runtime of the algorithm over  $T$  days can be bounded by  $O(1)$  times the total radius that it searches.*

*Proof.* Fix a collection of trajectories  $(\{P_t : t \in \mathcal{T}^{(1)}\}, \dots, \{P_t : t \in \mathcal{T}^{(k)}\})$ . We begin by bounding the cost of Algorithm 10.3 on days that correspond to trajectory 1.

Consider a particular day  $t$  that corresponds to trajectory 1. We claim that eventually, if the algorithm runs long enough, there will be at most  $k - 1$  active threads with higher rates than thread  $\text{prev}^{(1)}(t)$ .

We proceed iteratively. If there are  $\leq k - 1$  active threads with higher rates than thread  $\text{prev}^{(1)}(t)$ , we are done. Otherwise, of the  $k$  threads with rate  $1, \dots, \frac{1}{k^2 \ln^2 k}$ , there must be two of them that belong to the same trajectory  $q$ . Let  $\mathcal{T}_t^{(q)}$  be the set of days between  $\text{prev}^{(1)}(t)$  and  $t$  that are associated with trajectory  $q$ . We can bound the total distance between any

two solutions in  $\mathcal{T}_t^{(q)}$  by the total distance the  $q$ th trajectory moves in this time as

$$\leq \sum_{t' \in \mathcal{T}_t^{(q)} \setminus \{\text{prev}^{(q)}(t)\}} \mathbf{d}(t', \text{next}^{(q)}(t')).$$

Let  $t^{(q)*}$  be the thread of  $\mathcal{T}_t^{(q)}$  with the highest rate. By the selection of  $q$ , we know that  $t^{(q)*}$  has rate at least  $\frac{1}{(k-1)^2 \ln^2(k-1)}$ . By [Lemma 10.5.15](#), this means that all other members of  $\mathcal{T}_t^{(q)}$  will be subsumed in

$$\leq k^3 \ln^2 k \cdot \sum_{t' \in \mathcal{T}_t^{(q)} \setminus \{\text{prev}^{(q)}(t)\}} \mathbf{d}(t', \text{next}^{(q)}(t'))$$

additional units of virtual radius. Thus, after this many units of virtual radius, there can be at most one thread belonging to trajectory  $q$  that has a higher rate than  $\text{prev}^{(1)}(t)$ . We say that trajectory  $q$  has “collapsed” between  $\text{prev}^{(1)}(t)$  and  $t$ .

Iterating this argument at most  $k - 1$  times gives us that after at most

$$k^3 \ln^2 k \cdot \sum_{q=2}^k \sum_{t' \in \mathcal{T}_t^{(q)} \setminus \{\text{prev}^{(q)}(t)\}} \mathbf{d}(t', \text{next}^{(q)}(t'))$$

iterations of the algorithm, there can be at most  $k - 1$  active threads with higher rate than thread  $\text{prev}^{(1)}(t)$ . Thus, thread  $\text{prev}^{(1)}(t)$  has rate  $\geq \frac{1}{k^2 \ln^2 k}$ .

This means that in at most  $k^2 \ln^2 k \cdot \mathbf{d}(S_{\text{prev}^{(1)}(t)}, S_t)$  additional units of virtual radius, we have that

$$\mathbf{d}(S_{\text{prev}^{(1)}(t)}, S_t) \leq \text{radius}(\text{prev}^{(1)}(t)).$$

Therefore, by [Lemma 10.5.12](#), there must be some thread that solves day  $t$  in at most

$$k^3 \ln^2 k \cdot \sum_{q=2}^k \sum_{t' \in \mathcal{T}_t^{(q)} \setminus \{\text{prev}^{(q)}(t)\}} \mathbf{d}(t', \text{next}^{(q)}(t')) + k^2 \ln^2 k \cdot \mathbf{d}(S_{\text{prev}^{(1)}(t)}, S_t)$$

units of virtual radius, and by [Lemma 10.5.13](#) the virtual radius of an algorithm is within a  $O(1)$  factor of the total radius searched.

Now, we can sum the total radius searched over all days  $t$  belonging to trajectory 1. The total radius searched over these days can be bounded by

$$\begin{aligned} &\leq \sum_{t \in \mathcal{T}^{(1)}} \left[ O(k^3 \ln^2 k) \cdot \sum_{q=2}^k \sum_{t' \in \mathcal{T}_t^{(q)} \setminus \{\text{prev}^{(q)}(t)\}} \mathbf{d}(t', \text{next}^{(q)}(t')) + O(k^2 \ln^2 k) \cdot \mathbf{d}(S_{\text{prev}^{(1)}(t)}, S_t) \right] \\ &\leq O(k^3 \ln^2 k) \sum_{q=1}^k \sum_{t \in \mathcal{T}^{(q)}} \mathbf{d}(S_{\text{prev}^{(q)}(t)}, S_t) \\ &\leq O(k^3 \ln^2 k) \cdot \sum_{q=1}^k \text{cost}(\{P_t : t \in \mathcal{T}^{(q)}\}), \end{aligned}$$

where the last line follows by [Corollary 10.5.5](#) because  $\sum_{t \in \mathcal{T}^{(q)}} \mathbf{d}(S_{\text{prev}^{(q)}(t)}, S_t)$  is the cost of “Predict yesterday’s solution” run on trajectory  $q$  alone.

Finally, the same bound holds for all  $k$  trajectories, so we can conclude that the total radius searched by [Algorithm 10.3](#) over all days is

$$\leq O(k^4 \ln^2 k) \cdot \sum_{q=1}^k \text{cost}(\{P_t : t \in \mathcal{T}^{(q)}\}),$$

and therefore [Algorithm 10.3](#) is  $O(k^4 \ln^2 k)$  competitive against the  $k$  trajectories. Finally, [Lemma 10.5.14](#) tells us that the total runtime of [Algorithm 10.3](#) is at most  $O(1)$  times the total radius searched by [Algorithm 10.3](#). □



*Remark 10.5.17.* Instead of running the threads at quadratically decaying rates, i.e., the  $i$ th fastest thread runs at rate  $\frac{1}{i^2 \ln^2 i}$ , we could run them at harmonically decaying rates, i.e., the  $i$ th fastest thread runs at rate  $\frac{1}{i \ln^2 i}$ . This would result in an improved competitive ratio of  $O(k^3 \ln^2 k)$  against any  $k$  trajectories, in the sum of the radii searched (number of steps taken of the subroutine  $\mathcal{A}$ ). However, the total runtime could scale as badly as an  $O(T)$  factor times the sum of the radii, where  $T$  is the time horizon.

The competitive ratio of the harmonic rate decay strategy is still worse than that of the reduction to  $k$ -server, which achieves competitive ratio  $O(k^2)$  ([Corollary 10.5.7](#)). However, the rate decay strategy also has the benefit that it is oblivious to the setting of  $k$ , and therefore achieves the guarantee for all  $k$  simultaneously.

## 10.6 Future Directions

Our work leaves open a number of interesting research directions.

- Investigate whether it is possible to get more efficient algorithms for settings with a specific structure. In this work, we considered the setting where we can group instances using a  $k$ -wise partition. Are there other forms of structure that we can take advantage of?
- Provide an algorithm with an improved competitive ratio for the online setting, or provide a lower bound for this setting. Currently, the best lower bound we have is  $\Omega(k)$ , which carries over from the offline setting ([Remark 10.3.6](#)). It would also be interesting to match the competitive ratio  $O(k^2)$  of the  $k$ -server based algorithm, but achieve it for all  $k$  simultaneously.
- Find other warm-start and/or local-search type settings to which our techniques are

applicable.

- In this work, we define the Online Ball Search problem ([Definition 10.5.1](#)), and investigate some connections to the  $k$ -server problem. Can algorithms for Online Ball Search be transformed into algorithms for  $k$ -server? Alternatively, can we show a separation between these two problems?

## 10.7 Chapter Acknowledgements

This chapter is based on joint work with Avrim Blum. Part of this work was supported by the IDEAL (Institute for Data, Econometrics, Algorithms, and Learning) summer exchange program.

# Chapter 11

## Conclusion and Outlook

In this thesis we developed a new toolkit for designing reliable methods that provably extract value from unreliable black-box models. We explored two main approaches. The first approach was a task-agnostic approach that “wraps” a black-box model in a layer that ensures reliability for any downstream task. This built on the paradigm of *conformal prediction*, and provides distributional guarantees. The second approach was task-specific, and showed how to use black-box methods to achieve speed ups when solving sequences of instances of the same task. This built on the paradigms of *algorithms with predictions* and *data-driven algorithm design*, and provided both distributional and worst-case single-shot guarantees.

We see many exciting directions for future exploration, that build on the ideas in this thesis. In [Section 11.1](#), we discuss directions related to the first approach and volume-optimal conformal prediction. In [Section 11.2](#), we discuss directions related to the second approach and solving sequences of related instances. Finally in [Section 11.3](#), we discuss other promising approaches, and connections to other lines of work in algorithms and theoretical machine learning.

## 11.1 Outlook for Volume-Optimal Conformal Prediction

In this thesis we formulated the problem of volume-optimal conformal prediction, and designed the first statistically and computationally efficient methods for this problem, to the best of our knowledge.

We introduced a novel toolkit to address this problem in the basic, high-dimensional, and online settings. We also showed connections between our problem and other fundamental problems in algorithms and statistics such as robust estimation, and robust subspace recovery.

Going forward, we see this as the beginning of an area of study that has many promising directions to expand.

**Other set families  $\mathcal{C}$ .** There are many other set families  $\mathcal{C}$  for which it would be interesting to design computationally efficient methods to learn prediction sets. In this work, in the high dimensional setting, we provide methods to learn Euclidean balls and ellipsoids. Other interesting classes to study include  $L_p$  balls, products of intervals (axis aligned boxes), or perhaps even neural networks. In the high-dimensional setting, the work in this thesis gives two algorithmic approaches— through robust-estimation, and through the primal-dual structure of the algorithmic problem of enclosing all of the samples. It would be interesting to see one of these paradigms, or a new paradigm, could extend broadly to a large family of hypothesis classes  $\mathcal{C}$ . Broadly, we phrase the following questions.

**Question 11.1.1.** *What prediction set families  $\mathcal{C}$  can be learned computationally and statistically efficiently?*

**Applications to other statistical problems.** Our algorithms for finding volume-optimal prediction sets can be seen as a subroutine that is useful for identifying dense regions of arbitrary distributions. This primitive has already been useful for us in designing an algorithm for robust subspace recovery. We believe that this subroutine has promise in tackling many other high-dimensional robust estimation questions, in a way that does not make strong parametric assumptions.

Some works have also made connections between conformal prediction and other problems such as *risk-averse decision making* [Kiyani, Pappas, Roth, and Hassani, 2025], and *chance-constrained optimization* [Zhao, Yu, Sesia, Deshmukh, and Lindemann, 2026]. It would be interesting to see whether volume-optimal conformal techniques can be used to achieve optimality in these other related settings.

**Other notions of efficiency.** In this thesis, we have been concerned with using the volume, or Lebesgue measure, of a prediction set as a measure of efficiency, or how informative it is. This models settings where, for example, the prediction set captures a set of alternatives that have to be tried in some downstream task.

Depending on the task, there may be other measures of efficiency that are more suitable. For example, one may want to measure prediction sets according to a prescribed measure (instead of the Lebesgue measure that we consider in this work), or according to the specific cost of some downstream decision. It would be interesting to design methods that explicitly optimize for these specific cost models.

**Statistical limits of learning prediction sets.** We still do not have a full characterization of when it is possible to learn an approximately volume-optimal prediction set from a family  $\mathcal{C}$  statistically efficiently. We provide some partial results along this line in [Chapter 7](#).

It would be very interesting to resolve this question for the distribution-free setting. It would also be interesting investigate whether this problem becomes more tractable in a beyond-worst-case setting, when we assume extra properties about the underlying distribution  $P$ , for example, in the *smoothed analysis* setting.

**Extension to the supervised setting.** We have studied some ways to lift our algorithms for the unsupervised setting to the supervised setting, by predicting uncertainty sets for the residuals, or training a black-box estimator for the conditional distribution (Section 3.3). It would be interesting to investigate what kind of volume-optimal guarantees are possible in the supervised setting.

## 11.2 Outlook for Solving Sequences of Instances

In this thesis, we explored two different ways of incorporating unreliable black-boxes to solve specific tasks more efficiently. The first strategy used the framework of *dynamic algorithms*, and we showed how to use unreliable predictions to lift algorithms from a *promise* setting, where they are guaranteed accurate knowledge of future updates, to a *predictions* setting, where the information of future updates is subject to error.

The second strategy used *warm start* algorithms, a type of algorithm with prediction, to show that it is possible to solve sequences of instances faster than one at a time, by *learning* information about the sequence, when the sequence has some structure.

These lead to a variety of interesting directions to explore in the future.

**From promise problems to unreliable black-boxes.** In Chapter 9, we showed how to “lift” incremental, decremental, and offline algorithms to the fully dynamic setting using unreliable predictions of future updates. It would be interesting to explore such transforma-

tions in other settings. There are many other algorithmic tasks that have been studied under *promise* settings, where the algorithm has access to some extra information in addition to the instance, to help it solve the problem. These problems are potentially good candidates for an algorithms with predictions approach.

**Question 11.2.1.** *What other algorithms for promise problems can be lifted to the case where the promised information is noisy or unreliable?*

**End-to-end guarantees for other tasks.** In [Chapter 10](#), we establish end-to-end guarantees for solving sequences of instances of a task, for which we have a *warm start* algorithm. It would be interesting to establish similar guarantees for other kinds of algorithms, that take advantage of unreliable information in other ways.

**Question 11.2.2.** *What other algorithmic subroutines can be used to solve sequences of related instances more efficiently than one at a time?*

**Incorporating black-box machine learning models in data-driven algorithms.** In [Chapter 10](#), we design end-to-end algorithms that see sample instances from a distribution or sequence, and learn predictions that are used with a *warm start* algorithm. We are able to argue that this can result in gains over solving each individual instance from scratch. In this section, we mostly study specific learning algorithms, or prescribed methods such as *empirical risk minimization* (ERM). We see this as a proof of concept that there is something to be gained by learning. It would be interesting to incorporate black-box models into this process, to potentially gain over the methods that we have guarantees for. This direction applies to many settings in data-driven algorithms, where typically we study metrics like ERM, which are not always computationally efficient, and could potentially be aided by black-box approaches.

**Question 11.2.3.** *Are there ways to incorporate black-box predictions into data-driven al-*

*gorithms, either to achieve stronger performance gains for the algorithmic task, or to make the process of learning predictions more efficient?*

**Warm start algorithms: subroutines and applications.** Chapter 10 uses *warm start* algorithms with predictions as a subroutine to solve sequences of instances. Despite being an early paradigm of algorithms with predictions, this paradigm of using a predicted solution to speed up an algorithm that finds the true solution has not seen as many results as other areas of algorithms with predictions. We believe that part of this is motivation— it seemed like a very strong assumption to be able to get such a large and informative prediction. In Chapter 10, we show that this paradigm is indeed powerful, and can be used, even in theory, to achieve large gains in solving sequences of related instances. Given this, it is well motivated to ask what other tasks admit such guarantees.

**Question 11.2.4.** *What algorithmic tasks admit a warm start algorithm with predictions?*

We believe that, to start, many algorithms that follow a *local search* paradigm, can be phrased as *warm start* guarantees. This could include powerful techniques such as *convex optimization*, *primal-dual algorithms*, and more. It would be useful to see whether such interpretations can be codified.

## 11.3 Future Directions for Reliable Machine Learning with Unreliable Black Boxes

In this thesis, we studied two main approaches to achieve reliability with black-box models. The first was a task-agnostic strategy that “wrapped” a black box model in a layer that ensures reliability, that built on the ideas of *conformal prediction*. The second was task-specific strategies, that built on the ideas of *algorithms with predictions* and *data-driven*



*algorithm design*. In addition to these, there have been many other interesting paradigms that have been studied in the last few years to address the issue of *reliable* machine learning. A very promising direction is to extend these lines of work to provide methods that use black-boxes, and provide provable reliability guarantees. We believe that the new frameworks and methods that we develop in this thesis can serve as a foundation on which to extend this theory to many other exciting settings. We sketch a few potential directions below.

**Distribution shift.** Reliability of black-box models under *distribution shift* is problem of huge practical importance. This problem arises in many high stakes settings, including medical settings and self-driving cars, where classifiers trained on one population may be deployed on another. This has led to an exciting line of theory and empirical work addressing distribution shift.

Recent work has highlighted two broad approaches to designing machine learning methods with provable guarantees under distribution shift. One approach is to provide machine learning methods that satisfy provable guarantees. For this to be tractable, we must often either settle for weak guarantees, like in *domain adaptation* [David, Lu, Luu, and Pal, 2010] or *distribution-free PQ learning* [Goldwasser, Kalai, Kalai, and Montasser, 2020], or make strong distributional assumptions, like in *distribution-specific testable learning with distribution shift* [Klivans, Stavropoulos, and Vasilyan, 2024].

Another approach, is to take a black-box predictor, and design methods to estimate its accuracy on data from the test distribution. This can be used to decide how much stock to place in the black-box predictions. At a high level, this is the idea behind strategies like *prediction-powered inference* [Angelopoulos, Bates, Fannjiang, Jordan, and Zrnic, 2023b], and *average thresholded confidence* [Garg, Balakrishnan, Lipton, Neyshabur, and Sedghi, 2022]. These methods will work very well if the black-box model is highly accurate, but may

result in a trivial guarantee in the worst-case.

There is an opportunity here to design methods that gracefully tradeoff between the two extremes.

**Question 11.3.1.** *Can we design efficient methods for reliable predictions under distribution shift, that adaptively transition between black-box based guarantees and distributional/worst-case guarantees, based on which one is better suited to the data?*

**Robust Statistics.** *Robust statistics* is a paradigm that aims to deal with the presence of adversarial training examples. Typically, in this setting, we are given access to corrupted dataset of  $n$  samples. For some corruption parameter  $\alpha$ , we are promised that  $1 - \alpha$  fraction of the samples come from some true inlier distribution  $\mathcal{D}$ . The other  $\alpha$  fraction are arbitrary adversarial corruptions. The goal of the algorithm is to, without knowing which samples are which, train a classifier that performs well on the uncorrupted distribution  $\mathcal{D}$  [see the book of [Diakonikolas and Kane, 2023](#), for an overview of this area]. Such worst-case methods are limited in the kinds of guarantees that they can provide, and often have to make strong distributional assumptions on the inlier distribution  $\mathcal{D}$  for learning to be possible, and for computational efficiency.

Perhaps a black-box predictor could be helpful to make this process more efficient.

**Question 11.3.2.** *Can we design a robust regression algorithm, that takes in a black-box predictor  $f(\cdot)$ , and has performance that degrades gracefully as a function of the black-box predictor  $f(\cdot)$ 's accuracy on the inlier distribution  $\mathcal{D}$ ?*

**Calibration.** A recent rapidly growing line of work has addressed the reliability of machine learning models via *calibration*. A machine learning model  $f(X)$  that predicts a target variable  $Y$  is said to be *calibrated* if, conditioned on the model outputting a particular

prediction  $\hat{y}$ , the expected value of  $Y$  is  $\hat{y}$ ,

$$\forall \hat{y}, \text{ s.t. } f(X) \text{ supported on } \hat{y}, \quad \mathbb{E}_{(X,Y)} [Y|f(X) = \hat{y}] = \hat{y}.$$

Calibration has nice properties in economics, decision theory, and machine learning, and at a high level, allows a downstream user to treat the model as an *unbiased* estimator for  $Y$  [Foster and Vohra, 1998]. While calibration has been studied for decades, it has recently become a popular avenue to evaluate the trustworthiness of machine learning models [Wang, 2025]. This has led to a recent line of theoretical work designing metrics for measuring calibration (i.e., how far a model is from perfectly calibrated) [Błasiok, Gopalan, Hu, and Nakkiran, 2023, Hu and Wu, 2024], and designing online predictors that achieve low calibration errors [Foster and Vohra, 1998, Arunachaleswaran, Collina, Roth, and Shi, 2025, Hartline, Hu, and Wu, 2026].

However, to the best of our knowledge, this line of work has not addressed the problem of calibrating a black-box model online.

**Question 11.3.3.** *Given a black-box model  $f(\cdot)$  that predicts  $Y$  from  $X$ , can we generate predictions  $\hat{y}_1, \dots, \hat{y}_T$  online for an arbitrary sequence  $(X_1, Y_1), \dots, (X_T, Y_T)$ , such that the  $\hat{y}_i$  is well-calibrated with respect to the  $Y_i$ , and the  $\hat{y}_i$  achieve prediction accuracy competitive with  $f(X_1), \dots, f(X_T)$ ?*

**Distillation and certification.** An exciting new line of work has developed methods with theoretical guarantees for *model distillation* [Liu and Moitra, 2025, Golowich, Liu, and Shetty, 2026, 2025]. The goal is, given query access to a black-box generative model  $f(\cdot)$ , to learn a distilled representation  $\hat{f}(\cdot)$  that allows us to approximately sample from the distribution of  $f(\cdot)$ . Among other things, such an approach potentially allows one to turn a black-box model into a “white-box” model, since we have explicit access to the distilled

model  $\hat{f}(\cdot)$ .

Once we have explicit access to a model, we potentially have more leverage to evaluate and certify various reliability criteria.

**Question 11.3.4.** *Can we design model distillation methods that output “white-box” models  $\hat{f}(\cdot)$  that have certifiable reliability guarantees?*

Even if it is hard to provide reliability guarantees in the worst case, it would be interesting if the white-box representation allowed us to more easily evaluate how reliable the model is, or implement higher level reliability features.

**New modes of reliable machine learning.** Finally, the area of designing methods that use black-box machine learning models and provide provable guarantees is a nascent field. There are many more notions of reliability, and scope to design new frameworks that apply to problems of many different kinds. We hope that the work in this thesis can serve as a foundation to reason about what such frameworks can and should do, and give a toolkit for designing these methods.

# Bibliography

- Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 434–443. IEEE, 2014.
- Ittai Abraham, David Durfee, Ioannis Koutis, Sebastian Krinninger, and Richard Peng. On fully dynamic graph sparsifiers. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 335–344. IEEE, 2016.
- Ittai Abraham, Shiri Chechik, and Sebastian Krinninger. Fully dynamic all-pairs shortest paths with worst-case update-time revisited. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 440–452. SIAM, 2017.
- Arpit Agarwal and Eric Balkanski. Learning-augmented dynamic submodular maximization. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS ’24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.
- Pankaj Agarwal, Sariel Har Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. *Combinatorial and Computational Geometry*, 52, 11 2004.
- Sinem Ahipasaoglu. Fast algorithms for the minimum volume estimator. *Journal of Global Optimization*, 59(2–3):351–376, June 2014. doi: 10.1007/s10898-013-0116-5.
- Amir Ali Ahmadi, Dmitry Malioutov, and Ronny Luss. Robust minimum volume ellipsoids and higher-order polynomial level sets. In *NIPS Workshop on Optimization for Machine Learning*, Montréal, Québec, Canada, 2014.
- Ibrahim Umit Akgun, Ali Selman Aydin, Andrew Burford, Michael McNeill, Michael Arkhangelskiy, and Erez Zadok. Improving storage systems using machine learning. *ACM Trans. Storage*, 19(1), January 2023. ISSN 1553-3077. doi: 10.1145/3568429. URL <https://doi.org/10.1145/3568429>.
- Noga Alon and Yossi Azar. On-line steiner trees in the euclidean plane. In *Proceedings of the Eighth Annual Symposium on Computational Geometry*, SCG ’92, page 337–343, New

- York, NY, USA, 1992. Association for Computing Machinery. ISBN 0897915178. doi: 10.1145/142675.142744. URL <https://doi.org/10.1145/142675.142744>.
- Keerti Anand, Rong Ge, Amit Kumar, and Debmalya Panigrahi. Online algorithms with multiple predictions. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 582–598. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/anand22a.html>.
- Anastasios Angelopoulos, Emmanuel Candes, and Ryan J Tibshirani. Conformal pid control for time series prediction. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 23047–23074. Curran Associates, Inc., 2023a. URL [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/47f2fad8c1111d07f83c91be7870f8db-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/47f2fad8c1111d07f83c91be7870f8db-Paper-Conference.pdf).
- Anastasios N. Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Found. Trends Mach. Learn.*, 16(4):494–591, March 2023. ISSN 1935-8237. doi: 10.1561/22000000101. URL <https://doi.org/10.1561/22000000101>.
- Anastasios N. Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023b. doi: 10.1126/science.adi6000. URL <https://www.science.org/doi/abs/10.1126/science.adi6000>.
- Anastasios N. Angelopoulos, Rina Foygel Barber, and Stephen Bates. Theoretical foundations of conformal prediction, 2025. URL <https://arxiv.org/abs/2411.11824>.
- Anastasios Nikolas Angelopoulos, Rina Barber, and Stephen Bates. Online conformal prediction with decaying step sizes. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 1616–1630. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/angelopoulos24a.html>.
- Antonios Antoniadis, Christian Coester, Marek Elias, Adam Polak, and Bertrand Simon. Mixing predictions for online metric algorithms. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 969–983. PMLR, 23–29 Jul 2023. URL <https://proceedings.mlr.press/v202/antoniadis23b.html>.
- Eshwar Ram Arunachaleswaran, Natalie Collina, Aaron Roth, and Mirah Shi. *An Elemen-*

- tary Predictor Obtaining  $2 \sqrt{T} + 1$  Distance to Calibration*, pages 1366–1370. 2025. doi: 10.1137/1.9781611978322.41. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611978322.41>.
- Dionysis Arvanitakis, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Guarantees for alternating least squares in overparameterized tensor decompositions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=9FDErIfoVE>.
- Mihai Badoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 250–257, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581134959. doi: 10.1145/509907.509947. URL <https://doi.org/10.1145/509907.509947>.
- Ricardo Baeza-Yates, Joseph C. Culberson, and Gregory J. E. Rawlins. Searching in the plane. *Inf. Comput.*, 106:234–252, 1993. URL <https://api.semanticscholar.org/CorpusID:15209807>.
- Yu Bai, Song Mei, Huan Wang, Yingbo Zhou, and Caiming Xiong. Efficient and differentiable conformal prediction with general function classes. In *International Conference on Learning Representations*, 2022.
- Ainesh Bakshi and Pravesh K Kothari. List-decodable subspace recovery: Dimension independent error in polynomial time. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1279–1297. SIAM, 2021.
- Maria-Florina Balcan. *Data-Driven Algorithm Design*, page 626–645. Cambridge University Press, 2021.
- Nina Balcan, Mikhail Khodak, Dravyansh Sharma, and Ameet Talwalkar. Learning-to-learn non-convex piecewise-lipschitz functions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=USq7LP5pnDH>.
- Keith Ball. Volumes of sections of cubes and related problems. In *Geometric Aspects of Functional Analysis*, pages 251–260. Springer, Berlin, Heidelberg, 1989. doi: 10.1007/bfb0090058.
- Keith Ball. Volume ratios and a reverse isoperimetric inequality. *Journal of the London Mathematical Society*, 44:351–359, 1991. doi: 10.1112/jlms/s2-44.2.351.
- Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph (Seffi) Naor. A

- polylogarithmic-competitive algorithm for the k-server problem. *J. ACM*, 62(5), nov 2015. ISSN 0004-5411. doi: 10.1145/2783434. URL <https://doi.org/10.1145/2783434>.
- Rina Foygel Barber, Emmanuel J. Candès, Aaditya Ramdas, and Ryan J. Tibshirani. Conformal prediction beyond exchangeability. *The Annals of Statistics*, 2022. URL <https://api.semanticscholar.org/CorpusID:247158820>.
- Andrew R Barron. Uniformly powerful goodness of fit tests. *The Annals of Statistics*, 17(1): 107–124, 1989.
- Franck Barthe. On a reverse form of the brascamp-lieb inequality. *Inventiones mathematicae*, 134:335–361, 1997. URL <https://api.semanticscholar.org/CorpusID:14967794>.
- Surender Baswana, Sumeet Khurana, and Soumojit Sarkar. Fully dynamic randomized algorithms for graph spanners. *ACM Transactions on Algorithms (TALG)*, 8(4):1–51, 2012.
- Surender Baswana, Shreejit Ray Chaudhury, Keerti Choudhary, and Shahbaz Khan. Dynamic dfs in undirected graphs: Breaking the  $o(m)$  barrier. *SIAM Journal on Computing*, 48(4):1335–1363, 2019.
- Shai Ben-David, Nicolò Cesa-Bianchi, and Philip M. Long. Characterizations of learnability for classes of  $O, \dots, n$ -valued functions. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92*, page 333–340, New York, NY, USA, 1992. Association for Computing Machinery. ISBN 089791497X. doi: 10.1145/130385.130423. URL <https://doi.org/10.1145/130385.130423>.
- Jon Louis Bentley. Decomposable searching problems. *Inf. Process. Lett.*, 8(5):244–251, 1979.
- Jon Louis Bentley and James B Saxe. Decomposable searching problems i. static-to-dynamic transformation. *Journal of Algorithms*, 1(4):301–358, 1980.
- Thorsten Bernholt. Robust estimators are hard to compute. Technical Reports 2005,52, Technische Universität Dortmund, Sonderforschungsbereich 475: Komplexitätsreduktion in multivariaten Datenstrukturen, 2006. URL <https://ideas.repec.org/p/zbw/sfb475/200552.html>.
- Aaron Bernstein and Shiri Chechik. Incremental topological sort and cycle detection in expected total time. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 21–34. SIAM, 2018.
- Jan Betley, Jorio Cocola, Dylan Feng, James Chua, Andy Ardit, Anna Sztyber-Betley, and Owain Evans. Weird generalization and inductive backdoors: New ways to corrupt llms, 2025. URL <https://arxiv.org/abs/2512.09742>.



- Aditya Bhaskara, Aidao Chen, Aidan Perreault, and Aravindan Vijayaraghavan. Smoothed analysis in unsupervised learning via decoupling. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2019.
- Aditya Bhaskara, Eric Evert, Vaidehi Srinivas, and Aravindan Vijayaraghavan. New tools for smoothed analysis: Least singular value bounds for random matrices with dependent entries. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 375–386, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703836. doi: 10.1145/3618260.3649765. URL <https://doi.org/10.1145/3618260.3649765>.
- Rajendra Bhatia. *Matrix Analysis*, volume 169. Springer, 1997. ISBN 0387948465.
- Aadyot Bhatnagar, Huan Wang, Caiming Xiong, and Yu Bai. Improved online conformal prediction via strongly adaptive online learning. In *Proceedings of the 40th International Conference on Machine Learning, ICML’23*. JMLR.org, 2023.
- Sayan Bhattacharya and Janardhan Kulkarni. An improved algorithm for incremental cycle detection and topological ordering in sparse graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2509–2521. SIAM, 2020.
- Sayan Bhattacharya, Martin Costa, Silvio Lattanzi, and Nikos Parotsidis. Fully dynamic k-clustering in tildeo(k) update time. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=dnGEPkmnz0>.
- Jarosław Błasiok, Parikshit Gopalan, Lunjia Hu, and Preetum Nakkiran. A unifying theory of distance from calibration. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, page 1727–1740, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450399135. doi: 10.1145/3564246.3585182. URL <https://doi.org/10.1145/3564246.3585182>.
- Avrim Blum and Vaidehi Srinivas. *Competitive strategies to use “warm start” algorithms with predictions*, pages 3775–3801. 2025. doi: 10.1137/1.9781611978322.127. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611978322.127>.
- Avrim Blum, John Hopcroft, and Ravindran Kannan. *Foundations of Data Science*. Cambridge University Press, 2020.
- Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman,

- Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Muniyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *ArXiv*, 2021. URL <https://crfm.stanford.edu/assets/report.pdf>.
- Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. *Introduction to Statistical Learning Theory*, pages 169–207. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-28650-9. doi: 10.1007/978-3-540-28650-9\_8. URL [https://doi.org/10.1007/978-3-540-28650-9\\_8](https://doi.org/10.1007/978-3-540-28650-9_8).
- Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- Sacha Braun, Liviu Aolaritei, Michael I. Jordan, and Francis R. Bach. Minimum volume conformal sets for multivariate regression. *CoRR*, abs/2503.19068, March 2025. URL <https://doi.org/10.48550/arXiv.2503.19068>.
- Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In Shie Mannor, Nathan Srebro, and Robert C. Williamson, editors, *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23 of *Proceedings of Machine Learning Research*, pages 42.1–42.23, Edinburgh, Scotland, 25–27 Jun 2012. PMLR. URL <https://proceedings.mlr.press/v23/bubeck12b.html>.
- Giuseppe Carlo Calafiore and Laurent El Ghaoui. Ellipsoidal bounds for uncertain linear equations and dynamical systems. *Autom.*, 40(5):773–787, 2004. URL <https://doi.org/10.1016/j.automatica.2004.01.001>.
- José A. Agulló Candela. Exact iterative computation of the multivariate minimum volume ellipsoid estimator with a branch and bound algorithm. In Alfredo Prat, edi-

- tor, *COMPSTAT*, pages 175–180, Heidelberg, 1996. Physica-Verlag HD. doi: 10.1007/978-3-642-46992-3\_16.
- Xinyuan Cao, Jingbang Chen, Li Chen, Chris Lambert, Richard Peng, and Daniel Sleator. Learning-augmented b-trees. *CoRR*, abs/2211.09251, 2022. doi: 10.48550/ARXIV.2211.09251. URL <https://doi.org/10.48550/arXiv.2211.09251>.
- Miguel A Carreira-Perpinán and Christopher KI Williams. On the number of modes of a gaussian mixture. In *International Conference on Scale-Space Theories in Computer Vision*, pages 625–640. Springer, 2003.
- Parinya Chalermsook, Syamantak Das, Yunbum Kook, Bundit Laekhanukit, Yang P Liu, Richard Peng, Mark Sellke, and Daniel Vaz. Vertex sparsification for edge connectivity. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1206–1225. SIAM, 2021.
- Timothy M Chan. Three problems about dynamic convex hulls. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 27–36, 2011.
- Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.
- Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, page 47–60, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055491. URL <https://doi.org/10.1145/3055399.3055491>.
- Justin Chen, Sandeep Silwal, Ali Vakilian, and Fred Zhang. Faster fundamental graph algorithms via learned predictions. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 3583–3602. PMLR, 17–23 Jul 2022.
- Li Chen, Gramoz Goranci, Monika Henzinger, Richard Peng, and Thatchaphol Saranurak. Fast dynamic cuts, distances and effective resistances via vertex sparsifiers. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1135–1146. IEEE, 2020.
- Lijie Chen, Ran Duan, Ruosong Wang, Hanrui Zhang, and Tianyi Zhang. An improved algorithm for incremental DFS tree in undirected graphs. In David Eppstein, editor, *16th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2018, June 18–20, 2018, Malmö, Sweden*, volume 101 of *LIPIcs*, pages 16:1–16:12. Schloss Dagstuhl -

- Leibniz-Zentrum für Informatik, 2018. doi: 10.4230/LIPIcs.SWAT.2018.16. URL <https://doi.org/10.4230/LIPIcs.SWAT.2018.16>.
- Ming-Yen Cheng and Peter Hall. Calibrating the excess mass and dip tests of modality. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(3):579–589, 1998.
- Victor Chernozhukov, Kaspar Wüthrich, and Yinchu Zhu. Distributional conformal prediction. *Proceedings of the National Academy of Sciences*, 118(48):e2107794118, 2021.
- Michael B. Cohen, Ben Cousins, Yin Tat Lee, and Xin Yang. A near-optimal algorithm for approximating the john ellipsoid. In Alina Beygelzimer and Daniel Hsu, editors, *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99 of *Proceedings of Machine Learning Research*, pages 849–873. PMLR, 25–28 Jun 2019. URL <https://proceedings.mlr.press/v99/cohen19a.html>.
- Debarati Das, Maximilian Probst Gutenberg, and Christian Wulff-Nilsen. A near-optimal offline algorithm for dynamic all-pairs shortest paths in planar digraphs. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3482–3495. SIAM, 2022.
- Shai Ben David, Tyler Lu, Teresa Luu, and David Pal. Impossibility theorems for domain adaptation. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 129–136, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL <https://proceedings.mlr.press/v9/david10a.html>.
- Sami Davies, Benjamin Moseley, Sergei Vassilvitskii, and Yuyan Wang. Predictive flows for faster ford-fulkerson. In *Proceedings of the 40th International Conference on Machine Learning*, Jul 2023.
- Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2001.
- Guneet S Dhillon, George Deligiannidis, and Tom Rainforth. On the expected size of conformal prediction sets. In *International Conference on Artificial Intelligence and Statistics*, pages 1549–1557. PMLR, 2024.
- Persi Diaconis and David A. Freedman. Finite exchangeable sequences. *Annals of Probability*, 8:745–764, 1980. URL <https://api.semanticscholar.org/CorpusID:119580521>.
- Ilias Diakonikolas and Daniel M. Kane. *Algorithmic High-Dimensional Robust Statistics*. Cambridge University Press, 2023.

- Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664. IEEE, 2016.
- Ilias Diakonikolas, Daniel Kane, Daniel Kongsgaard, Jerry Li, and Kevin Tian. List-decodable mean estimation in nearly-pca time. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 10195–10208. Curran Associates, Inc., 2021. URL [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/547b85f3fafdf30856386753dc21c4e1-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/547b85f3fafdf30856386753dc21c4e1-Paper.pdf).
- Ilias Diakonikolas, Christos Tzamos, and Daniel M. Kane. A strongly polynomial algorithm for approximate forster transforms and its application to halfspace learning. *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, 2022. URL <https://api.semanticscholar.org/CorpusID:254275040>.
- Hu Ding. A Sub-Linear Time Framework for Geometric Optimization with Outliers in High Dimensions. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms (ESA 2020)*, volume 173 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:21, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-162-7. doi: 10.4230/LIPIcs.ESA.2020.38. URL <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ESA.2020.38>.
- Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Faster matchings via learned duals. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL <https://openreview.net/forum?id=kB8eks2Edt8>.
- Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Algorithms with prediction portfolios. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL [https://openreview.net/forum?id=wwWCZ7sER\\_C](https://openreview.net/forum?id=wwWCZ7sER_C).
- Irit Dinur, Shafi Goldwasser, and Huijia Lin. The computational benefit of correlated instances. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS ’15, page 219–228, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450333337. doi: 10.1145/2688073.2688082. URL <https://doi.org/10.1145/2688073.2688082>.
- David Dobkin and Subhash Suri. Maintenance of geometric extrema. *Journal of the ACM (JACM)*, 38(2):275–298, 1991.

- Paul Dütting, Federico Fusco, Silvio Lattanzi, Ashkan Norouzi-Fard, and Morteza Zadimoghaddam. Fully dynamic submodular maximization over matroids. *arXiv preprint arXiv:2305.19918*, 2023. To appear in ICML 2023.
- John HJ Einmahl and David M Mason. Generalized quantile processes. *The Annals of Statistics*, pages 1062–1078, 1992.
- David Eppstein. Offline algorithms for dynamic minimum spanning tree problems. *Journal of Algorithms*, 17(2):237 – 250, 1994. ISSN 0196-6774. doi: <https://doi.org/10.1006/jagm.1994.1033>. URL <http://www.sciencedirect.com/science/article/pii/S0196677484710339>.
- David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Amnon Nissenzweig. Sparsification—a technique for speeding up dynamic graph algorithms. *J. ACM*, 44(5):669–696, sep 1997. ISSN 0004-5411. doi: 10.1145/265910.265914. URL <https://doi.org/10.1145/265910.265914>.
- Jittat Fakcharoenphol and Satish Rao. Planar graphs, negative weight edges, shortest paths, and near linear time. *Journal of Computer and System Sciences*, 72(5):868–889, 2006.
- Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *Journal of Computer and System Sciences*, 69(3):485–497, 2004. ISSN 0022-0000. doi: <https://doi.org/10.1016/j.jcss.2004.04.011>. URL <https://www.sciencedirect.com/science/article/pii/S0022000004000637>. Special Issue on STOC 2003.
- Uriel Feige. Vertex cover is hardest to approximate on regular graphs. 11 2003.
- Moran Feldman, Paul Liu, Ashkan Norouzi-Fard, Ola Svensson, and Rico Zenklusen. Streaming submodular maximization under matroid constraints. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 59:1–59:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi: 10.4230/LIPIcs.ICALP.2022.59. URL <https://doi.org/10.4230/LIPIcs.ICALP.2022.59>.
- Shai Feldman, Liran Ringel, Stephen Bates, and Yaniv Romano. Achieving risk control in online learning settings. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL <https://openreview.net/forum?id=5Y04GWvoJu>.
- Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009. doi: 10.1137/070684914. URL <https://doi.org/10.1137/070684914>.



- Herbert Fleischner, Gert Sabidussi, and Vladimir I Sarvanov. Maximum independent sets in 3-and 4-regular hamiltonian graphs. *Discrete mathematics*, 310(20):2742–2749, 2010.
- Sebastian Forster, Gramoz Goranci, and Monika Henzinger. Dynamic maintenance of low-stretch probabilistic tree embeddings with applications. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1226–1245. SIAM, 2021.
- Sebastian Forster, Gramoz Goranci, Yasamin Nazari, and Antonis Skarlatos. Bootstrapping dynamic distance oracles. *Proceedings of the 2023 Annual European Symposium on Algorithms (ESA)*, 2023a.
- Sebastian Forster, Yasamin Nazari, and Maximilian Probst Gutenberg. Deterministic incremental APSP with polylogarithmic update time and stretch. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1173–1186. ACM, 2023b. doi: 10.1145/3564246.3585213. URL <https://doi.org/10.1145/3564246.3585213>.
- Dean P. Foster and Rakesh V. Vohra. Asymptotic calibration. *Biometrika*, 85(2):379–390, 1998. ISSN 00063444, 14643510. URL <http://www.jstor.org/stable/2337364>.
- Rina Foygel Barber, Emmanuel J Candes, Aaditya Ramdas, and Ryan J Tibshirani. The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, 10(2):455–482, 2021.
- Zvi Galil, Giuseppe F. Italiano, and Neil Sarnak. Fully dynamic planarity testing with applications. *J. ACM*, 46(1):28–91, jan 1999. ISSN 0004-5411. doi: 10.1145/300515.300517. URL <https://doi.org/10.1145/300515.300517>.
- Alex Gammerman, Volodya Vovk, and Vladimir Vapnik. Learning by transduction. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, UAI’98*, page 148–155, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc. ISBN 155860555X.
- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Computing high-dimensional confidence sets for arbitrary distributions. In Nika Haghtalab and Ankur Moitra, editors, *Proceedings of Thirty Eighth Conference on Learning Theory*, volume 291 of *Proceedings of Machine Learning Research*, pages 2227–2269. PMLR, 30 Jun–04 Jul 2025a. URL <https://proceedings.mlr.press/v291/gao25b.html>.
- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Volume optimality in conformal prediction with structured prediction sets. In *Forty-second International Conference on Machine Learning*, 2025b. URL <https://openreview.net/forum?id=oNDhnGrD51>.

- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Learning confidence ellipsoids and applications to robust subspace recovery. In *Proceedings of 67th Symposium on the Foundations of Computer Science*, 2026. URL <https://arxiv.org/abs/2512.16875>.
- Yu Gao, Yang Liu, and Richard Peng. Fully dynamic electrical flows: Sparse maxflow faster than goldberg–rao. *SIAM Journal on Computing*, (0):FOCS21–85, 2023.
- Javier Nuñez Garcia, Zoltan Kutalik, Kwang-Hyun Cho, and Olaf Wolkenhauer. Level sets and minimum volume sets of probability density functions. *International journal of approximate reasoning*, 34(1):25–47, 2003.
- Saurabh Garg, Sivaraman Balakrishnan, Zachary Chase Lipton, Behnam Neyshabur, and Hanie Sedghi. Leveraging unlabeled data to predict out-of-distribution performance. In *ICLR*, 2022. URL <https://arxiv.org/abs/2201.04234>.
- Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koepl, Preslav Nakov, and Iryna Gurevych. A survey of confidence estimation and calibration in large language models. In *NAACL-HLT*, pages 6577–6595, 2024. URL <https://doi.org/10.18653/v1/2024.naacl-long.366>.
- Subhashis Ghosal and Aad van der Vaart. Posterior convergence rates of dirichlet mixtures at smooth densities. *Annals of Statistics*, 35(2):697–723, 2007.
- Isaac Gibbs and Emmanuel Candes. Adaptive conformal inference under distribution shift. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 1660–1672. Curran Associates, Inc., 2021. URL [https://proceedings.neurips.cc/paper\\_files/paper/2021/file/0d441de75945e5acbc865406fc9a2559-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2021/file/0d441de75945e5acbc865406fc9a2559-Paper.pdf).
- Isaac Gibbs and Emmanuel Candès. Conformal inference for online prediction with arbitrary distribution shifts. *J. Mach. Learn. Res.*, 25(1), March 2025. ISSN 1532-4435.
- Andrew V Goldberg and Satish Rao. Beyond the flow decomposition barrier. *Journal of the ACM (JACM)*, 45(5):783–797, 1998.
- Shafi Goldwasser and Dhiraj Holden. The Complexity of Problems in P Given Correlated Instances. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:19, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-029-3. doi: 10.4230/LIPIcs.ITCS.2017.13. URL <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2017.13>.
- Shafi Goldwasser, Adam Tauman Kalai, Yael Tauman Kalai, and Omar



- Montasser. Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. In *NeurIPS 2020*. ACM, July 2020. URL <https://www.microsoft.com/en-us/research/publication/beyond-perturbations-learning-guarantees-with-arbitrary-adversarial-test-examples/>.
- Sreenivas Gollapudi and Debmalya Panigrahi. Online algorithms for rent-or-buy with expert advice. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2319–2327. PMLR, 09–15 Jun 2019. URL <https://proceedings.mlr.press/v97/gollapudi19a.html>.
- Noah Golowich, Allen Liu, and Abhishek Shetty. Provably learning from modern language models via low logit rank, 2025. URL <https://arxiv.org/abs/2512.09892>.
- Noah Golowich, Allen Liu, and Abhishek Shetty. Sequences of logits reveal the low rank structure of language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://openreview.net/forum?id=gdZ6J5hZzF>.
- Gramoz Goranci. *Dynamic Graph Algorithms and Graph Sparsification: New Techniques and Connections*. PhD thesis, University of Vienna, 2019.
- Gramoz Goranci, Monika Henzinger, and Thatchaphol Saranurak. Fast incremental algorithms via local sparsifiers. Manuscript, 2019.
- Gramoz Goranci, Harald Räcke, Thatchaphol Saranurak, and Zihan Tan. The expander hierarchy and its applications to dynamic graph algorithms. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2212–2228. SIAM, 2021.
- Albert Gu, Anupam Gupta, and Amit Kumar. The power of deferral: Maintaining a constant-competitive steiner tree online. *SIAM Journal on Computing*, 45(1):1–28, 2016. doi: 10.1137/140955276.
- Anupam Gupta and Kevin Lewi. The online metric matching problem for doubling metrics. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming*, pages 424–435, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- Chirag Gupta, Arun K Kuchibhotla, and Aaditya Ramdas. Nested conformal prediction and quantile out-of-bag ensemble methods. *Pattern Recognition*, 127:108496, 2022.
- Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL <https://www.gurobi.com>.
- Bernhard Haeupler, Telikepalli Kavitha, Rogers Mathew, Siddhartha Sen, and Robert E.

- Tarjan. Incremental cycle detection, topological ordering, and strong component maintenance. *ACM Trans. Algorithms*, 8(1), jan 2012. ISSN 1549-6325. doi: 10.1145/2071379.2071382. URL <https://doi.org/10.1145/2071379.2071382>.
- Moritz Hardt and Ankur Moitra. Algorithms and hardness for robust subspace recovery. In Shai Shalev-Shwartz and Ingo Steinwart, editors, *Proceedings of the 26th Annual Conference on Learning Theory*, volume 30 of *Proceedings of Machine Learning Research*, pages 354–375, Princeton, NJ, USA, 12–14 Jun 2013. PMLR. URL <https://proceedings.mlr.press/v30/Hardt13.html>.
- John A Hartigan. Estimation of a convex density contour in two dimensions. *Journal of the American Statistical Association*, 82(397):267–270, 1987.
- Jason Hartline, Lunjia Hu, and Yifan Wu. A perfectly truthful calibration measure. In Steve Hanneke and Tor Lattimore, editors, *Proceedings of Thirty Ninth Conference on Learning Theory*, volume 336 of *Proceedings of Machine Learning Research*, pages 3185–3223. PMLR, 29 Jun–03 Jul 2026. URL <https://proceedings.mlr.press/v336/hartline26a.html>.
- Iain Henderson, Adrien Mazoyer, and Fabrice Gamboa. Adaptive inference with random ellipsoids through conformal conditional linear expectation, 2025. URL <https://arxiv.org/abs/2409.18508>.
- Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 21–30, 2015.
- Monika Henzinger, Sebastian Krinninger, and Danupon Nanongkai. Dynamic approximate all-pairs shortest paths: Breaking the  $o(mn)$  barrier and derandomization. *SIAM Journal on Computing*, 45(3):947–1006, 2016.
- Monika Henzinger, Barna Saha, Martin P. Seybold, and Christopher Ye. On the complexity of algorithms with predictions for dynamic graph problems, 2023.
- Monika R Henzinger and Valerie King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM (JACM)*, 46(4):502–516, 1999.
- Monika Rauch Henzinger and Valerie King. Maintaining minimum spanning trees in dynamic graphs. In Pierpaolo Degano, Roberto Gorrieri, and Alberto Marchetti-Spaccamela, editors, *Automata, Languages and Programming, 24th International Colloquium, ICALP’97, Bologna, Italy, 7-11 July 1997, Proceedings*, volume 1256 of *Lecture Notes in Computer Science*, pages 594–604. Springer, 1997. doi: 10.1007/3-540-63165-8\_214. URL [https://doi.org/10.1007/3-540-63165-8\\_214](https://doi.org/10.1007/3-540-63165-8_214).

- Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi: 10.1080/01621459.1963.10500830. URL <https://www.tandfonline.com/doi/abs/10.1080/01621459.1963.10500830>.
- Jacob Holm and Eva Rotenberg. Worst-case polylog incremental spqr-trees: Embeddings, planarity, and triconnectivity. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2378–2397. SIAM, 2020.
- Jacob Holm, Kristian De Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM (JACM)*, 48(4):723–760, 2001.
- Max Hopkins, Daniel M. Kane, Shachar Lovett, and Gaurav Mahajan. Point location and active learning: Learning halfspaces almost optimally, 2020. URL <https://arxiv.org/abs/2004.11380>.
- Lunjia Hu and Yifan Wu. Predict to minimize swap regret for all payoff-bounded tasks. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 244–263, 2024. doi: 10.1109/FOCS61266.2024.00024.
- Peter J Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.
- Peter J. Huber. *Robust Statistics*. Wiley Series in Probability and Statistics - Applied Probability and Statistics Section Series. Wiley, 2004. ISBN 9780471650720. URL <https://books.google.com/books?id=e62RhdqIdMkC>.
- Rob J Hyndman. Computing and graphing highest density regions. *The American Statistician*, 50(2):120–126, 1996.
- Makoto Imase and Bernard M. Waxman. Dynamic steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991. doi: 10.1137/0404033.
- Rafael Izbicki and Ann B Lee. Converting high-dimensional regression to high-dimensional conditional density estimation. *Electronic Journal of Statistics*, 11:2800–2831, 2017.
- Rafael Izbicki, Gilson Shimizu, and Rafael Stern. Flexible distribution-free conditional predictive bands using density estimators. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3068–3077. PMLR, 26–28 Aug 2020a. URL <https://proceedings.mlr.press/v108/izbicki20a.html>.

- Rafael Izbicki, Gilson T. Shimizu, and Rafael Bassi Stern. Cd-split and hpd-split: Efficient conformal regions in high dimensions. *J. Mach. Learn. Res.*, 23:87:1–87:32, 2020b. URL <https://api.semanticscholar.org/CorpusID:238354408>.
- Wenyu Jin and Xiaorui Sun. Fully dynamic st edge connectivity in subpolynomial time. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 861–872. IEEE, 2022.
- Fritz John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays : Courant Anniversary Volume*, pages 187–204. John Wiley & Sons : Interscience Division, New York, 1948.
- Ming-Yang Kao, John H. Reif, and Stephen R. Tate. Searching in an unknown environment: an optimal randomized algorithm for the cow-path problem. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, page 441–447, USA, 1993. Society for Industrial and Applied Mathematics. ISBN 0898713137.
- Leonid Khachiyan. On the complexity of approximating extremal determinants in matrices. *J. Complex.*, 11(1):138–153, March 1995. ISSN 0885-064X. doi: 10.1006/jcom.1995.1005. URL <https://doi.org/10.1006/jcom.1995.1005>.
- Mikhail Khodak, Nina Balcan, Ameet Talwalkar, and Sergei Vassilvitskii. Learning predictions for algorithms with predictions. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=50LcPQaYTVg>.
- Mikhail Khodak, Edmond Chow, Maria Florina Balcan, and Ameet Talwalkar. Learning to relax: Setting solver parameters across a sequence of linear system instances. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=5t57omGVMw>.
- Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 767–775, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581134959. doi: 10.1145/509907.510017. URL <https://doi.org/10.1145/509907.510017>.
- Valerie King. Fully dynamic algorithms for maintaining all-pairs shortest paths and transitive closure in digraphs. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 81–89. IEEE, 1999.
- Shayan Kiyani, George Pappas, and Hamed Hassani. Length optimization in conformal prediction, 2024. URL <https://arxiv.org/abs/2406.18814>.
- Shayan Kiyani, George J. Pappas, Aaron Roth, and Hamed Hassani. Decision theoretic

- foundations for conformal prediction: Optimal uncertainty quantification for risk-averse agents. In *Forty-second International Conference on Machine Learning*, 2025. URL <https://openreview.net/forum?id=Ukjl86EsIk>.
- Philip N Klein. Multiple-source shortest paths in planar graphs. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 146–155, 2005.
- Philip N Klein and Sairam Subramanian. A fully dynamic approximation scheme for shortest paths in planar graphs. *Algorithmica*, 22(3):235–249, 1998.
- Adam Klivans, Konstantinos Stavropoulos, and Arsen Vasilyan. Testable learning with distribution shift. In Shipra Agrawal and Aaron Roth, editors, *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pages 2887–2943. PMLR, 30 Jun–03 Jul 2024. URL <https://proceedings.mlr.press/v247/klivans24a.html>.
- Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1272–1287. SIAM, 2016.
- Elias Koutsoupias. The k-server problem. *Computer Science Review*, 3(2):105–118, 2009. ISSN 1574-0137. doi: <https://doi.org/10.1016/j.cosrev.2009.04.002>. URL <https://www.sciencedirect.com/science/article/pii/S1574013709000197>.
- Elias Koutsoupias and Christos H. Papadimitriou. On the k-server conjecture. *J. ACM*, 42(5):971–983, sep 1995. ISSN 0004-5411. doi: [10.1145/210118.210128](https://doi.org/10.1145/210118.210128). URL <https://doi.org/10.1145/210118.210128>.
- Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD '18*, page 489–504, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450347037. doi: [10.1145/3183713.3196909](https://doi.org/10.1145/3183713.3196909). URL <https://doi.org/10.1145/3183713.3196909>.
- Willem Kruijer, Judith Rousseau, and Aad van der Vaart. Adaptive bayesian density estimation with location-scale mixtures. *Electronic Journal of Statistics*, 4:1225–1257, 2010.
- Ravi Kumar, Manish Purohit, and Zoya Svitkina. Improving online algorithms via ml predictions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL [https://proceedings.neurips.cc/paper\\_files/paper/2018/file/73a427badebe0e32caa2e1fc7530b7f3-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2018/file/73a427badebe0e32caa2e1fc7530b7f3-Paper.pdf).
- Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory

- in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.
- Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 665–674. IEEE, 2016.
- Lucien LeCam and Lorraine Schwartz. A necessary and sufficient condition for the existence of consistent estimates. *The Annals of Mathematical Statistics*, 31(1):140–150, 1960.
- Jing Lei. Classification with confidence. *Biometrika*, 101(4):755–769, 2014.
- Jing Lei and Larry Wasserman. Distribution-free prediction bands for non-parametric regression. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 76(1):71–96, 2014.
- Jing Lei, James M. Robins, and Larry A. Wasserman. Distribution-free prediction sets. *Journal of the American Statistical Association*, 108:278 – 287, 2013. URL <https://api.semanticscholar.org/CorpusID:17499892>.
- Jing Lei, Alessandro Rinaldo, and Larry Wasserman. A conformal prediction approach to explore functional data. *Annals of Mathematics and Artificial Intelligence*, 74:29–43, 2015.
- Gilad Lerman and Tyler Maunu. An overview of robust subspace recovery. *Proceedings of the IEEE*, 106(8):1380–1410, 2018. doi: 10.1109/JPROC.2018.2853141.
- Elliott H. Lieb. Gaussian kernels have only gaussian maximizers. *Inventiones mathematicae*, 102:179–208, 1990. URL <https://api.semanticscholar.org/CorpusID:189830869>.
- Honghao Lin, Tian Luo, and David Woodruff. Learning augmented binary search trees. In *International Conference on Machine Learning*, pages 13431–13440. PMLR, 2022.
- Alexander Lindermayr and Nicole Megow. Algorithms with predictions (open source running bibliography), 2021. URL <https://algorithms-with-predictions.github.io/>.
- Allen Liu and Ankur Moitra. Model stealing for any low-rank language model. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC ’25, page 1755–1761, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715105. doi: 10.1145/3717823.3718220. URL <https://doi.org/10.1145/3717823.3718220>.
- Qiyu Liu, Siyuan Han, Jianwei Liao, Jin Li, Jingshu Peng, Jun Du, and Lei Chen. Learned data compression: Challenges and opportunities for the future, 2024. URL <https://arxiv.org/abs/2412.10770>.



- Quanquan C. Liu and Vaidehi Srinivas. The predicted-updates dynamic model: Offline, incremental, and decremental to fully dynamic transformations. In Shipra Agrawal and Aaron Roth, editors, *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pages 3582–3641. PMLR, 30 Jun–03 Jul 2024. URL <https://proceedings.mlr.press/v247/liu24c.html>.
- Yunfan Lu, Yiqi Lin, Hao Wu, Yunhao Luo, Xu Zheng, Hui Xiong, and Lin Wang. Priors in deep image restoration and enhancement: A survey, 2023. URL <https://arxiv.org/abs/2206.02070>.
- Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *J. ACM*, 68(4), jul 2021. ISSN 0004-5411. doi: 10.1145/3447579. URL <https://doi.org/10.1145/3447579>.
- Mark Manasse, Lyle McGeoch, and Daniel Sleator. Competitive algorithms for on-line problems. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, page 322–333, New York, NY, USA, 1988. Association for Computing Machinery. ISBN 0897912640. doi: 10.1145/62212.62243. URL <https://doi.org/10.1145/62212.62243>.
- Mark S Manasse, Lyle A McGeoch, and Daniel D Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11(2):208–230, 1990. ISSN 0196-6774. doi: [https://doi.org/10.1016/0196-6774\(90\)90003-W](https://doi.org/10.1016/0196-6774(90)90003-W). URL <https://www.sciencedirect.com/science/article/pii/019667749090003W>.
- Samuel McCauley, Benjamin Moseley, Aidin Niaparast, Helia Niaparast, and Shikha Singh. Incremental Approximate Single-Source Shortest Paths with Predictions. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, volume 334 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 117:1–117:20, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-372-0. doi: 10.4230/LIPIcs.ICALP.2025.117. URL <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2025.117>.
- Andrew McGregor, Krzysztof Onak, and Rina Panigrahy. The oil searching problem. pages 504–515, 09 2009. ISBN 978-3-642-04127-3. doi: 10.1007/978-3-642-04128-0\_45.
- Nicolai Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(35):983–999, 2006. URL <http://jmlr.org/papers/v7/meinshausen06a.html>.
- Soundouss Messoudi, Sébastien Destercke, and Sylvain Rousseau. Ellipsoidal conformal inference for multi-target regression. In Ulf Johansson, Henrik Boström, Khuong An Nguyen, Zhiyuan Luo, and Lars Carlsson, editors, *Proceedings of the Eleventh Sym-*

- posium on Conformal and Probabilistic Prediction with Applications*, volume 179 of *Proceedings of Machine Learning Research*, pages 294–306. PMLR, 24–26 Aug 2022. URL <https://proceedings.mlr.press/v179/messoudi22a.html>.
- Michael Mitzenmacher and Sergei Vassilvitskii. *Algorithms with Predictions*, page 646–662. Cambridge University Press, 2021. doi: 10.1017/9781108637435.037.
- Dietrich W Müller and Günther Sawitzki. Excess mass estimates and tests for multimodality. *Journal of the American Statistical Association*, 86(415):738–746, 1991.
- Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunye Koh, and Sungchul Kim. Continuous-time dynamic network embeddings. In *Companion proceedings of the the web conference 2018*, pages 969–976, 2018.
- Aleksandar Nikolov. Randomized rounding for the largest simplex problem. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, STOC ’15, page 861–870, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450335362. doi: 10.1145/2746539.2746628. URL <https://doi.org/10.1145/2746539.2746628>.
- Aleksandar Nikolov and Mohit Singh. Maximizing determinants under partition constraints. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC ’16, page 192–201, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341325. doi: 10.1145/2897518.2897649. URL <https://doi.org/10.1145/2897518.2897649>.
- Deborah Nolan. The excess-mass ellipsoid. *Journal of multivariate analysis*, 39(2):348–371, 1991.
- Liam O’Carroll, Vaidehi Srinivas, and Aravindan Vijayaraghavan. The burer-monteiro SDP method can fail even above the barvinok-pataki bound. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=C2Mikd2Wp0c>.
- Mark H Overmars and Jan van Leeuwen. Two general methods for dynamizing decomposable searching problems. 1979.
- Mark H Overmars and Jan Van Leeuwen. Dynamization of decomposable searching problems yielding good worst-case bounds. In *Theoretical Computer Science: 5th GI-Conference Karlsruhe, March 23–25, 1981*, pages 224–233. Springer, 2005.
- Chiwoo Park, Jianhua Z Huang, and Yu Ding. A computable plug-in estimator of minimum volume sets for novelty detection. *Operations Research*, 58(5):1469–1480, 2010.



- Binghui Peng and Aviad Rubinfeld. Fully-dynamic-to-incremental reductions with known deletion order (e.g. sliding window). In *2023 Symposium on Simplicity in Algorithms (SOSA)*, pages 261–271, 2023. doi: 10.1137/1.9781611977585.ch24. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611977585.ch24>.
- Richard Peng, Bryce Sandlund, and Daniel Dominic Sleator. Optimal offline dynamic 2, 3-edge/vertex connectivity. In *Workshop on Algorithms and Data Structures*, 2017.
- Wolfgang Polonik. Measuring mass concentrations and estimating density contour clusters—an excess mass approach. *The annals of Statistics*, pages 855–881, 1995.
- Wolfgang Polonik. Minimum volume sets and generalized quantile processes. *Stochastic processes and their applications*, 69(1):1–24, 1997.
- Wolfgang Polonik. Concentration and goodness-of-fit in higher dimensions:(asymptotically) distribution-free methods. *The Annals of Statistics*, 27(4):1210–1229, 1999.
- Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. *Advances in Neural Information Processing Systems*, 35:32928–32941, 2022.
- Drew Prinster, Samuel Stanton, Anqi Liu, and Suchi Saria. Conformal validity guarantees exist for any data distribution (and how to find them). In *Proceedings of the 41st International Conference on Machine Learning, ICML’24*. JMLR.org, 2024.
- Maximilian Probst Gutenberg, Virginia Vassilevska Williams, and Nicole Wein. New algorithms and hardness for incremental single-source shortest paths in directed graphs. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 153–166, 2020.
- Kha Gia Quach, Pha Nguyen, Huu Le, Thanh-Dat Truong, Chi Nhan Duong, Minh-Triet Tran, and Khoa Luu. Dyglip: A dynamic graph model with link prediction for accurate multi-camera multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13784–13793, 2021.
- Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing, STOC ’10*, page 755–764, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450300506. doi: 10.1145/1806689.1806792. URL <https://doi.org/10.1145/1806689.1806792>.
- Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *2012 IEEE 27th Conference on Computational Complexity*, pages 64–73, 2012. doi: 10.1109/CCC.2012.43.

- Philippe Rigollet and Régis Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, pages 1154–1178, 2009.
- Liam Roditty. Decremental maintenance of strongly connected components. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, page 1143–1150, USA, 2013. Society for Industrial and Applied Mathematics. ISBN 9781611972511.
- Liam Roditty and Uri Zwick. On dynamic shortest paths problems. In *Algorithms–ESA 2004: 12th Annual European Symposium, Bergen, Norway, September 14–17, 2004. Proceedings 12*, pages 580–591. Springer, 2004.
- Jasper Roebroek. sklearn-quantile, 2023. URL <https://sklearn-quantile.readthedocs.io/en/latest/index.html>.
- Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.
- Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.
- Tim Roughgarden, editor. *Beyond the Worst-Case Analysis of Algorithms*. Cambridge University Press, 2021. doi: 10.1017/9781108637435.
- Peter Rousseeuw. Multivariate estimation with high breakdown point. *Mathematical Statistics and Applications B*, 1985.
- Peter J Rousseeuw. Least median of squares regression. *Journal of the American statistical association*, 79(388):871–880, 1984.
- Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR, 2018.
- Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.
- Mauricio Sadinle, Jing Lei, and Larry A. Wasserman. Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, 114:223 – 234, 2016. URL <https://api.semanticscholar.org/CorpusID:622583>.
- Mauricio Sadinle, Jing Lei, and Larry Wasserman. Least ambiguous set-valued classifiers

- with bounded error levels. *Journal of the American Statistical Association*, 114(525): 223–234, 2019. doi: 10.1080/01621459.2017.1395341.
- Thomas W Sager. Estimation of a multivariate mode. *The Annals of Statistics*, 6(4):802–812, 1978.
- Thomas W Sager. An iterative method for estimating a multivariate mode and isopleth. *Journal of the American Statistical Association*, 74(366a):329–339, 1979.
- Kevin Schaul and Nitasha Tiku. The white house said anthropic’s powerful ai was ‘jailbroken.’ here’s what that means. *The Washington Post*, 2026. URL <https://www.washingtonpost.com/technology/2026/06/18/surprisingly-simple-ways-ai-can-be-tricked-into-breaking-its-own-rules/>.
- Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.
- Clayton Scott and Robert Nowak. Learning minimum volume sets. *Advances in neural information processing systems*, 18, 2005.
- Robert Serfling. Probability inequalities for the sum in sampling without replacement. *Annals of Statistics*, 2:39–48, 1974. URL <https://api.semanticscholar.org/CorpusID:120916609>.
- Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9(12):371–421, 2008. URL <http://jmlr.org/papers/v9/shafer08a.html>.
- Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.
- Dravyansh Sharma, Maria-Florina Balcan, and Travis Dick. Learning piecewise lipschitz functions in changing environments. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3567–3577. PMLR, 26–28 Aug 2020. URL <https://proceedings.mlr.press/v108/sharma20a.html>.
- Vladimir Shenmaier. Complexity and approximation of the smallest k-enclosing ball problem. *European Journal of Combinatorics*, 48:81–87, 2015. ISSN 0195-6698. doi: <https://doi.org/10.1016/j.ejc.2015.02.011>. URL <https://www.sciencedirect.com/science/article/pii/S0195669815000335>. Selected Papers of EuroComb’13.
- Vaidehi Srinivas. Online conformal prediction with efficiency guarantees. In *Proceedings of*

- the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2026. URL <https://arxiv.org/abs/2507.02496>.
- Vaidehi Srinivas, David P. Woodruff, Ziyu Xu, and Samson Zhou. Memory bounds for the experts problem. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, page 1158–1171, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392648. doi: 10.1145/3519935.3520069. URL <https://doi.org/10.1145/3519935.3520069>.
- Jacob Steinhardt. Lecture notes for stat260 (robust statistics). 2019.
- Ingo Steinwart, Don Hush, and Clint Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6(2), 2005.
- Peng Sun and Robert M. Freund. Computation of minimum-volume covering ellipsoids. *Oper. Res.*, 52(5):690–706, October 2004. ISSN 0030-364X. doi: 10.1287/opre.1040.0115. URL <https://doi.org/10.1287/opre.1040.0115>.
- David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004.
- Ryan J. Tibshirani, Rina Foygel Barber, Emmanuel J. Candès, and Aaditya Ramdas. *Conformal prediction under covariate shift*. Curran Associates Inc., Red Hook, NY, USA, 2019.
- Michael J. Todd. *Minimum-Volume Ellipsoids: Theory and Algorithms*. SIAM-Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2016. ISBN 1611974372.
- Michael J. Todd and E. Alper Yıldırım. On khachiyan’s algorithm for the computation of minimum-volume enclosing ellipsoids. *Discrete Appl. Math.*, 155(13):1731–1744, August 2007. ISSN 0166-218X. doi: 10.1016/j.dam.2007.02.013. URL <https://doi.org/10.1016/j.dam.2007.02.013>.
- Alexandre B Tsybakov. On nonparametric estimation of density level sets. *The Annals of Statistics*, 25(3):948–969, 1997.
- Stefan Van Aelst and Peter Rousseeuw. Minimum volume ellipsoid. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(1):71–82, 2009.
- Jan van den Brand, Sebastian Forster, Yasamin Nazari, and Adam Polak. On dynamic graph algorithms with predictions. *CoRR*, abs/2307.09961, 2023. doi: 10.48550/ARXIV.2307.09961. URL <https://doi.org/10.48550/arXiv.2307.09961>.
- Eleftherios E. Vlahakis, Lars Lindemann, Pantelis Sopasakis, and Dimos V. Dimarogonas.

- Conformal prediction for distribution-free optimal control of linear stochastic systems, 2024. URL <https://arxiv.org/abs/2411.19132>.
- Vladimir Vovk. On-line confidence machines are well-calibrated. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 187–196, 2002. doi: 10.1109/SFCS.2002.1181895.
- Vladimir Vovk. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning*, pages 475–490. PMLR, 2012.
- Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0387001522.
- Vladimir Vovk, Valentina Fedorova, Ilia Nouretdinov, and Alexander Gammerman. Criteria of efficiency for conformal prediction. In *Conformal and Probabilistic Prediction with Applications: 5th International Symposium, COPA 2016, Madrid, Spain, April 20-22, 2016, Proceedings 5*, pages 23–39. Springer, 2016.
- Cheng Wang. Calibration in deep learning: A survey of the state-of-the-art, 2025. URL <https://arxiv.org/abs/2308.01222>.
- Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. *arXiv preprint arXiv:2101.05974*, 2021.
- Zhendong Wang, Ruijiang Gao, Mingzhang Yin, Mingyuan Zhou, and David Blei. Probabilistic conformal prediction using conditional random samples. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 8814–8836. PMLR, 25–27 Apr 2023. URL <https://proceedings.mlr.press/v206/wang23n.html>.
- Alexander Wei and Fred Zhang. Optimal robustness-consistency trade-offs for learning-augmented online algorithms. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS’20*, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.
- Chen Xu, Hanyang Jiang, and Yao Xie. Conformal prediction for multi-dimensional time series by ellipsoidal sets. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 55076–55099. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/xu24m.html>.

- Chenyang Xu and Benjamin Moseley. Learning-augmented algorithms for online steiner tree. In *AAAI Conference on Artificial Intelligence*, 2021. URL <https://api.semanticscholar.org/CorpusID:245117802>.
- Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. Jailbreak attacks and defenses against large language models: A survey, 2024. URL <https://arxiv.org/abs/2407.04295>.
- Margaux Zaffran, Olivier Feron, Yannig Goude, Julie Josse, and Aymeric Dieuleveut. Adaptive conformal predictions for time series. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 25834–25866. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/zaffran22a.html>.
- Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.
- Yiqi Zhao, Xinyi Yu, Matteo Sesia, Jyotirmoy V. Deshmukh, and Lars Lindemann. Conformal predictive programming for chance constrained optimization, 2026. URL <https://arxiv.org/abs/2402.07407>.

# Appendix A

## Details for Dynamic Algorithms with Predictions

### A.1 Preprocessing Proofs

In this section, we provide the proofs for preprocessing that we omitted in the main body of [Chapter 9](#). We first recall [Lemma 9.3.3](#).

**Lemma 9.3.3** (Initial Scheduling Quality and Runtime). *Let  $\mathbf{p}$  be the vector of predictions, mapping each event to a day in  $[T]$ . Let  $\mathbf{r}$  be the (unknown to the algorithm) true vector of real events, mapping each event to the day in  $[T]$  that it actually occurs.*

*Given  $\mathbf{p}$ , we can compute an assignment vector  $\mathbf{a}_0$  such that  $\mathbf{a}_0$  assigns at most one insertion event and at most one deletion event to each day, assigns all insertion events before deletion events, and has error*

$$\mathbf{E} [\|\mathbf{a}_0 - \mathbf{r}\|_1] = O(\|\mathbf{p} - \mathbf{r}\|_1 \cdot \log(T)).$$

Furthermore, we can compute this in time  $O(T \log^*(T))$ .

*Proof.* We iterate through the events in  $P$  in arbitrary order; the reason is that the online metric matching algorithm can handle requests that occur in any order. At each predicted event, we match it to an available day assignment, where only one event can be scheduled on each day. Then, this problem becomes an instance of online metric matching, where the predicted events are requests, and there is one available server at each day. In particular, our problem instances are over the line metric.

We implement the harmonic algorithm for online metric matching on line metrics, due to [Gupta and Lewi \[2012\]](#). In this algorithm, a request is matched to either its closest open server on the left, or the closest open server on the right. Let  $d_{\text{left}}$  be the distance from the request to the closest open server to the left, and  $d_{\text{right}}$  be the distance from the request to the closest open server on the right. Then the request is assigned to the left server with probability  $\frac{1/d_{\text{left}}}{1/d_{\text{left}} + 1/d_{\text{right}}}$ , and to the right server otherwise.

We note that this algorithm can be implemented in amortized  $O(\log^*(T))$  time. We maintain a union-find data structure that keeps track of contiguous blocks of assigned servers where initially each  $t \in [T]$  is assigned to a different disjoint set. For each block, we maintain metadata with the first open server to the left and the first open server to the right. Then, when a new request arrives, finding the closest open servers can be done by looking up the request's representative element in the union-find data structure. One request is assigned per arrival, so at most three sets must be merged for each arrival. Since union-find can be implemented with amortized  $O(\log^*(T))$  update time, and we use a constant number of calls to union-find (with constant additional work), we can implement this algorithm with amortized  $O(\log^*(T))$  update time.



Now we argue that it meets our correctness guarantees. The harmonic algorithm is  $O(\log(T))$  competitive, for  $T$  requests. Let  $\mathbf{r}$  be the vector of the real update days of each event,  $\mathbf{p}$  be the predicted update times of each event, and  $\mathbf{a}_0$  be the assigned update days of each event. We have

$$\begin{aligned} \|\mathbf{a}_0 - \mathbf{r}\|_1 &\leq \|\mathbf{a}_0 - \mathbf{p}\|_1 + \|\mathbf{p} - \mathbf{r}\|_1 \\ &\leq O(\log(T)) \cdot \|\mathbf{c} - \mathbf{p}\|_1 + \|\mathbf{p} - \mathbf{r}\|_1, \end{aligned}$$

where  $\mathbf{c}$  is the feasible assignment with the lowest  $\ell_1$  distance to  $\mathbf{p}$ , in hindsight. Since we know that  $\mathbf{r}$  is also a feasible assignment, we have that

$$\|\mathbf{c} - \mathbf{p}\|_1 \leq \|\mathbf{p} - \mathbf{r}\|_1$$

Thus,

$$\|\mathbf{a}_0 - \mathbf{r}\|_1 \leq O(\log(T)) \cdot \|\mathbf{p} - \mathbf{r}\|_1.$$

Finally, to resolve the ordering constraints among events on the same element  $e$ , we iterate through all of our assigned days and maintain a set of all elements for which we have seen a deletion event but not an insertion event. Suppose that for element  $e$  we see an insertion event on day  $t_{e,i}$  *after* its deletion event on day  $t_{e,d}$ , we reassign the day for the deletion event to  $t_{e,i}$ . If there are multiple deletion events that occur before an insertion event, we reassign each deletion event to the next available insertion event and to  $T + 1$  otherwise. Since we are guaranteed at most one event per day before our reassignment of events and our assignment procedure ensures that we reassign at most one deletion event for each element  $e$  to its corresponding insertion event, the maximum number of events assigned to each day

$t \in [T]$  is two and there is at most one insertion event and one deletion event assigned to  $t$ . We produce a new vector  $\mathbf{a}'_0$  after this final processing.

We now prove that our new  $\mathbf{a}'_0$  also satisfies our error bounds. We show the following inequality since all insertion events are assigned the same days as  $\mathbf{a}_0$  and only deletions are potentially reassigned:

$$\|\mathbf{a}'_0 - \mathbf{r}\|_1 \leq \|\mathbf{a}_0 - \mathbf{r}\|_1 + \|\mathbf{a}'_{0,\text{del}} - \mathbf{r}_{\text{del}}\|_1,$$

where  $\mathbf{a}'_{0,\text{del}}$  and  $\mathbf{r}_{\text{del}}$  are the assigned and real days, respectively, of the deletion events. We now show that  $\|\mathbf{a}'_{0,\text{del}} - \mathbf{r}_{\text{del}}\|_1 \leq \|\mathbf{a}_0 - \mathbf{r}\|_1$  through several cases. Let  $\hat{t}_{i,e,1} < \hat{t}_{i,e,2} < \dots$  be the insertion times for element  $e$  in  $\mathbf{a}_0$ ,  $\hat{t}_{d,e,1} < \hat{t}_{d,e,2} < \dots$  be the deletion times for element  $e$  in  $\mathbf{a}_0$ ,  $t_{i,e,1} < t_{i,e,2} < \dots$  be the insertion times for element  $e$  in  $\mathbf{r}$ , and  $t_{d,e,1} < t_{d,e,2} < \dots$  be the deletion times for element  $e$  in  $\mathbf{r}$ . We assign  $t_{i,e,j} = T + 1$  for any updates where we have a greater number of predicted events than real events. Then, we show, for any  $j$ :

1. If  $\hat{\mathbf{t}}_{i,e,j} < \mathbf{t}_{i,e,j} < \mathbf{t}_{d,e,j}$  or  $\mathbf{t}_{i,e,j} < \hat{\mathbf{t}}_{i,e,j} < \mathbf{t}_{d,e,j}$ , then  $|t_{d,e,j} - \hat{t}_{i,e,j}| < |\hat{t}_{d,e,j} - t_{d,e,j}|$ .
2. Otherwise, if  $\mathbf{t}_{i,e,j} < \mathbf{t}_{d,e,j} < \hat{\mathbf{t}}_{i,e,j}$ , then  $|t_{d,e,j} - \hat{t}_{i,e,j}| < 2 \cdot |\hat{t}_{i,e,j} - t_{i,e,j}|$ .

Hence, via the casework above, we show that

$$\|\mathbf{a}'_0 - \mathbf{r}\|_1 \leq 2 \cdot \|\mathbf{a}_{0,\text{ins}} - \mathbf{r}_{\text{ins}}\|_1 + \|\mathbf{a}_{0,\text{del}} - \mathbf{r}_{\text{del}}\|_1 \leq 2 \cdot \|\mathbf{a}_0 - \mathbf{r}\|_1,$$

where  $\mathbf{a}_{0,\text{ins}}$  and  $\mathbf{r}_{\text{ins}}$  are the assigned and real days, respectively, of the insertion events. This proof can be extended beyond insertions and deletions to any constant number of ordering constraints of events on elements. Such a reassignment to produce  $\mathbf{a}'_0$  requires  $O(T)$  additional work.  $\square$

**Lemma 9.3.5** (Partition-Tree has  $O(\log(T))$  Depth). *The depth of a random partition-tree drawn over  $T$  days is  $O(\log(T))$  in expectation. Furthermore, for any constant  $k \geq 36$ , the depth is*

$$\leq k \ln(T) \quad \text{with probability} \quad \geq 1 - T^{-\frac{k}{24}}.$$

*Proof.* Consider an arbitrary day  $t \in [T]$  which is a leaf in the partition-tree as defined in [Definition 9.3.1](#). We analyze the depth of the leaf containing  $t$  in the partition-tree for every  $t \in [T]$ . Consider the path from the root of the partition-tree to this leaf, that is the path of all windows containing  $t$ .

Consider some non-leaf window  $W$  of size  $S$  that contains  $t$ ; the size of window  $W$  is a random variable with value  $S$  conditioned on a set of events. We know that with probability at least  $\frac{1}{2}$ , the random partition of  $W$  results in both children of  $W$  having size at most  $\frac{3S}{4}$ . We call this the “good event.” (Consider the center  $\lfloor \frac{S}{2} \rfloor$  elements of  $W$ . There are strictly more dividers that border one of these center elements than there are dividers that do not border these elements. If any of these dividers is chosen, the smaller child has size at least  $\frac{S}{4}$ , thus both children have size at most  $\frac{3S}{4}$ .)

Good events can occur at most  $\log_{\frac{4}{3}}(T) = \frac{\ln(T)}{\ln \frac{4}{3}} < 4 \ln(T)$  times over the windows on the root-to-leaf path to  $t$  before there is one element left and the path ends.

Now we bound the probability that  $t$ 's path in the partition-tree has length more than  $k \ln(T)$ , for some constant  $k \geq 36$ . For this to occur, it is necessary that in the partitioning for the first  $k \ln(T)$  windows on  $t$ 's path, the good event occurred fewer than  $4 \ln(T)$  times.

We can bound the probability that good events occur fewer than  $4 \ln(T)$  times with the Chernoff-Hoeffding inequality. Furthermore, for this analysis, we assume that the number

of windows along  $t$ 's path is at least  $k \ln(T)$ ; otherwise, we have our desired property. Let

$$X = X_1 + \cdots + X_{k \ln(T)},$$

where  $X_j$  is the indicator random variable that the good event occurred at the  $j$ th window of  $i$ 's path. We have  $\mathbf{E}[X_j] \geq \frac{1}{2}$  and  $\mathbf{E}[X] \geq \frac{k \ln(T)}{2}$ . Furthermore, each good event is independent of previous good events. Thus,

$$\begin{aligned} \Pr[X \leq \mathbf{E}[X] - t] &\leq e^{-\frac{t^2}{3\mathbf{E}[X]}} \\ \Pr[X \leq 4 \ln(T)] &= \Pr\left[X \leq \mathbf{E}[X] - \left(\frac{k}{2} - 4\right) \ln(T)\right] \\ &\leq e^{-\frac{((\frac{k}{2}-4) \ln(T))^2}{\frac{3k \ln(T)}{2}}} \\ &= e^{-(\frac{k}{6} - \frac{8}{3} + \frac{32}{3k}) \ln(T)} \\ &\leq e^{-\frac{k}{12} \ln T} \quad \text{for } k \geq 28 \\ &= T^{-\frac{k}{12}}. \end{aligned}$$

This shows that any fixed day  $t$  has depth at most  $k \ln T$  in the random partition-tree, with probability at least  $1 - T^{-\frac{k}{12}}$ .

Now, we can take a union bound over the  $T$  days, and say that all  $T$  days have depth at most  $k \ln T$  in the random partition-tree with probability

$$\geq 1 - T \cdot T^{-\frac{k}{12}} = 1 - T^{-\frac{k}{12}+1} \geq 1 - T^{-\frac{k}{24}}.$$

To bound the expected depth of the tree, we can multiply the probability of the depth being low by the depth, and, otherwise, multiply the maximum possible depth of  $T$  by the

remaining probability. Using the above bound for  $k = 96$ , we get

$$\mathbf{E}[\text{depth}] \leq 96 \ln T \cdot (1 - T^{-4}) + T \cdot T^{-4} = O(\log(T)).$$

□

**Lemma 9.3.6** (Work to Maintain Schedule). *Algorithm 9.1 performs at most  $O(T \log T)$  reschedules to the predictions of updates and maintain the predicted schedule of events over the course of the online dynamic updates.*

*Proof.* Lemma 9.3.3 guarantees that at the end of preprocessing, we have at most two events scheduled per day. Call this assignment of events  $\mathbf{a}_0$ . Over the course of the algorithm, events are rescheduled via a doubling search procedure using PROCESSEVENTLATERTHANPREDICTION. For an event  $e$ , call this sequence of reschedulings  $\mathbf{a}_1(e), \mathbf{a}_2(e), \dots$ , where  $\mathbf{a}_i(e) = \mathbf{a}_0(e) + 2^i$ , if it exists. Thus the set of existent  $\mathbf{a}_i(e)$ , for event  $e$ , is some subset of the events in  $\mathbf{a}_0$ , shifted by exactly  $2^i$ . Thus, each  $\mathbf{a}_i$  contributes at most a constant number of scheduled events on each day.

Now, we note that since there are only  $T$  possible days, an event can only be rescheduled  $O(\log T)$  times. Thus,  $\mathbf{a}_i(e)$  only exist for  $i \in [k \cdot \log T]$  for some constant  $k \geq 1$ . Together, this results in at most  $O(\log T)$  events (re)scheduled for any given day. There are  $T$  days, resulting in  $O(T \log T)$  reschedules.

Now, consider the work done by our algorithm to maintain the schedule on a given day  $t$ . The algorithm must potentially reschedule the prediction of one real event for day  $t$  using PROCESSEVENTEARLIERTHANPREDICTION, and reschedule (via the guess and double procedure in PROCESSEVENTLATERTHANPREDICTION) the predicted events that were scheduled for day  $t$  but did not occur. Rescheduling a predicted event using PROCES-

SEVENTHEARLIERTHANPREDICTION is done once per real event; since there are  $T$  real events, this procedure requires  $O(T)$  reschedules. By our previous accounting, there are at most  $O(T \log T)$  events that are rescheduled by PROCESSEVENTLATERTHANPREDICTION. Hence, our algorithm does  $O(T \log T)$  reschedules in total.  $\square$

## A.2 Best-of-All-Worlds Backstop Proofs

**Theorem 9.3.11** (Best-of-All-Worlds Backstop). *Fix  $N$  algorithms  $\{A_1, \dots, A_N\}$ , that will see the same, a priori unknown, input sequence over  $T$  days. Define  $R_{A_i}(t)$  to be the total work that  $A_i$  does over the first  $t$  days.*

*We can design a meta-algorithm  $M$ , such that*

$$\forall t, \quad R_M(t) = O(N \cdot \min\{R_{A_1}(t), \dots, R_{A_N}(t)\}),$$

*where  $R_M(t)$  is the total work that  $M$  does over the first  $t$  days.*

*Proof.* At a high level, our meta-algorithm will update all of the algorithms in synchrony, but it will perform computations at the rate of the faster algorithm. Thus, the faster algorithm will be up-to-date with the current timestep and will provide the output computation; each slower algorithm may have a backlog of computations that it has not yet performed. This is described in [Algorithm 9.3](#).

We fix a day  $t$  and analyze the work done by the meta-algorithm up through the  $t$ -th day. [Algorithm 9.3](#) does  $O(t \cdot N)$  work over  $t$  updates. We can assume that  $R_{A_i}(t) = \Omega(t)$  for all of the algorithms  $A_i$  (where the algorithms perform  $\Omega(1)$  processing per update), and thus the contribution of  $O(t \cdot N)$  to the work will be dominated by faster growing terms.

Algorithm 9.3 maintains the invariant that each of the algorithms  $A_1, \dots, A_N$  has been run for the same number of computation steps. Thus, the first algorithm to complete the computation for all of events in its buffer after the  $t$ -th update will be precisely the algorithm  $A_i$  that minimizes  $R_{A_i}(t)$ . Over the course of all updates up to  $t$ , the meta algorithm has run  $A_i$  for

$$R_{A_i}(t) = \min\{R_{A_1}(t), \dots, R_{A_N}(t)\} \text{ steps.}$$

Since it has run each of the  $N$  algorithms for the same number of steps, it has done total work

$$O(N \cdot \min\{R_{A_1}(t), \dots, R_{A_N}(t)\}).$$

Correctness of the algorithm follows because a solution output on day  $t$  is the solution output by  $A_i$  upon seeing exactly the stream of events up to day  $t$ . Thus, if  $A_i$  is correct, the meta-algorithm is also correct. □

# Appendix B

## Details for Volume Optimality

## Framework for Conformal Prediction

### B.1 Additional Numerical Experiments

#### B.1.1 Construction of Nested Systems

Recall the construction of the nested system described in [Section 3.4.1](#). It immediately follows from [Proposition 3.2.4](#) that the construction satisfies [Assumption 3.2.5](#) in the unsupervised setting.

In the supervised setting, the construction of the nested system is based on  $\hat{F}(y \mid x)$ . For each  $x \in \{X_{n+1}, \dots, X_{2n+1}\}$ , we generate  $Y_1(x), \dots, Y_L(x)$  according to the quantile level

$$Y_l(x) = \operatorname{argmax}\{y : \hat{F}(y \mid x) \leq l/L\}.$$

Then, the greedy expansion and contraction procedure described in [Section 3.4.1](#) is applied



on  $Y_1(x), \dots, Y_L(x)$ . Effectively, this is equivalent to using  $\tilde{F}(y | x) = \frac{1}{L} \sum_{l=1}^L \mathbb{I}\{Y_l(x) \leq y\}$  as input. By its definition,  $\tilde{F}(y | x)$  is a uniform approximation to  $\hat{F}(y | x)$  with error  $1/L$ . Thus, [Assumption 3.3.1](#) is still satisfied for  $\tilde{F}(y | x)$ . In all of our experiments, we set  $L = m$ .

*Remark B.1.1.* It is clear that the details of the greedy expansion step and the greedy contraction step do not matter much for [Assumption 3.2.5](#) or [Assumption 3.3.1](#) to be satisfied. However, different choices will indeed affect practical performance, especially in the supervised setting when  $\hat{F}(y | x)$  is not close to  $F(y | x)$ . To be more specific, sensible choices of expansion and contraction sets from the  $S_{j^*}$  generated by DP will serve as a safety net against model misspecification. We discuss this in more detail in [Section B.1.3](#), see e.g. [Figure B.14](#).

## B.1.2 Unsupervised Setting

Given i.i.d. observations  $Y_1, Y_2, \dots, Y_{2n} \in \mathbb{R}$  drawn from some distribution  $P$ , the goal is to find a prediction set  $\hat{C} = \hat{C}(Y_1, \dots, Y_{2n})$  such that  $\mathbb{P}(Y_{2n+1} \in \hat{C}) \geq 1 - \alpha$  for an independent future observation  $Y_{2n+1}$  drawn from the same  $P$ . We implement the proposed conformalized dynamic programming (DP) method  $\hat{C}_{\text{CP-DP}}$ , and compare it with the conformalized kernel density estimation (KDE) proposed by [Lei, Robins, and Wasserman \[2013\]](#) on the following synthetic datasets:

- (1) Gaussian
- (2) Censored Gaussian
- (3) Mixture of Gaussians
- (4) ReLU-Transformed Gaussian

Though the original conformalized KDE was proposed in the full conformal framework, we will consider its split conformal version for a direct comparison. We believe the comparison between the full conformal versions of the two methods will lead to the same conclusion. For the conformalized DP method, the conformity score is constructed based on the nested system described in [Section B.1.1](#) with  $m = 50$  and  $\delta = \sqrt{(k + \log n)/n}$ . The conformalized KDE is also in the form of (3.8), with the conformity score given by

$$q_{\text{KDE}}(x) = \frac{1}{n\rho} \sum_{i=1}^n K\left(\frac{y - Y_i}{\rho}\right),$$

where  $K(\cdot)$  is the standard Gaussian kernel and  $\rho$  is the bandwidth parameter. Both methods involve a single tuning parameter,  $k$  for conformalized DP and  $\rho$  for conformalized KDE.

**Gaussian.** Our first distribution is  $P = N(0, 1)$ , which is a benign example as a sanity check. We consider sample size being 100, and set the coverage probability  $1 - \alpha = 30\%$  for a more transparent comparison between the two methods. The conformalized DP is computed with number of intervals  $k$  ranging from 1 to 10. It turns out that the output of the prediction set is quite stable when  $k$  varies ([Figure B.2](#)). Even for  $k = 10$ , our method still produces a single interval in this unimodal distribution.

The conformalized KDE is implemented with bandwidth  $\rho$  ranging from 0.001 to 0.005. We observe that the quality of the prediction set is quite sensitive to the choice of the bandwidth. As is shown by [Figure B.1](#), if the bandwidth of KDE is too small, the conformal prediction will output almost the entire support of the data set. This is because if the KDE overfits the training samples, the level set of the KDE will likely not cover the future observation. Therefore, a conformal procedure, which guarantees finite sample coverage, has to be conservative by outputting the entire support. [Figure B.2](#) shows that this issue will be alleviated as the bandwidth gets larger.

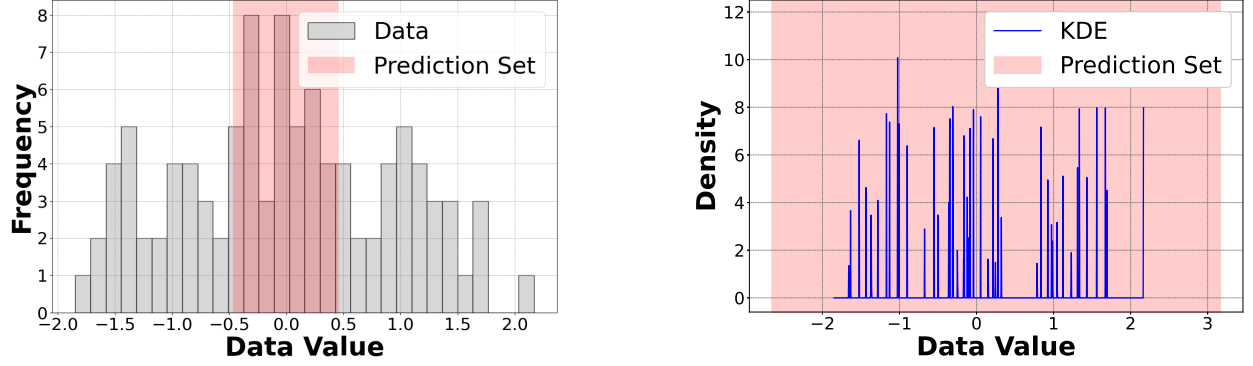


Figure B.1: Conformal prediction sets on the Gaussian dataset. The left plot shows the histogram of the dataset and the prediction set produced by conformalized DP with  $k = 1$ ; the right plot shows the kernel density estimation with bandwidth  $\rho = 0.001$  and the prediction set given by the conformalized KDE.

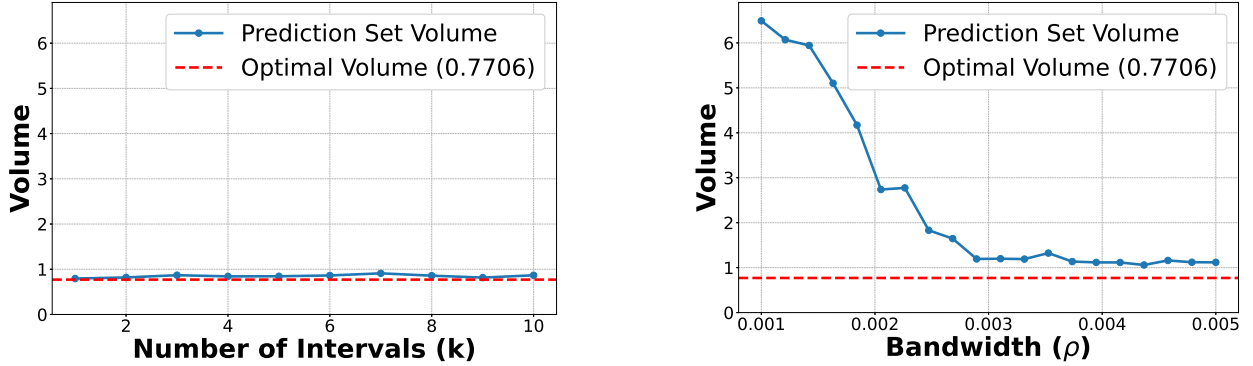


Figure B.2: Volumes of prediction sets of the two methods on the Gaussian dataset (blue) and the benchmark  $\text{OPT}_1(N(0,1), 0.3) = 0.7706$  (red). The blue curves are computed by averaging 100 independent experiments.

**Censored Gaussian.** We next consider  $P$  being a censored Gaussian distribution. We take the sample size to be 100, and each sample can be generated according to

$$Y_i = \sigma(Z_i + 1) - \sigma(Z_i - 1)$$

with  $Z_i \sim N(0,1)$  and  $\sigma(t) = \max(t, 1)$  being the ReLU transform. This is equivalently a truncated Gaussian distribution, which has a standard Gaussian density on  $(0, 2)$  and a

point mass at 0 with probability  $\mathbb{P}(Z_i \leq -1)$  and another point mass at 2 with probability  $\mathbb{P}(Z_i \geq 1)$ . Again, for the sake of comparison, we set the coverage probability to be  $1 - \alpha = 30\%$ .

Since  $\mathbb{P}(|Z_i| \leq -1) \geq 1 - \alpha$ , the population optimal volume is

$$\text{OPT}(P, 0.3) = \text{OPT}_2(P, 0.3) = 0$$

due to the point masses at  $\{0, 2\}$ . By setting  $k = 2$  for the conformalized DP procedure, the prediction set concentrates on the two point masses (Figure B.3). Moreover, it produces very similar results as we increase  $k$  up to 10. Figure B.4 shows that the only exception is  $k = 1$ , since one short interval obviously cannot cover two points that are far away from each other.

We also run conformalized KDE with bandwidth  $\rho$  ranging from 0.001 to 1. Since the distribution does not even have a density function on the entire support, KDE is not really suitable for this setting. Not surprisingly, for a typical choice of bandwidth that is not too small, the conformalized KDE will not identify the two point masses due to smoothing (Figure B.3). Figure B.4 reports the volume of the prediction set as we vary bandwidth, and the volume of the prediction set is close to optimal only when the bandwidth is extremely close to 0.

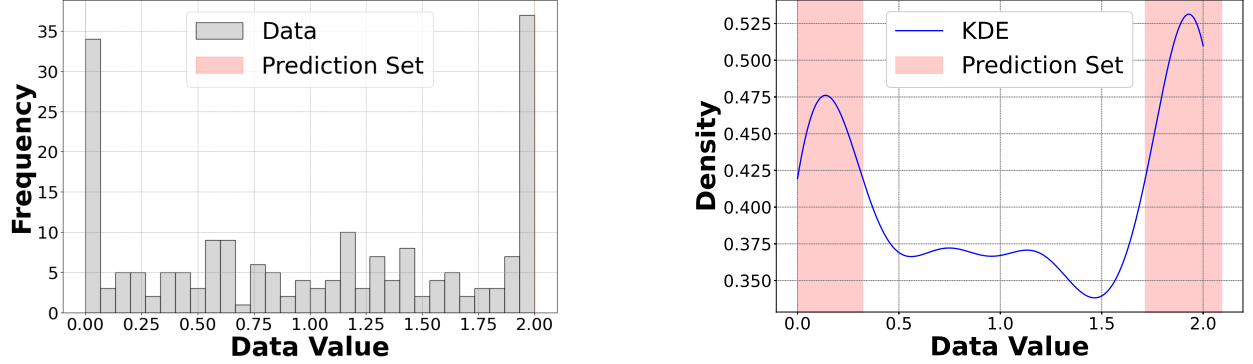


Figure B.3: Conformal prediction sets on the censored Gaussian dataset. The left plot shows the histogram of the dataset and the prediction set given by conformalized DP with  $k = 2$  intervals (The prediction set is two zero-length intervals at 0.0 and 2.0). The right plot shows the kernel density estimation with bandwidth  $\rho = 0.2$  and the prediction set by conformalized KDE.

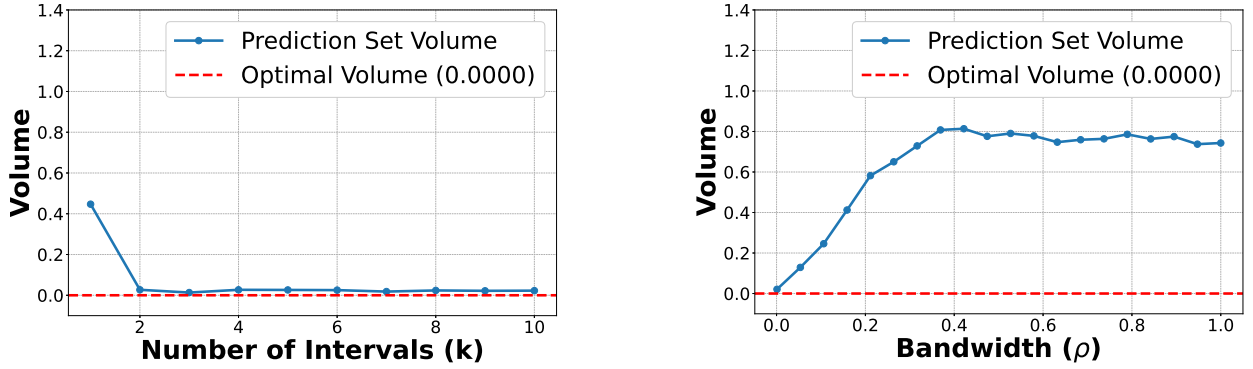


Figure B.4: Volumes of prediction sets of the two methods on the censored Gaussian dataset (blue) and the optimal volume (red). The blue curves are computed by averaging 100 independent experiments.

**Mixture of Gaussians.** In this experiment, we consider

$$P = \frac{1}{3}N(-6, 0.0001) + \frac{1}{3}N(0, 1) + \frac{1}{3}N(8, 0.25).$$

The sample size and coverage probability are set as 600 and  $1 - \alpha = 80\%$ , respectively. The two methods are compared with  $k$  ranging from 1 to 10 in conformalized DP and bandwidth

$\rho$  ranging from 0.001 to 5 in conformalized KDE.

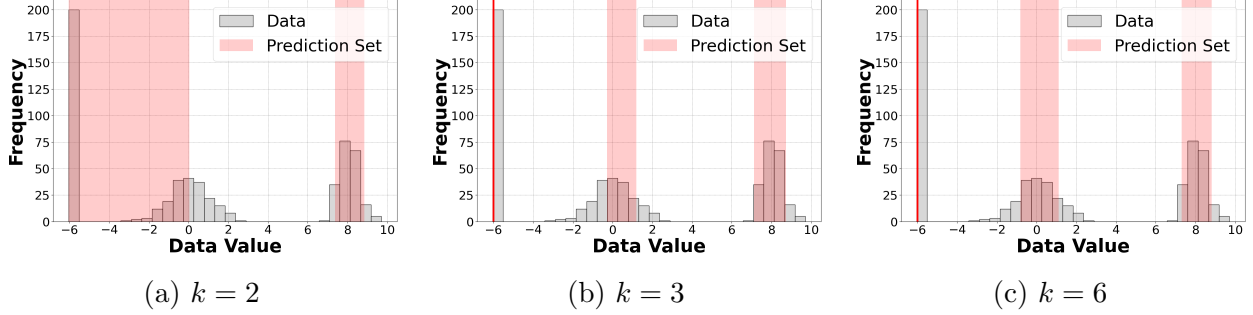


Figure B.5: Prediction sets provided by the conformalized DP method with the number of intervals  $k = 2, 3, 6$  on the mixture of Gaussians dataset.

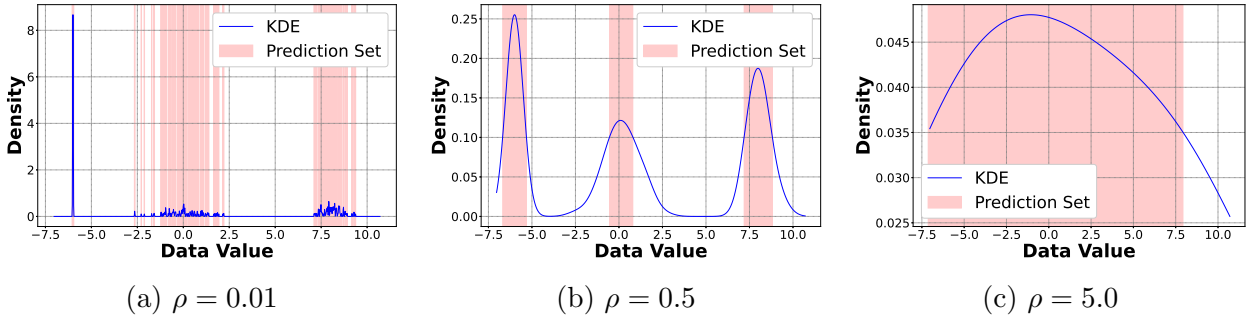


Figure B.6: Prediction sets provided by the conformalized KDE using bandwidth  $\rho = 0.01, 0.5, 5.0$  on the mixture of Gaussians dataset.

We report typical results of conformalized DP with  $k \in \{2, 3, 6\}$  in Figure B.5 and report those of conformalized KDE with  $\rho \in \{0.01, 0.5, 5.0\}$  in Figure B.6. The proposed method based on DP produces similar prediction sets close to optimal as long as  $k \geq 3$  (Figure B.7). This is because  $\text{OPT}(P, 0.8) = \text{OPT}_3(P, 0.8)$  with  $P$  being a Gaussian mixture of three components. In comparison, the results based on KDE are quite sensitive to the bandwidth choice, since different bandwidths lead to kernel density estimators with completely different numbers of modes. Figure B.7 shows that for the optimal choice of bandwidth around 0.5, the KDE successfully identifies the three modes of the Gaussian mixture. However, even with the optimal bandwidth, the volume of the prediction set is in general still greater than

that of the conformalized DP. This is partly because the three components of the Gaussian mixture do not have the same variance parameters, and thus cannot be optimally estimated by KDE with a single bandwidth.

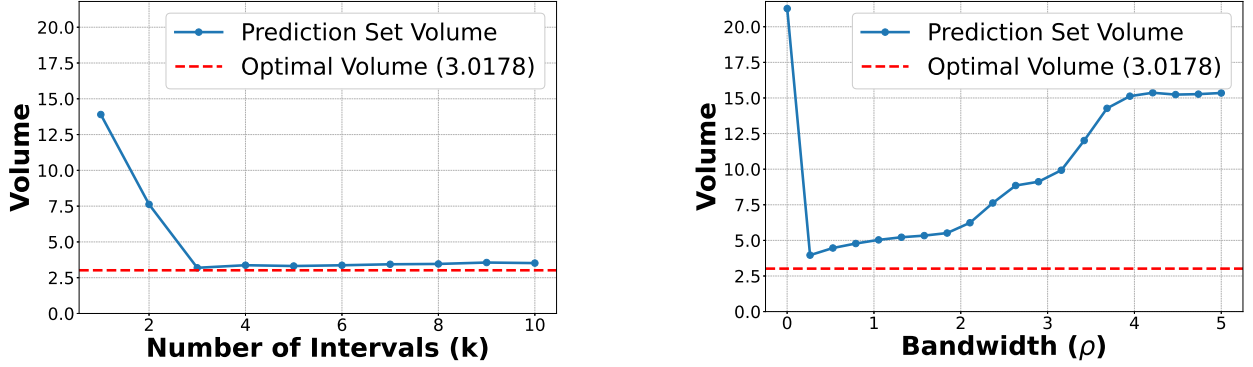


Figure B.7: Volumes of prediction sets of the two methods on the mixture of Gaussians dataset (blue) and the optimal volume  $\text{OPT}(P, 0.8) = 3.0178$  (red). The blue curves are computed by averaging 100 independent experiments.

**ReLU-Transformed Gaussian.** The ReLU-Transformed Gaussian is generated according to

$$X_i = \sum_{j=1}^t a_j * \sigma(w_j * Z_i + b_j),$$

with  $Z_i \sim N(0, 1)$ . It includes the censored Gaussian as a special case. Here, we take  $t = 7$  and take a randomly generated set of coefficients. The resulting density function is plot in Figure B.8a. The sample size and coverage probability are taken as 600 and  $1 - \alpha = 80\%$ , respectively.

Figure B.8 also shows a typical prediction set produced by conformalized DP with  $k = 4$  and one produced by conformalized KDE with bandwidth  $\rho = 0.02$ . Figure B.9 gives a more thorough comparison. The proposed conformalized DP achieves near optimality when  $k \geq 4$ , since the distribution has 4 modes. The KDE solutions are sensitive to the choice of bandwidth for this complicated distribution.

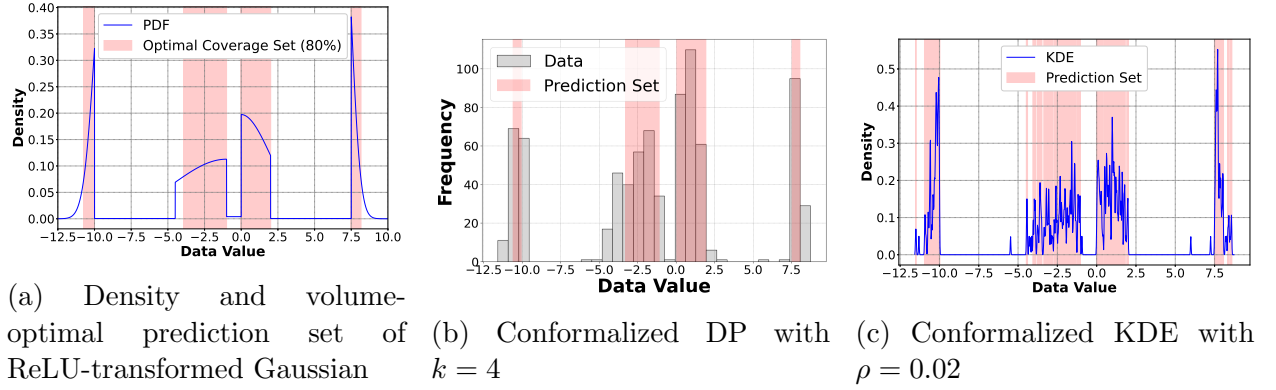


Figure B.8: Prediction sets on the ReLU-Transformed Gaussian

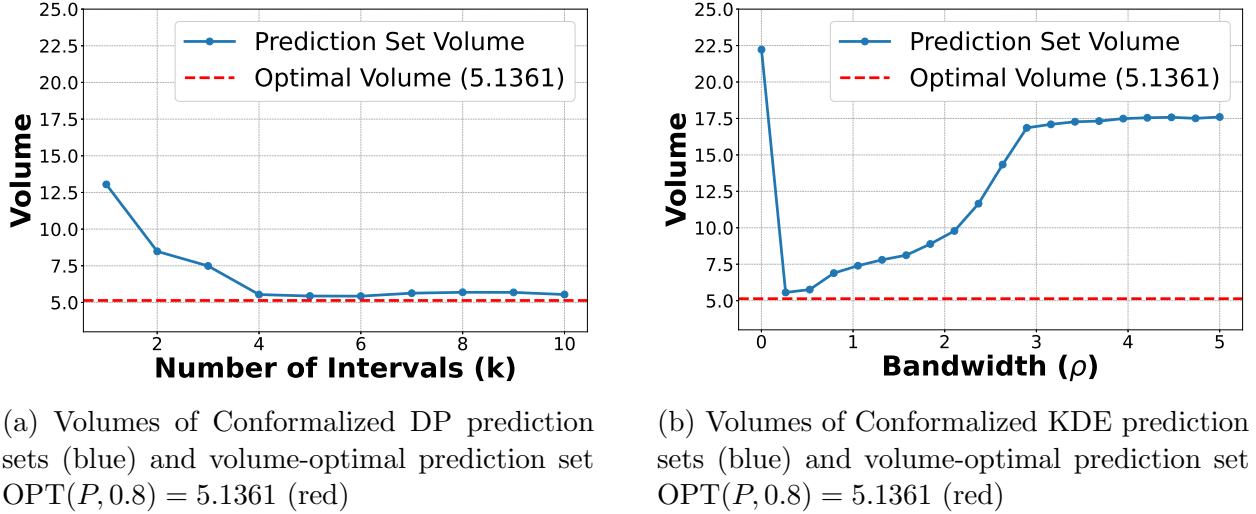


Figure B.9: Method comparison on ReLU-transformed Gaussian. The blue curves are computed by averaging 100 independent experiments.

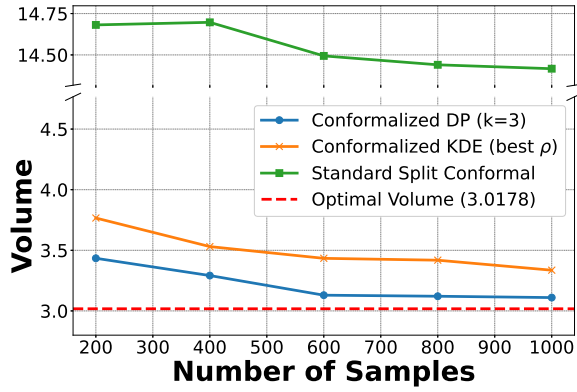
**Effects of Sample Sizes and Coverage Probabilities.** Finally, we study the effects of sample sizes and coverage probabilities for the two methods. Specifically, we examine how the volume of the prediction set decays as the sample size increases and how it varies with different coverage probabilities. The experiments will be conducted with data generated from the following two distributions:

1.  $\frac{1}{3}N(-6, 0.0001) + \frac{1}{3}N(0, 1) + \frac{1}{3}N(8, 0.25).$

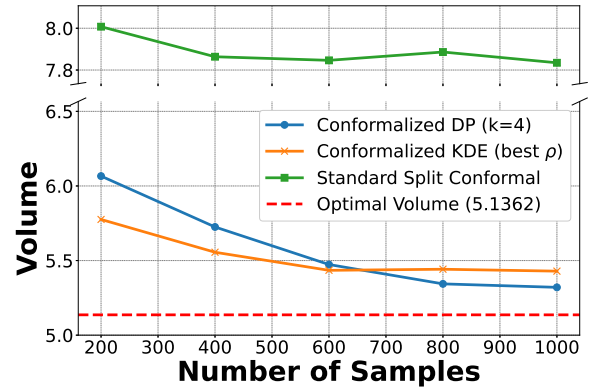


2. The ReLU-Transformed Gaussian  $Y_i = \sum_{j=1}^t a_j * \sigma(w_j * Z_i + b_j)$  with  $Z_i \sim N(0, 1)$ ,  $t = 7$  and coefficients are the same as in the previous experiment.

For conformalized DP, we will set  $k = 3$  for the first distribution and  $k = 4$  for the second one to match the number of modes in the two cases. For conformalized KDE, since the method is sensitive to the choice of bandwidth, we will scan the bandwidth  $\rho$  from 0.001 to 0.2, and only report the one with the smallest volume. We also benchmark the performances of the two methods by the optimal volume and a standard split conformal procedure with conformity score  $q_{\text{standard}}(y) = -|y - \frac{1}{n} \sum_{i=1}^n Y_i|$ .



(a)  $\frac{1}{3}N(-6, 0.0001) + \frac{1}{3}N(0, 1) + \frac{1}{3}N(8, 0.25)$



(b) ReLU-Transformed Gaussian

Figure B.10: Volume of prediction set against sample size. All curves are plotted by averaging results from 100 independent experiments.

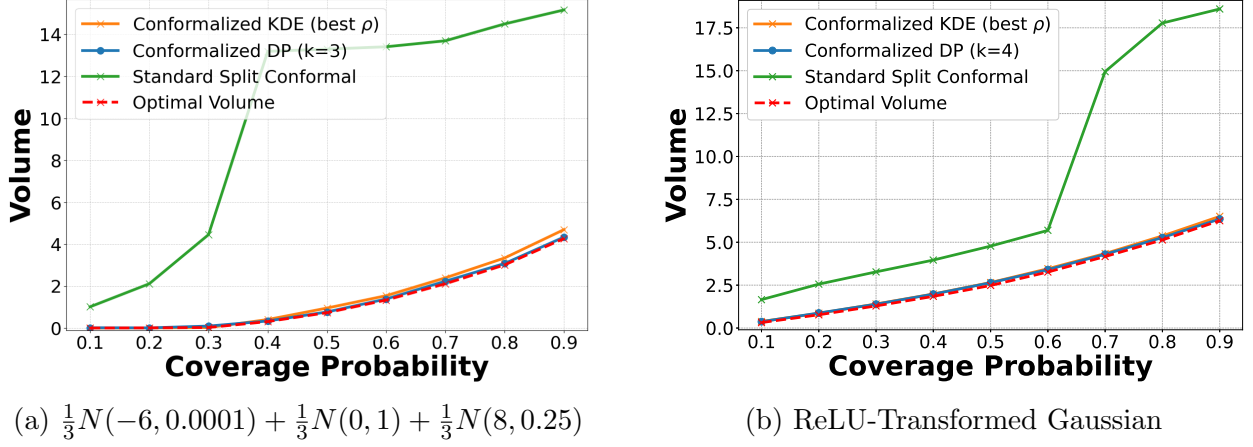


Figure B.11: Volume of prediction set against coverage probability. All curves are plotted by averaging results from 100 independent experiments.

Figure B.10 shows the results with sample size ranging from 200 to 1000 with the coverage probability fixed by  $1 - \alpha = 80\%$ . Both conformalized DP and conformalized KDE produce smaller prediction sets as sample size increase. Even with the bandwidth optimally tuned for conformalized KDE, which is not feasible in practice, the proposed conformalized DP tends to achieve smaller volumes in most cases. In setting of the ReLU-Transformed Gaussian, we observe that the volume of conformalized KDE prediction set barely decreases after sample size 600, since in this case density estimation is very hard for KDE.

Figure B.11 considers coverage probability ranging from 0.1 to 0.9, with sample size fixed at 600. The conformalized DP constantly achieves smaller volume than the conformalized KDE even though the later is computed with optimally tuned bandwidth. This demonstrates the robustness of the conformalized DP in handling varying coverage requirements while maintaining efficiency in volume.

**Real-World Dataset.** We compare the performance of conformalized KDE and conformalized dynamic programming (DP) methods on the Enzyme dataset in Table B.1 and

[Table B.2](#). For conformalized KDE, we report results across a range of bandwidths from 0.1 to 0.9; for conformalized DP, we vary the number of intervals  $k = 1, \dots, 5$ . We output the average and standard deviation for volume and empirical coverage over 50 trials and 80% target coverage. We observe that conformalized DP achieves higher coverage with lower volume, indicating more efficient coverage compared to conformalized KDE.

Table B.1: Conformalized KDE on Enzyme Dataset

| Bandwidth | Volume                            | Coverage            |
|-----------|-----------------------------------|---------------------|
| 0.1       | <b>0.9826</b> $\pm$ <b>0.0969</b> | 0.8081 $\pm$ 0.0217 |
| 0.3       | 1.4640 $\pm$ 0.0940               | 0.8056 $\pm$ 0.0231 |
| 0.5       | 1.5218 $\pm$ 0.1332               | 0.7999 $\pm$ 0.0268 |
| 0.7       | 1.4526 $\pm$ 0.1809               | 0.7962 $\pm$ 0.0272 |
| 0.9       | 1.3867 $\pm$ 0.1799               | 0.7971 $\pm$ 0.0236 |

Table B.2: Conformalized DP on Enzyme Dataset

| $k$ | Volume                            | Coverage            |
|-----|-----------------------------------|---------------------|
| 1   | 1.2207 $\pm$ 0.1188               | 0.8101 $\pm$ 0.0461 |
| 2   | <b>0.8997</b> $\pm$ <b>0.1414</b> | 0.8144 $\pm$ 0.0353 |
| 3   | <b>1.0118</b> $\pm$ <b>0.1975</b> | 0.8416 $\pm$ 0.0390 |
| 4   | <b>1.0131</b> $\pm$ <b>0.1878</b> | 0.8458 $\pm$ 0.0379 |
| 5   | <b>1.0520</b> $\pm$ <b>0.2002</b> | 0.8558 $\pm$ 0.0353 |

### B.1.3 Supervised Setting

In the supervised setting, we validate our results on the simulated datasets in [Romano, Patterson, and Candès \[2019\]](#) and [Izbicki, Shimizu, and Stern \[2020a\]](#). We compare against the

methods of *Conformalized Quantile Regression* (CQR) of [Romano, Patterson, and Candès \[2019\]](#) and *Distributional Conformal Prediction* (DCP) of [Chernozhukov, Wüthrich, and Zhu \[2021\]](#) and CD-split and HPD-split of [Izbicki, Shimizu, and Stern \[2020b\]](#).

**Simulated Dataset** [\[Romano, Patterson, and Candès, 2019\]](#). We first describe the simulated dataset in [Romano, Patterson, and Candès \[2019\]](#). In this data, each one-dimensional predictor variable  $X_i$  is sampled uniformly from the range  $[0, 5]$ . The response variable is then sampled according to

$$Y_i \sim \text{Pois}(\sin^2(X_i) + 0.1) + 0.03 X_i \varepsilon_{1,i} + 25 \mathbf{1}\{U_i < 0.01\} \varepsilon_{2,i},$$

where  $\text{Pois}(\lambda)$  is the Poisson distribution with mean  $\lambda$ ,  $\varepsilon_{1,i}$  and  $\varepsilon_{2,i}$  are independent standard Gaussian noise, and  $U_i$  is drawn uniformly on the interval  $[0, 1]$ . The first component of the distribution,  $\text{Pois}(\sin^2(X_i) + 0.1)$ , generates a distribution that is clustered around positive integer values of  $Y$ , with variance that changes periodically in  $X$ . The second component of the distribution,  $0.03 X_i \varepsilon_{1,i}$ , adds some additional variance to each of the integer centered clusters, where the magnitude of the variance increases with  $X$ . The final component,  $25 \mathbf{1}\{U_i < 0.01\} \varepsilon_{2,i}$ , adds a small fraction of outliers to the distribution. We generate 2000 training examples, and 5000 test examples, as in the work of [Romano, Patterson, and Candès \[2019\]](#). The same subset of training and test examples are used in the illustration of each of these methods. The set of test examples is visualized in [Figure B.12](#), with the full range of  $Y$  values including the outliers. In the plots associated with our conformal output, we zoom in on the  $Y$  axis for readability, leaving the outliers off the chart.

**Simulated Dataset** [\[Izbicki, Shimizu, and Stern, 2020a\]](#). We now describe the simulated dataset in [Izbicki, Shimizu, and Stern \[2020a\]](#). In this data, the predictor variables  $X = (X_1, \dots, X_d)$  with  $d = 20$  dimensions are independently and uniformly sampled from

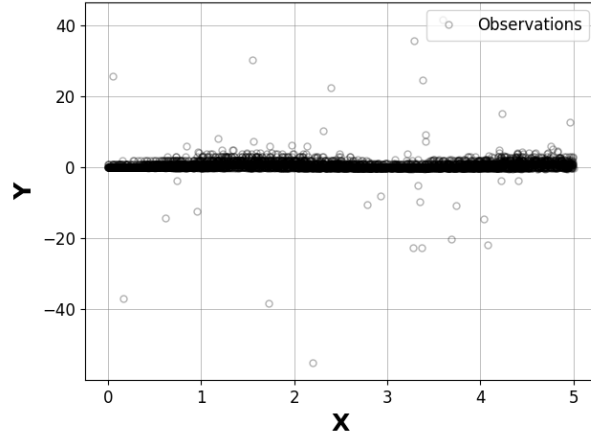


Figure B.12: Simulated data of [Romano, Patterson, and Candès \[2019\]](#), including outliers.

the range  $[-1.5, 1.5]$ . The response variable  $Y$  is then generated according to the following bimodal conditional distribution:

$$Y \mid X \sim 0.5\mathcal{N}(f(X) - g(X), \sigma^2(X)) + 0.5\mathcal{N}(f(X) + g(X), \sigma^2(X)).$$

where the functions  $f(X)$ ,  $g(X)$ , and  $\sigma^2(X)$  are defined as:

$$f(X) = (X_1 - 1)^2(X_1 + 1), \quad g(X) = 2\mathbb{I}(X_1 \geq -0.5)\sqrt{X_1 + 0.5}, \quad \sigma^2(X) = \frac{1}{4} + |X_1|.$$

Here,  $\mathcal{N}(\mu, \sigma^2)$  denotes a normal distribution with mean  $\mu$  and variance  $\sigma^2$ , and the indicator function  $\mathbb{I}(X_1 \geq -0.5)$  accounts for the bimodal nature of the data, introducing a piecewise behavior in the response variable. The first term  $f(X)$  captures a polynomial relationship with  $X_1$ , while the second term  $g(X)$  introduces an asymmetric bimodal effect depending on the value of  $X_1$ . The variance  $\sigma^2(X)$  increases linearly with  $|X_1|$ , adding heteroscedasticity to the distribution.

We generate 2000 training examples and 5000 test examples. The same training and test sets are used consistently across all experiments to ensure reproducibility. The test set is

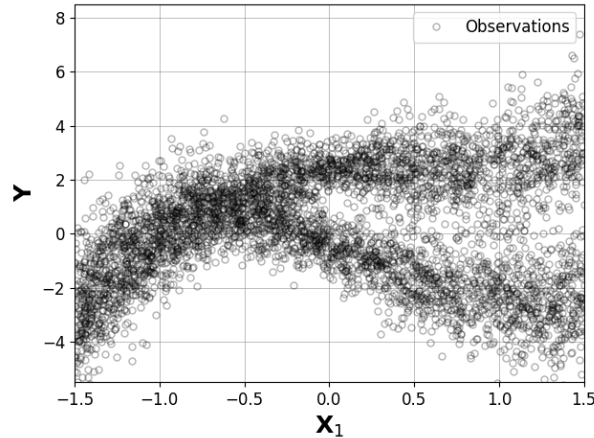


Figure B.13: Simulated data from [Izbicki, Shimizu, and Stern \[2020a\]](#), illustrating the bimodal distribution of the response variable.

visualized in [Figure B.13](#), showcasing the full range of  $Y$  values, including the effects of bimodality and variance heterogeneity.

**Methods.** We compare our conformalized DP with the following methods:

- *Conformalized Quantile Regression* (CQR) of [Romano, Patterson, and Candès \[2019\]](#),
- *Distributional Conformal Prediction via Quantile Regression* (DCP-QR) and *Optimal Distributional Conformal Prediction via Quantile Regression* (DCP-QR\*) of [Chernozhukov, Wüthrich, and Zhu \[2021\]](#),
- *CD-split* and *HPD-split* of [Izbicki, Shimizu, and Stern \[2020b\]](#).

We now describe the implementation of these methods. The compared methods, CQR, DCP-QR, DCP-QR\*, and our conformalized DP rely on quantile regression. For simulated dataset [[Romano, Patterson, and Candès, 2019](#)] with single dimensional predictor variable, we use the package `sklearn-quantile` [[Roebroek, 2023](#)] to implement the quantile regression, which implements the method of *Quantile Regression Forests*, due to [Meinshausen](#)

[2006]. The CD-Split and HPD-Split methods require the conditional density estimation, which is achieved by the R package `FlexCoDE` [Izbicki and Lee, 2017]. For simulated dataset [Izbicki, Shimizu, and Stern, 2020a] with high dimensional predictor variables, the quantile regression by `sklearn-quantile` is not informative. For CQR, DCP-QR, DCP-QR\*, and our conformalized DP, we first use the R package `FlexCoDE` to generate the conditional density estimation and then integrate the conditional density estimation to get quantile regression and conditional CDF. The CD-Split and HPD-Split methods again use the conditional density estimation provided by `FlexCoDE`. All methods are implemented within the split conformal framework, where the training data is randomly divided into two equal parts. Specifically, half of the data is allocated for model training, while the remaining half is used as the calibration set to ensure valid coverage guarantees.

For convenience, we will refer to the  $q$ th estimated quantile of  $Y$  given  $X = x$  as  $\hat{Q}(q, x)$ . Some of the following methods use quantile regression to estimate the whole conditional c.d.f. of  $Y$  given  $X$ , by estimating a set of quantiles from a fine grid. This gives us an estimate of the conditional c.d.f., which gives us access to  $\hat{F}(y | x)$ , the inverse of  $\hat{Q}$ . (That is,  $\hat{F}(y | x) = q$ , such that  $y = \hat{Q}(q, x)$ . Since we only have  $\hat{Q}$  for values of  $q$  in the grid, we set  $\hat{F}(y | x)$  to be the smallest  $q$  in the grid such that  $y \leq \hat{Q}(q, x)$ .)

- Conformalized Quantile Regression (CQR) [Romano, Patterson, and Candès, 2019]: This method fits a model to two quantiles of the data,  $q_{\text{low}} = \frac{\alpha}{2}$  and  $q_{\text{high}} = 1 - \frac{\alpha}{2}$ . On a new test example  $X_{\text{test}}$ , CQR uses the model to estimate the low and high quantile, and the conformal procedure will output the interval

$$\left[ \hat{Q}(q_{\text{low}}, X_{\text{test}}) - b, \hat{Q}(q_{\text{high}}, X_{\text{test}}) + b \right],$$

where  $b$  is a buffer value chosen in the calibration step of the conformal procedure to

guarantee coverage.

- **Distributional Conformal Prediction via Quantile Regression (DCP-QR)** [Chernozhukov, Wüthrich, and Zhu, 2021]: In this framework, we assume access to a model  $\hat{F}$  that can estimate the conditional c.d.f. of the distribution of  $Y$  given  $X$ , which we estimate via quantile regression. Similar to CQR, we start with  $q_{\text{low}} = \frac{\alpha}{2}$  and  $q_{\text{high}} = 1 - \frac{\alpha}{2}$ . In DCP, instead of adding the buffer in the  $Y$  space, the buffer is added in the quantile space. That is, on a new test example  $X_{\text{test}}$ , DCP will output the interval

$$\left[ \hat{Q}(q_{\text{low}} - b, X_{\text{test}}), \hat{Q}(q_{\text{high}} + b, X_{\text{test}}) \right],$$

where  $b$  is a buffer value chosen in the calibration step of the conformal procedure to guarantee coverage.

- **Optimal Distributional Conformal Prediction via Quantile Regression (DCP-QR\*)** [Chernozhukov, Wüthrich, and Zhu, 2021]: The optimal DCP is very similar to DCP, except that  $q_{\text{low}}$  and  $q_{\text{high}}$  need not be symmetric around the median ( $q = \frac{1}{2}$ ). Instead, they are chosen to provide the minimum volume interval that achieves the desired coverage. We note that the buffer is still applied symmetrically in the quantile space. That is, the lower quantile is lowered by some value  $b$ , and the upper quantile is raised by the same value  $b$ .
- **CD-Split** [Izbicki, Shimizu, and Stern, 2020a,b]: This method provides prediction sets based on the conditional density estimation and a partitioning of the feature space. The conformity score in CD-split is based on a conditional density estimator, which allows the method to approximate the highest predictive density (HPD) set. The feature space is partitioned based on the profile of the conditional density estimator, and the cut-off



values are computed locally within each partition. This approach enables CD-split to achieve local and asymptotic conditional validity while providing more informative prediction sets, especially for multimodal distributions.

- HPD-Split [Izbicki, Shimizu, and Stern, 2020b]: The HPD-split method outputs prediction sets based on the highest predictive density (HPD) sets of the conditional density estimation. Unlike CD-split, which partitions the feature space, HPD-Split uses the conformity score based on the conditional CDF of the condition density estimator. Since this conditional CDF is independent of the feature variable, HPD-Split does not require the partition of the feature space and tuning parameters for that as in CD-Split. When the conditional density estimation is accurate, HPD-Split converges to the highest predictive density (HPD) sets.
- Conformalized Dynamic Programming,  $k = 1$  and  $k = 5$ : We implement a modification of the procedure described in this work. In the unsupervised setting, we described the dynamic programming procedure that outputs the minimum volume set of  $k$  intervals that contain a desired fraction of samples. In this setting, given a new test example  $X$ , we do not have access to samples. Instead, we have access to a grid of estimated quantiles of  $Y$  given  $X$ . We implement a version of the dynamic programming procedure that operates on this quantile grid instead of a set of points, to output the minimum volume set of  $k$  intervals that cover at least the desired probability mass. We can also modify our greedy contraction and expansion procedures to provide a nested system of sets for different coverage levels.

**Discussion.** The results of our experiments are illustrated in [Figure B.15](#). Our experiments show that Conformalized Quantile Regression (CQR) and Distributional Conformal Prediction via Quantile Regression (DCP-QR) perform approximately as well as each other

on this dataset, achieving average volume 1.42 and 1.48 respectively (see [Figure B.15a](#), [Figure B.15b](#)).

Optimal Distributional Conformal Prediction via Quantile Regression (DCP-QR\*) achieves a significant improvement over DCP-QR on this data, achieving average volume 1.29 (see [Figure B.15c](#)). This is due to the fact that the distribution of  $Y$  values is not symmetric around, or peaked at the median  $Y$  value. Thus, DCP-QR suffers a disadvantage, because it outputs intervals that are centered around the median in quantile space, and does not take into account the relative volumes of the quantiles in  $Y$  space. DCP-QR\* on the other hand, is able to take advantage of the fact that, for this data, quantiles close to 0 have very low volume, and output intervals that use these quantiles.

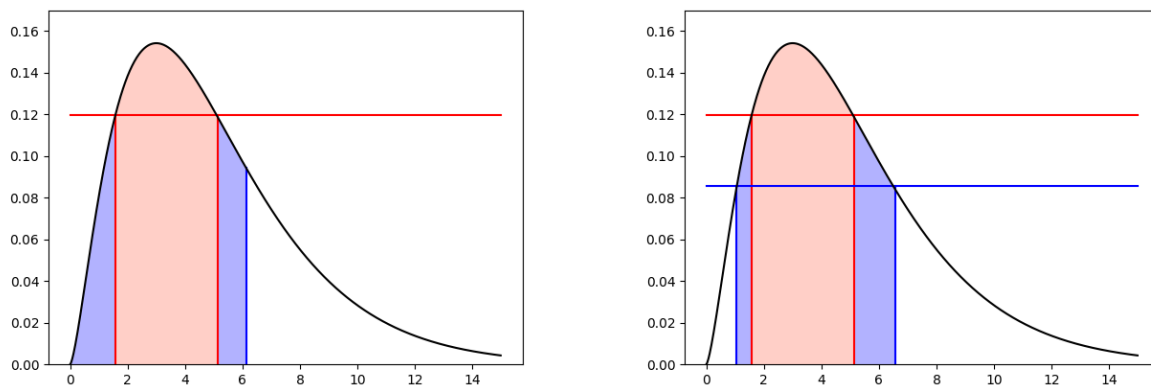
While DCP-QR\* uses information about the relative volume of the quantiles to choose  $q_{\text{low}}$  and  $q_{\text{high}}$ , which define the output intervals before conformalization, it does not take the volume into account during the conformalization step. Expanding the interval by a buffer value  $b$  that is small in quantile space, can lead to a large difference in  $Y$  space, increasing the volume of the output interval significantly. For example in [Figure B.15c](#), the intervals for  $X$  just larger than 4 stretch very far into the negative  $Y$  region, as a small adjustment in quantile space is a large adjustment in  $Y$  space.

This issue is avoided by our Conformalized Dynamic Programming (Conformalized DP) method with greedy expansion and contraction for  $k = 1$  interval. Before conformalization, the interval output by Conformalized DP and DCP-QR\* is the same: it is the volume optimal interval that achieves a given coverage according to the estimated c.d.f.. However, our method takes the relative volume of different quantiles into account in the conformalization step, and avoids the issue of expanding the interval in quantile space in directions that add too much volume in  $Y$  space. This allows the method to achieve an improved average volume

of 1.14 (see [Figure B.15f](#)).

An illustration of this issue is given in [Figure B.14](#). Suppose that for a new test example  $X$ , the estimated conditional distribution of  $Y$  is skewed. (In this illustration it is  $\chi^2(5)$ .) Suppose that our target coverage was 0.5, and in the calibration phase we are required to expand coverage to 0.7. Both ConformalizedDP and DCP-QR\* will start by calculating the minimum volume interval that captures 0.5 of the probability mass. In this case it is the red region from  $x = 1.58$  to  $x = 5.14$  (i.e., the set of  $x$  such that  $f(x) > 0.12$ , where  $f(x)$  is the p.d.f. of the distribution). Then, each method must expand this interval to capture 0.7 of the probability mass. DCP-QR\* does this by adding two blue regions, each of which capture an additional 0.1 probability mass. This results in expanding the interval significantly to the right, even though the density is low. ConformalizedDP takes the volume (i.e., density) into account when expanding the interval, and produces the minimum volume interval that captures 0.7 of the distribution (i.e., the set of  $x$  such that  $f(x) > 0.085$ ), in this example. (We note that the expansion and contraction procedure of ConformalizedDP does not always result in the volume optimal prediction set for the adjusted coverage, only the original target coverage. However, in this case, since the distribution is unimodal and  $k = 1$ , we do indeed recover the volume optimal set even for the adjusted coverage.)

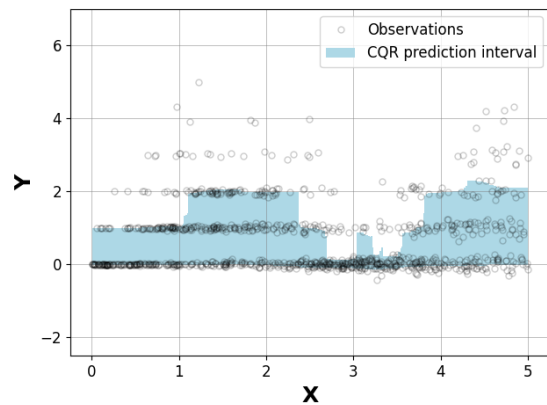
Finally, we also implement Conformalized DP with  $k = 5$  intervals. This allows us to fit to the multimodal shape of the  $Y$  data, and achieve a much lower average volume of 0.45 (see [Figure B.15g](#)).



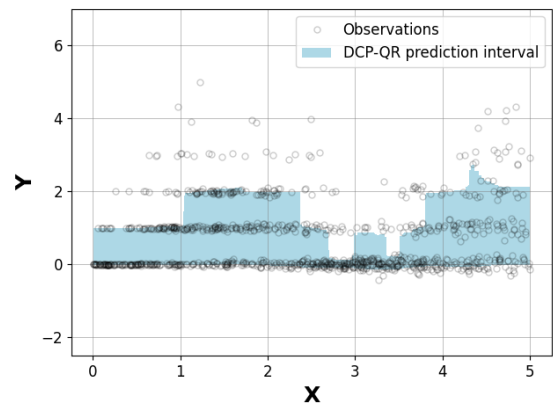
(a) When expanding from the red region, coverage 0.5, to a region of coverage 0.7, DCP-QR\* chooses the blue region with additional volume 2.56.

(b) When expanding from the red region, coverage 0.5, to a region of coverage 0.7, Conformalized DP chooses the blue region with additional volume 1.96.

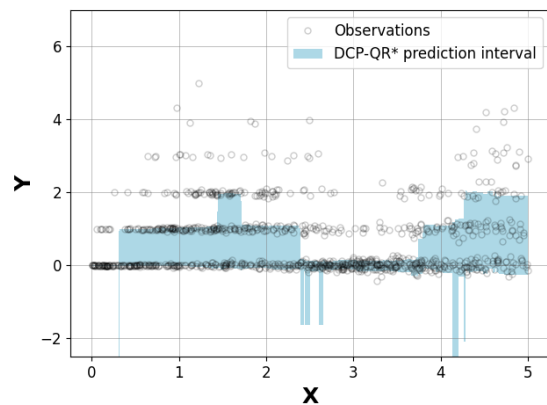
Figure B.14: We illustrate the difference between DCP-QR\* and Conformalized DP for  $k = 1$ , on the example where the estimated conditional distribution of  $Y$  for a new  $X_{\text{test}}$  is  $\chi^2(5)$ . We plot the intervals that are chosen by the methods against the p.d.f. of the estimated distribution.



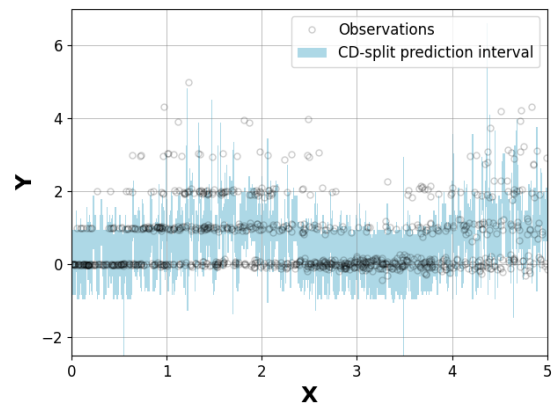
(a) Conformalized Quantile Regression (CQR) [Romano, Patterson, and Candès, 2019], achieves average volume 1.42 and empirical coverage 70.62%.



(b) Distributional Conformal Prediction (DCP) [Chernozhukov, Wüthrich, and Zhu, 2021], achieves average volume 1.48 and empirical coverage 71.6%.

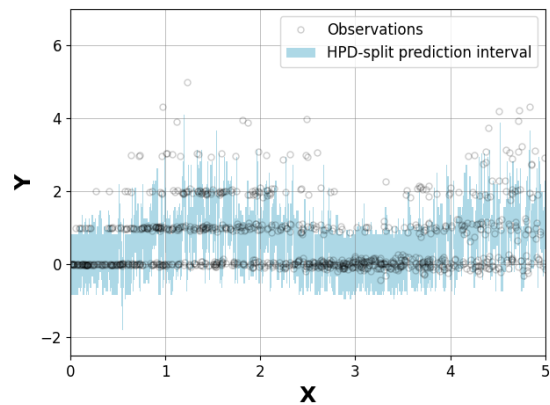


(c) Optimal Distributional Conformal Prediction (DCP-QR\*) [Chernozhukov, Wüthrich, and Zhu, 2021], achieves average volume 1.29 and empirical coverage 71.06%.

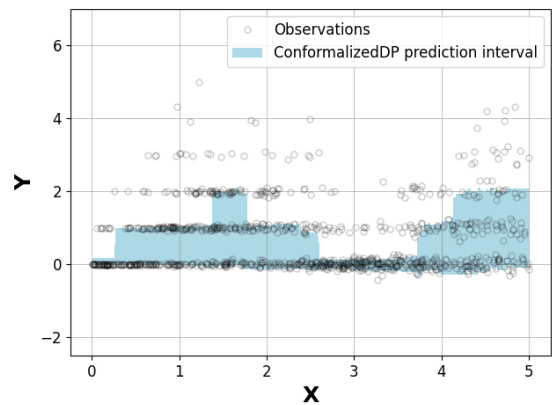


(d) CD-split Conformal Prediction [Izbicki, Shimizu, and Stern, 2020b], achieves average volume 1.83 and empirical coverage 69.94%.

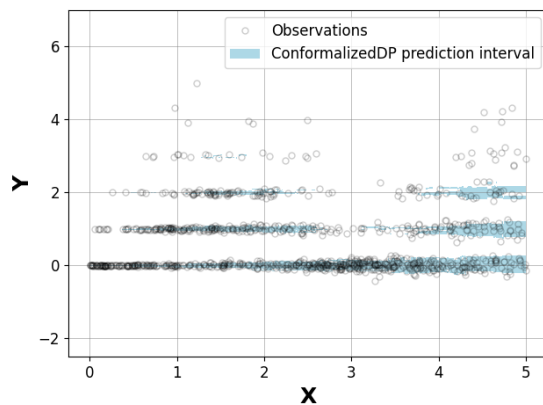
Figure B.15: Continued on following page.



(e) HPD-split Conformal Prediction [Izbicki, Shimizu, and Stern, 2020b], achieves average volume 1.75 and empirical coverage 69.44%.

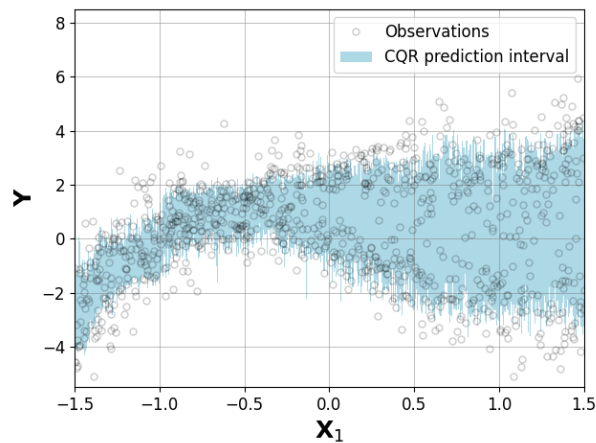


(f) Conformalized Dynamic Programming ( $k = 1$ ), achieves average volume 1.14 and empirical coverage 74.04%.

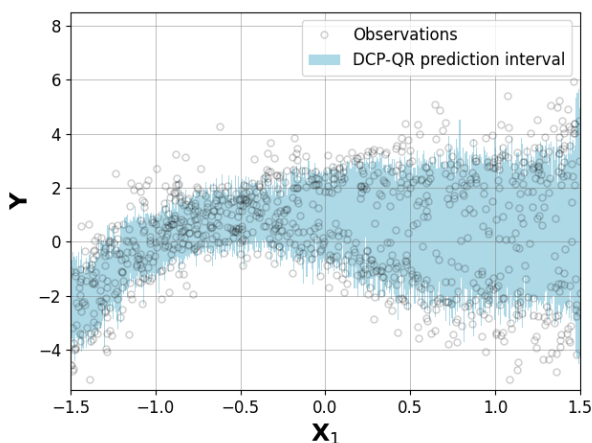


(g) Conformalized Dynamic Programming ( $k = 5$ ), achieves average volume 0.45 and empirical coverage 72.36%.

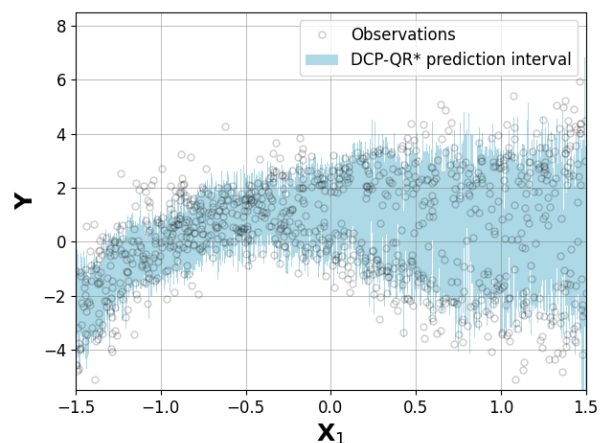
Figure B.15: Comparison of supervised conformal prediction methods on simulated data from Romano, Patterson, and Candès [2019]. All results are for a target coverage of 0.70.



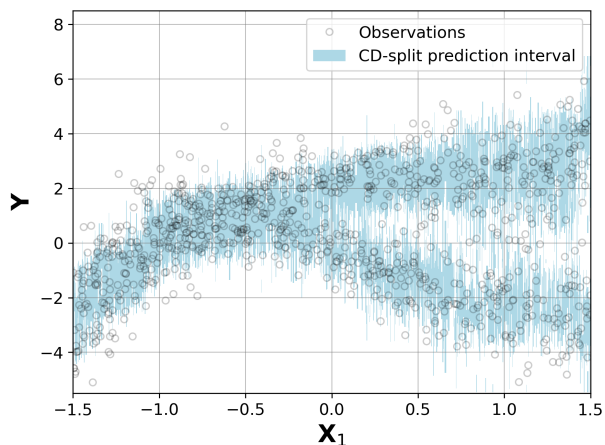
(a) CQR, achieves average volume 4.10 and empirical coverage 71.54%.



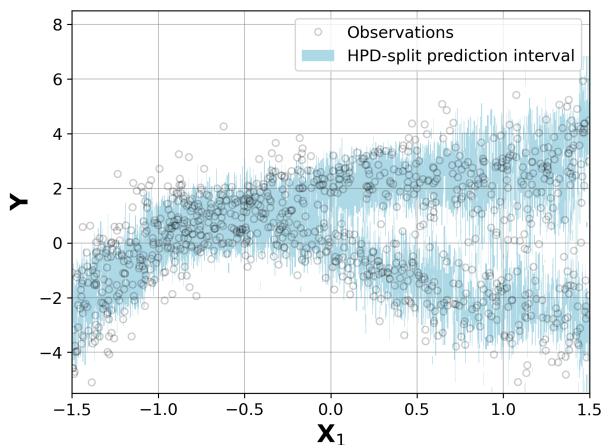
(b) DCP-QR, achieves average volume 4.04 and empirical coverage 70.85%.



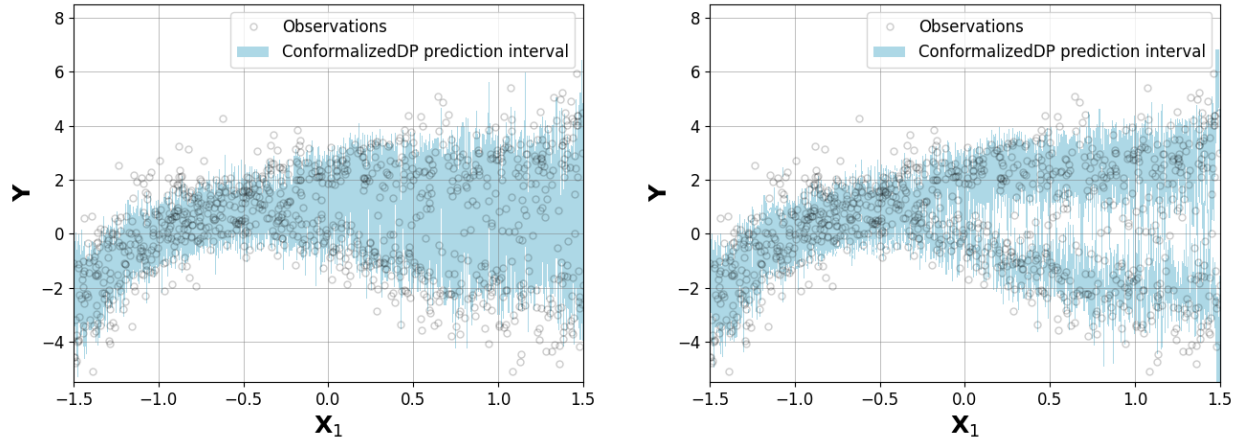
(c) DCP-QR\*, achieves average volume 4.05 and empirical coverage 69.66%.



(d) CD-Split, achieves average volume 3.69 and empirical coverage 69.86%.



(e) HPD-Split, achieves average volume 3.60 and empirical coverage 69.64%.



(f) Conformalized Dynamic Programming ( $k = 1$ ), achieves average volume 4.00 and empirical coverage 68.98%.

(g) Conformalized Dynamic Programming ( $k = 2$ ), achieves average volume 3.55 and empirical coverage 69.42%.

Figure B.16: Comparison of supervised conformal prediction methods on simulated data from [Izbicki, Shimizu, and Stern \[2020a\]](#). All results are for a target coverage of 0.70.

**Real-World Dataset** [[Dhillon, Deligiannidis, and Rainforth, 2024](#)]. We evaluate our method along with several baselines on two real-world regression datasets from the UCI repository: AirFoil and WineQuality. These two datasets were previously considered in the conformal prediction study by [Dhillon, Deligiannidis, and Rainforth \[2024\]](#). For each dataset, we compare the empirical coverage, average volume, and runtime of the prediction sets output by different methods, with a shared preprocessing time (for conditional CDF estimation using **FlexCoDE**) excluded from reported runtimes. For AirFoil and WineQuality, we repeat each experiment 20 times and report the mean and standard deviation of all results with target coverage 0.8.

Recall that our method for the supervised setting requires a conditional CDF estimator. When the feature space is high-dimensional in our experiments, this is constructed by using as a black-box the conditional density estimator of [Izbicki and Lee \[2017\]](#) that underlies CD-split and HPD-split. When the conditional density estimate is used as a black-box, the



optimal volume set is produced by taking an appropriate level set of the conditional density as taken by CD-split and HPD-split; hence we do not expect our method to outperform CD-split and HPD-split in these experiments. However, we observe that our method is still competitive in these experiments, as described below.

*Interpretation:* Among the competitive methods, CQR, DCP-QR, DCP-QR\* all output a single interval and are more suitable for unimodal distributions. On the other hand, CD-split and HPD-split output a union of intervals and are suitable for data generated by multimodal distributions. The experiments show that our proposed framework of Dynamic Programming is competitive against both settings with different values of  $k$ . To be specific, for the dataset in Table B.3, DP with  $k = 1$  is competitive with the DCP-QR and DCP-QR\* which achieves the smallest volume, while in Table B.4, DP with  $k = 5$  is competitive with HPD-split and CD-split which achieve the smallest volume. In practice, the value of  $k$  can be tailored based on the distribution, or chosen adaptively using the ideas in Bai, Mei, Wang, Zhou, and Xiong [2022].

Table B.3: Comparison of methods on the dataset AirFoil. Reported runtimes exclude a shared preprocessing time of  $6.3819 \pm 0.1456$ (s) for estimating the conditional CDF using FlexCoDE common to all methods.

| Method         | Average Volume                        | Empirical Coverage  | Runtime (s)             |
|----------------|---------------------------------------|---------------------|-------------------------|
| CQR            | <b>7.1167 <math>\pm</math> 0.4479</b> | 0.8061 $\pm$ 0.0266 | 0.3536 $\pm$ 0.0528     |
| DCP-QR         | <b>6.4662 <math>\pm</math> 0.4478</b> | 0.8032 $\pm$ 0.0268 | 0.8185 $\pm$ 0.0613     |
| DCP-QR*        | <b>6.4612 <math>\pm</math> 0.4078</b> | 0.7953 $\pm$ 0.0240 | 0.9003 $\pm$ 0.0555     |
| CD-split       | 7.5517 $\pm$ 0.5343                   | 0.7962 $\pm$ 0.0269 | 0.1311 $\pm$ 0.0110     |
| HPD-split      | <b>7.3650 <math>\pm</math> 0.6549</b> | 0.8013 $\pm$ 0.0278 | 1.8243 $\pm$ 0.0556     |
| DP ( $k = 1$ ) | <b>6.8287 <math>\pm</math> 0.3548</b> | 0.8065 $\pm$ 0.0245 | 32.3420 $\pm$ 0.1109    |
| DP ( $k = 5$ ) | 7.5002 $\pm$ 0.5302                   | 0.8032 $\pm$ 0.0211 | 206.3317 $\pm$ 196.6624 |

Table B.4: Comparison of methods on the dataset WineQuality. Reported runtimes exclude a shared preprocessing time of  $68.3879 \pm 2.8636$ (s) for estimating the conditional CDF using FlexCoDE common to all methods.

| Method          | Average Volume                        | Empirical Coverage  | Runtime (s)             |
|-----------------|---------------------------------------|---------------------|-------------------------|
| CQR             | $1.1251 \pm 0.2066$                   | $0.8058 \pm 0.0135$ | $5.0441 \pm 0.5437$     |
| DCP-QR          | $0.9980 \pm 0.0771$                   | $0.8042 \pm 0.0122$ | $7.2799 \pm 0.7250$     |
| DCP-QR*         | $0.9514 \pm 0.0721$                   | $0.8030 \pm 0.0115$ | $7.6734 \pm 0.9398$     |
| CD-split        | <b><math>0.5066 \pm 0.7077</math></b> | $0.7759 \pm 0.0368$ | $2.6191 \pm 0.1664$     |
| HPD-split       | <b><math>0.2694 \pm 0.0163</math></b> | $0.8010 \pm 0.0154$ | $9.0277 \pm 0.0973$     |
| C-DP( $k = 1$ ) | $0.9389 \pm 0.0555$                   | $0.8074 \pm 0.0134$ | $191.8998 \pm 213.9385$ |
| C-DP( $k = 5$ ) | <b><math>0.2947 \pm 0.0215</math></b> | $0.8048 \pm 0.0127$ | $704.8169 \pm 40.4745$  |

## B.2 KDE Optimality Implies DP Optimality

Suppose a distribution  $P$  on  $\mathbb{R}$  admits a density function  $p$ . The kernel density estimator depending on  $k$  i.i.d. samples  $Z_1, \dots, Z_k$  is defined by

$$p_k(y) = \frac{1}{k\rho} \sum_{j=1}^k K\left(\frac{y - Z_j}{\rho}\right), \quad (\text{B.1})$$

where  $K(\cdot)$  is a standard Gaussian kernel and  $\rho$  is a bandwidth parameter. The conformal prediction method by [Lei, Robins, and Wasserman \[2013\]](#) is based on the idea that the level set of  $p_k$  is close to that of  $p$  as long as  $p_k$  is close to  $p$ . In this section, we will show that as long as  $p_k$  is close to  $p$ , the dynamic programming also finds a prediction set whose volume is nearly optimal compared with the level set of  $p$ . This implies that DP always requires no stronger assumption to achieve volume optimality.

The existence of a KDE close to  $p$  can be even weakened into the following assumption.

**Assumption B.2.1.** *For any positive integer  $k$ , there exists some  $\varepsilon_k > 0$  and some Gaussian mixture  $P_k = \sum_{j=1}^k w_j N(\mu_j, \sigma_j^2)$  such that  $\text{TV}(P_k, P) \leq \varepsilon_k$ .*

In particular, the KDE (B.1) based on  $k$  samples is a special case of the Gaussian mixture, given that Gaussian kernel is used. Though the characterization of the closeness between  $p_k$  and  $p$  is through  $\ell_\infty$  norm by [Lei, Robins, and Wasserman \[2013\]](#), similar error bounds also apply to the  $\ell_1$  norm, which is the total variation distance. For example, suppose  $P$  has bounded support and the Hölder smoothness is  $\beta \in (0, 2]$ . Then, one can take  $\varepsilon_k = \tilde{\Theta}\left(k^{-\frac{\beta}{2\beta+1}}\right)$  with an appropriate choice of the bandwidth, where  $\tilde{\Theta}$  hides some logarithmic factor of  $k$ .

**Theorem B.2.2.** *Consider i.i.d. observations  $Y_1, \dots, Y_n, Y_{n+1}$  generated by some distribution  $P$  on  $\mathbb{R}$  that satisfies [Assumption B.2.1](#). For any  $\alpha, \delta, \gamma \in (0, 1)$  such that  $\delta \gg \sqrt{\frac{k+\log n}{n}}$  and  $\gamma + 2\delta + 2\varepsilon_k < \alpha$ , let  $\hat{C}_{\text{DP}} \in \mathcal{C}_k$  be the output of [Algorithm 3.1](#) with coverage level  $1 - \alpha + \delta$  and slack  $\gamma$ . Then, with probability at least  $1 - \delta$ , we have*

1.  $\mathbb{P}\left(Y_{n+1} \in \hat{C}_{\text{DP}} \mid Y_1, \dots, Y_n\right) \geq 1 - \alpha;$
2.  $\text{vol}(\hat{C}_{\text{DP}}) \leq \text{OPT}(P, 1 - \alpha + \gamma + 2\delta + 2\varepsilon_k).$

The result of [Theorem B.2.2](#) can also be conformalized as in [Section 3.2.4](#), so that the restricted volume optimality  $\text{OPT}_k(P, \cdot)$  in [Theorem 3.2.6](#) can be strengthened to  $\text{OPT}(P, \cdot)$  without restriction whenever  $P$  satisfies [Assumption B.2.1](#), which, in particular, includes the situation where the density of  $P$  can be well estimated by KDE.

The volume sub-optimality given by [Theorem B.2.2](#) is  $\gamma + 2\delta + 2\varepsilon_k$ . When the distribution  $P$  is  $\beta$ -smooth, the sub-optimality is of order  $\tilde{\Theta}\left(\sqrt{\frac{k}{n}} + k^{-\frac{\beta}{2\beta+1}}\right)$  by taking  $\varepsilon_k = \tilde{\Theta}\left(k^{-\frac{\beta}{2\beta+1}}\right)$ ,  $\delta = \tilde{\Theta}\left(\sqrt{\frac{k}{n}}\right)$ , and  $\gamma$  sufficiently small. Thus, optimizing this bound over  $k$  leads to the rate

$\tilde{\Theta}\left(n^{-\frac{\beta}{4\beta+1}}\right)$ . In comparison, the KDE achieves a faster rate  $\tilde{\Theta}\left(n^{-\frac{\beta}{2\beta+1}}\right)$  [Lei, Robins, and Wasserman, 2013] for smooth densities. This is actually a technical artifact by specializing Assumption B.2.1 to the KDE (B.1). In fact, when the density of  $P$  is  $\beta$ -smooth, it is well known that Assumption B.2.1 is satisfied with a better  $\varepsilon_k = \tilde{\Theta}(k^{-\beta})$  [Ghosal and van der Vaart, 2007, Kruijer, Rousseau, and van der Vaart, 2010], which then leads to the volume sub-optimality  $\tilde{\Theta}\left(\sqrt{\frac{k}{n}} + k^{-\beta}\right)$  that leads to the near optimal rate  $\tilde{\Theta}\left(n^{-\frac{\beta}{2\beta+1}}\right)$  with  $k = \tilde{\Theta}(n^{\frac{1}{2\beta+1}})$ .

We first state a lemma that shows that a level set of a Gaussian mixture with  $k$  components must belong to the class  $\mathcal{C}_k$ .

**Lemma B.2.3.** *For a Gaussian mixture  $P_k = \sum_{j=1}^k w_j N(\mu_j, \sigma_j^2)$  and any  $\alpha \in (0, 1)$ ,*

$$\text{OPT}_k(P_k, 1 - \alpha) = \text{OPT}(P_k, 1 - \alpha).$$

The proof of the lemma is in Section B.3.4. Now we are ready to state the proof of Theorem B.2.2.

*Proof of Theorem B.2.2.* By Proposition 3.2.4, we know that  $\hat{C}_{\text{DP}}$  satisfies  $\mathbb{P}_n(\hat{C}_{\text{DP}}) \geq 1 - \alpha + \delta$  and  $\text{vol}(\hat{C}_{\text{DP}}) \leq \text{OPT}_k(\mathbb{P}_n, 1 - \alpha + \delta + \gamma)$ . The condition on  $\delta$  implies that  $\sup_{C \in \mathcal{C}_k} |\mathbb{P}_n(C) - P(C)| \leq \delta$  with probability at least  $1 - \delta$  [Devroye and Lugosi, 2001]. Therefore, the coverage probability is

$$\mathbb{P}\left(Y_{n+1} \in \hat{C}_{\text{DP}} \mid Y_1, \dots, Y_n\right) \geq \mathbb{P}_n(\hat{C}_{\text{DP}}) - \delta \geq 1 - \alpha,$$

and the volume can be bounded by

$$\begin{aligned}
 \text{vol}(\widehat{C}_{\text{DP}}) &\leq \text{OPT}_k(\mathbb{P}_n, 1 - \alpha + \delta + \gamma) \\
 &\leq \text{OPT}_k(P, 1 - \alpha + 2\delta + \gamma) \\
 &\leq \text{OPT}_k(P_k, 1 - \alpha + 2\delta + \gamma + \varepsilon_k) \\
 &= \text{OPT}(P_k, 1 - \alpha + 2\delta + \gamma + \varepsilon_k) \\
 &\leq \text{OPT}(P, 1 - \alpha + 2\delta + \gamma + 2\varepsilon_k),
 \end{aligned}$$

where the identity above is by Lemma B.2.3. □

## B.3 Additional Proofs

### B.3.1 Proof of Theorem 3.2.2

The proof relies on the following technical lemma, whose proof will be given in Section B.3.4.

**Lemma B.3.1.** *For any  $\delta, \varepsilon > 0$  and any integer  $n > 0$ , there exists some distribution  $\Pi$  supported on*

$$\mathcal{P}_\varepsilon = \{P : \text{supp}(P) \subset [0, 1], \text{TV}(P, \lambda) \geq 1 - \varepsilon\},$$

*such that  $\text{TV}(\lambda^n, \int P^n d\Pi) \leq \delta$ .*

Now we are ready to state the proof of Theorem 3.2.2.

*Proof of Theorem 3.2.2.* By Lemma B.3.1, there exist  $\Pi_{n,\delta}$  and  $\Pi_{n+1,\delta}$  supported on  $\mathcal{P}_\delta$ , such that  $\text{TV}(\lambda^n, \int P^n d\Pi_{n,\delta}) \leq \delta$  and  $\text{TV}(\lambda^{n+1}, \int P^{n+1} d\Pi_{n+1,\delta}) \leq \delta$ .

Since  $P^{n+1}(Y_{n+1} \in \widehat{C}(Y_1, \dots, Y_n)) \geq 1 - \alpha$  for all  $P$ , we have

$$\int P^{n+1}(Y_{n+1} \in \widehat{C}(Y_1, \dots, Y_n)) d\Pi_{n+1, \delta} \geq 1 - \alpha.$$

By  $\text{TV}(\lambda^{n+1}, \int P^{n+1} d\Pi_{n+1, \delta}) \leq \delta$ , we have

$$\mathbb{E}_{Y_1, \dots, Y_n \sim \lambda^n}(\lambda(\widehat{C}(Y_1, \dots, Y_n))) = \lambda^{n+1}(Y_{n+1} \in \widehat{C}(Y_1, \dots, Y_n)) \geq 1 - \alpha - \delta.$$

By  $\text{TV}(\lambda^n, \int P^n d\Pi_{n, \delta}) \leq \delta$ , we have

$$\int \mathbb{E}_{Y_1, \dots, Y_n \sim P^n} \lambda(\widehat{C}(Y_1, \dots, Y_n)) d\Pi_{n, \delta} \geq 1 - \alpha - 2\delta.$$

Then, there must exists some  $P \in \text{supp}(\Pi_{n, \delta}) \subset \mathcal{P}_\delta$ , such that

$$\mathbb{E}_{Y_1, \dots, Y_n \sim P^n} \lambda(\widehat{C}(Y_1, \dots, Y_n)) \geq 1 - \alpha - 2\delta.$$

The fact that  $P \in \mathcal{P}_\delta$  implies  $\text{TV}(P, \lambda) \geq 1 - \delta$ . By the definition of total variation, there exists some set  $B$  such that  $P(B) - \lambda(B) \geq 1 - \delta$ , which implies  $P(B) \geq 1 - \delta$  and  $\lambda(B) \leq \delta$ . Therefore,  $\text{OPT}(P, 1 - \delta) \leq \delta$ .

We finally have for any  $\varepsilon \in (0, \alpha)$ , the expected volume of prediction set is

$$\begin{aligned} \mathbb{E}_{Y_1, \dots, Y_n \sim P^n} \text{vol}(\widehat{C}(Y_1, \dots, Y_n)) &\geq 1 - \alpha - 2\delta \\ &\geq \delta \\ &\geq \text{OPT}(P, 1 - \delta) \\ &\geq \text{OPT}(P, 1 - \alpha + \varepsilon), \end{aligned}$$

as long as  $\delta$  is sufficiently small so that  $\delta < \min\{(1-\alpha)/3, \alpha-\varepsilon\}$ . The proof is complete.  $\square$

### B.3.2 Proof of Theorem 3.2.6

Theorem 3.2.6 is a special case of Theorem 3.3.3 in the setting where  $F(\cdot \mid x)$  does not depend on  $x$  and  $\hat{F}(\cdot \mid x)$  is defined as the empirical CDF of  $Y_1, \dots, Y_n$ . Then, Assumption 3.3.2 is automatically satisfied by a standard VC dimension bound [Devroye and Lugosi, 2001].

### B.3.3 Proof of Theorem 3.3.3

We will prove the three properties of Theorem 3.3.3 separately. We note that the marginal coverage property holds without Assumption 3.3.1 and Assumption 3.3.2. It is a standard consequence of applying the split conformal framework, but we still include a proof here for completeness.

*Proof of Theorem 3.3.3 (marginal coverage).* By the construction of  $\hat{C}_{\text{DCP-DP}}(X_{2n+1})$ , we have

$$\mathbb{P}\left(Y_{2n+1} \in \hat{C}_{\text{DCP-DP}}(X_{2n+1})\right) = \mathbb{P}\left(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}\right).$$

Since the conformity score  $q$  is constructed from  $\hat{F}(\cdot \mid \cdot)$ , it is independent from the second half of the data. Thus,  $q(Y_{2n+1}, X_{2n+1})$  is exchangeable with  $q(Y_{n+1}, X_{n+1}), \dots, q(Y_{2n}, X_{2n})$ , which implies the desired conclusion by the definition of  $q_{\lfloor (n+1)\alpha \rfloor}$ .  $\square$

Next, we establish the conditional coverage property. We need the following property of the conformity score that is computed based on a nested system.

**Lemma B.3.2.** *For any  $j \in [m+1]$ ,  $y \in S_j(x)$  if and only if  $q(y, x) \geq m-j+1$ , where the set  $S_{m+1}(x)$  is defined as  $\mathbb{R}$ .*

The proof of the lemma will be given in [Section B.3.4](#).

*Proof of [Theorem 3.3.3](#) (approximate conditional coverage).* We first note that [Assumption 3.3.2](#) implies

$$\mathbb{E} \|\widehat{F}(\cdot | X_{2n+1}) - F(\cdot | X_{2n+1})\|_{k,\infty} \leq 2\delta. \quad (\text{B.2})$$

We use  $\mathcal{F}_{2n}$  to denote the  $\sigma$ -field generated by the random variables  $(X_1, Y_1), \dots, (X_{2n}, Y_{2n})$ . Let  $\mathbb{E}_{X_{2n+1}}$  and  $\mathbb{E}_{\mathcal{F}_{2n}}$  be the expectation operators under the marginal distributions of  $X_{2n+1}$ , and of  $(X_1, Y_1), \dots, (X_{2n}, Y_{2n})$ , respectively. Then, we have

$$\mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}) = \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor} | X_{2n+1}, \mathcal{F}_{2n}).$$

By [Lemma B.3.2](#),  $q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}$  is equivalent to  $Y_{2n+1} \in S_{\widehat{j}}(X_{2n+1})$  for some  $\widehat{j}$  measurable with respect to  $\mathcal{F}_{2n}$ . Then, we have

$$\begin{aligned} & \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor} | X_{2n+1}, \mathcal{F}_{2n}) \\ &= \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \mathbb{P}(Y_{2n+1} \in S_{\widehat{j}}(X_{2n+1}) | X_{2n+1}, \mathcal{F}_{2n}) \\ &= \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \int_{S_{\widehat{j}}(X_{2n+1})} dF(y | X_{2n+1}). \end{aligned}$$

Since  $S_{\widehat{j}}(X_{2n+1}) \in \mathcal{C}_k$ , by [\(B.2\)](#), we have

$$\begin{aligned} \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \int_{S_{\widehat{j}}(X_{2n+1})} dF(y | X_{2n+1}) &\leq \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \int_{S_{\widehat{j}}(X_{2n+1})} d\widehat{F}(y | X_{2n+1}) \\ &\quad + \mathbb{E} \|\widehat{F}(\cdot | X_{2n+1}) - F(\cdot | X_{2n+1})\|_{k,\infty} \\ &\leq \mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \int_{S_{\widehat{j}}(X_{2n+1})} d\widehat{F}(y | X_{2n+1}) + 2\delta. \end{aligned}$$



By [Assumption 3.3.1](#), we have  $\int_{S_{\hat{j}}(X_{2n+1})} d\hat{F}(y \mid X_{2n+1}) = \frac{\hat{j}}{m}$ , which is independent of  $X_{2n+1}$ , since  $\hat{j}$  measurable with respect to  $\mathcal{F}_{2n}$ . Thus, we have

$$\mathbb{E}_{\mathcal{F}_{2n}} \mathbb{E}_{X_{2n+1}} \int_{S_{\hat{j}}(X_{2n+1})} d\hat{F}(y \mid X_{2n+1}) + 2\delta = \mathbb{E}_{\mathcal{F}_{2n}} \int_{S_{\hat{j}}(X_{2n+1})} d\hat{F}(y \mid X_{2n+1}) + 2\delta.$$

By [Assumption 3.3.2](#), we have with probability at least  $1 - \delta$ ,

$$\begin{aligned} \mathbb{E}_{\mathcal{F}_{2n}} \int_{S_{\hat{j}}(X_{2n+1})} d\hat{F}(y \mid X_{2n+1}) + 2\delta &\leq \mathbb{E}_{\mathcal{F}_{2n}} \int_{S_{\hat{j}}(X_{2n+1})} dF(y \mid X_{2n+1}) + 3\delta \\ &= \mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor} \mid X_{2n+1}) + 3\delta. \end{aligned}$$

Therefore, with probability at least  $1 - \delta$ , the approximate conditional coverage holds

$$\mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor} \mid X_{2n+1}) \geq \mathbb{P}(q(Y_{2n+1}, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}) - 3\delta \geq 1 - \alpha - 3\delta.$$

□

Finally, we prove the last property on volume optimality.

*Proof of [Theorem 3.3.3](#) (conditional restricted volume optimality).* By [Assumption 3.3.1](#), there exists some  $j^* \in [m]$ , such that

$$\mathbb{E} \int_{S_{j^*}(X_{2n+1})} d\hat{F}(y \mid X_{2n+1}) \geq 1 - \alpha + \frac{1}{n} + 3\delta.$$

[Assumption 3.3.2](#) implies that

$$\begin{aligned}
 \mathbb{P}(Y_{2n+1} \in S_{j^*}(X_{2n+1})) &= \mathbb{E} \int_{S_{j^*}(X_{2n+1})} dF(y \mid X_{2n+1}) \\
 &\geq \mathbb{E} \int_{S_{j^*}(X_{2n+1})} d\widehat{F}(y \mid X_{2n+1}) \\
 &\quad - \mathbb{E} \|\widehat{F}(\cdot \mid X_{2n+1}) - F(\cdot \mid X_{2n+1})\|_{k,\infty} \\
 &\geq 1 - \alpha + \frac{1}{n} + \delta.
 \end{aligned}$$

By Hoeffding's inequality and the condition on  $\delta$ , we have

$$\frac{1}{n} \sum_{i=n+1}^{2n} \mathbb{I}\{Y_i \in S_{j^*}(X_i)\} \geq \mathbb{P}(Y_{2n+1} \in S_{j^*}(X_{2n+1})) - \delta,$$

with probability at least  $1 - \delta$ . Combining the two inequalities above and [Lemma B.3.2](#), we get

$$\frac{1}{n} \sum_{i=n+1}^{2n} \mathbb{I}\{q(Y_i, X_i) \geq m - j^* + 1\} \geq 1 - \alpha + \frac{1}{n},$$

with probability at least  $1 - \delta$ . This immediately implies  $q_{\lfloor n\alpha \rfloor} = q_{\lfloor n(\alpha - n^{-1}) + 1 \rfloor} \geq m - j^* + 1$  by the definition of order statistics. Therefore, the volume of the prediction set  $\widehat{C}_{\text{DCP-DP}}(X_{2n+1})$  is at most

$$\begin{aligned}
 &\text{vol}(\{y \in \mathbb{R} : q(y, X_{2n+1}) \geq q_{\lfloor (n+1)\alpha \rfloor}\}) \\
 &\leq \text{vol}(\{y \in \mathbb{R} : q(y, X_{2n+1}) \geq q_{\lfloor n\alpha \rfloor}\}) \\
 &\leq \text{vol}(\{y \in \mathbb{R} : q(y, X_{2n+1}) \geq m - j^* + 1\}) \\
 &= \text{vol}(S_{j^*}(X_{2n+1})),
 \end{aligned}$$

where the last identity is by [Lemma B.3.2](#). The volume of  $S_{j^*}(X_{2n+1})$  can be controlled by

Assumption 3.3.1,

$$\begin{aligned} \text{vol}(S_{j^*}(X_{2n+1})) &\leq \text{OPT}_k \left( \widehat{F}(\cdot \mid X_{2n+1}), 1 - \alpha + \frac{1}{n} + 3\delta + \gamma \right) \\ &\leq \text{OPT}_k \left( F(\cdot \mid X_{2n+1}), 1 - \alpha + \frac{1}{n} + 4\delta + \gamma \right), \end{aligned}$$

where the last inequality, which holds with probability at least  $1 - \delta$ , is by Assumption 3.3.2.

Combining the inequalities above with union bound, we get the conclusion.  $\square$

### B.3.4 Proofs of Proposition 3.2.4, Lemma B.2.3, Lemma B.3.1 and Lemma B.3.2

**Dynamic Programming Algorithm.** The dynamic programming table  $\text{DP}(i, j, l)$  stores the minimum volume of  $i$  intervals that collectively cover  $l\gamma n$  points from the sorted training data  $Y_{(1)}, \dots, Y_{(j)}$ , where the right endpoint of the rightmost interval is fixed at  $Y_{(j)}$ . Here,  $Y_{(1)}, \dots, Y_{(n)}$  are the training data points  $Y_1, \dots, Y_n$  sorted in non-decreasing order. For each state in the DP table, we iterate over all possible left endpoints of the rightmost interval, as well as the right endpoint of the preceding interval (if it exists). This allows us to systematically compute the optimal solution for each state by the following formula:

$$\begin{aligned} \text{If } i = 1, \quad &\text{DP}(i, j, l) = Y_{(j)} - Y_{(j - \lfloor l\gamma n \rfloor + 1)}, \\ \text{If } i > 1, \quad &\text{DP}(i, j, l) = \min_{i-1 \leq j'' < j' \leq j} \{Y_{(j)} - Y_{(j')} + \text{DP}(i-1, j'', l')\}, \end{aligned}$$

where  $l' = l - \lfloor (j - j' + 1)/(\gamma n) \rfloor$ . Finally, we find the minimum volume solution among all entries  $\text{DP}(k, j, \lceil (1 - \alpha)/\gamma \rceil)$  for all  $1 \leq j \leq n$ . Then, we use the standard backtrack approach on the DP table to find the prediction set  $\widehat{C}_{\text{DP}}$ .

*Proof of Proposition 3.2.4.* Without loss of generality, we assume that  $1/\gamma$  is an integer, otherwise, we can decrease  $\gamma$  to make this hold. For any  $i \in [k], j \in [n], l \in [1/\gamma]$ , we use the dynamic programming table entry  $\text{DP}(i, j, l)$  to store the minimum volume of  $i$  intervals that cover  $\lceil l \cdot \gamma n \rceil$  points in  $Y_{(1)}, \dots, Y_{(j)}$  and the right endpoint of the rightmost interval is at  $Y_{(j)}$ . If there is no feasible solution for this subproblem, we set  $\text{DP}(i, j, l) = \infty$ . This dynamic programming is shown in Algorithm 3.1. This dynamic programming runs in time  $O(n^3 k / \gamma)$ .

We then find the solution with the minimum volume among all subproblems  $\text{DP}(k, j, \lceil (1 - \alpha)/\gamma \rceil)$  for  $1 \leq j \leq n$ . It is easy to see that there exists a feasible solution. Let  $\hat{C}_{\text{DP}} \in \mathcal{C}_k$  be a union of  $k$  intervals in this solution. This solution covers at least  $\lceil (1 - \alpha)/\gamma \cdot (n\gamma) \rceil = \lceil (1 - \alpha)n \rceil$  points in  $X_1, \dots, X_n$ . Thus, we have

$$\mathbb{P}_n(\hat{C}_{\text{DP}}) \geq 1 - \alpha.$$

If the restricted optimal volume  $\text{OPT}_k(\mathbb{P}_n, ((1 - \alpha)/\gamma + 1) \cdot (n\gamma)/n)$  is smaller than the volume of  $\hat{C}_{\text{DP}}$ , then this solution cannot have the minimum volume among all subproblems  $\text{DP}(k, j, \lceil (1 - \alpha)/\gamma \rceil)$  for  $1 \leq j \leq n$ . Thus, the volume of this solution must satisfy

$$\begin{aligned} \text{vol}(\hat{C}_{\text{DP}}) &\leq \text{OPT}_k(\mathbb{P}_n, ((1 - \alpha)/\gamma + 1) \cdot (n\gamma)/n) \\ &= \text{OPT}_k(\mathbb{P}_n, 1 - \alpha + \gamma). \end{aligned}$$

The proof is thus complete. □

*Proof of Lemma B.2.3.* By [Carreira-Perpinán and Williams, 2003], there are  $k' \leq k$  local maxima for the density function  $p_k$ . We will use  $k'$  intervals and define the rest of the intervals to be empty. Suppose  $u_1 \leq u_2 \leq \dots \leq u_{k'} \in \mathbb{R}$  are the local maxima of  $p_k$ . The

density  $p_k(y)$  is differentiable, and its local minima and local maxima have to alternate. Hence there are exactly  $k' - 1$  local minima, denoted by  $\ell_1, \ell_2, \dots, \ell_{k'-1}$  with  $\ell_j \in [u_j, u_{j+1}]$  for all  $j \in \{1, 2, \dots, k' - 1\}$ . (Note that there are no local minima less than  $u_1$  or greater than  $u_{k'}$  since  $p_k(y) \rightarrow 0$  as  $y \rightarrow \pm\infty$ ). For notational convenience let  $\ell_0 = -\infty, \ell_{k'} = \infty$ . Let  $C^* \subset \mathbb{R}$  satisfies  $P_k(C^*) \geq 1 - \alpha$  and  $\text{vol}(C^*) = \text{OPT}(P_k, 1 - \alpha)$ .

We now show that there exist  $I_1, \dots, I_{k'} \in \mathcal{C}_1$  such that  $u_j \in I_j \subset [\ell_{j-1}, \ell_j]$  for all  $j \in [k']$ ,  $P_k(\cup_{j=1}^{k'} I_j) \geq 1 - \alpha$ , and  $\text{vol}(\cup_{j=1}^{k'} I_j) \leq \text{vol}(C^*)$ . This would imply the desired conclusion. Consider  $S_j = S^* \cap [\ell_{j-1}, \ell_j]$  for all  $j \in [k']$ . Next we observe for all  $j \in [k']$ ,  $p_k$  is monotonically increasing in the interval  $[\ell_{j-1}, u_j]$  and is monotonically decreasing in the intervals  $[u_j, \ell_j]$ , with a local maximum at  $u_j$ . Hence if  $S_j$  comprises multiple disjoint intervals within  $[\ell_{j-1}, \ell_j]$ , we can pick one interval with the same volume within  $[\ell_{j-1}, \ell_j]$  that also includes  $u_j$  and covers at least as much probability mass. This establishes the property of  $\cup_{j=1}^{k'} I_j$ , and hence the lemma.  $\square$

*Proof of Lemma B.3.1.* First, we construct a family of distributions supported on subsets of  $[0, 1]$ . Consider an integer  $m$ . We partition the interval  $[0, 1]$  into  $m$  intervals with length  $1/m$  each and define the subinterval  $A_j = [\frac{j-1}{m}, \frac{j}{m})$  for  $j \in [m]$ . We next define a family of distributions supported on these subintervals. For any vector  $Z \in \{0, 1\}^m$ , let  $A_Z = \bigcup_{j: Z_j=1} A_j$  denote the union of intervals corresponding to the indices where  $Z_j = 1$ . Then, we define the density function

$$p_Z(y) = \frac{\mathbb{I}\{y \in A_Z\}}{\frac{1}{m} \sum_{j=1}^m Z_j},$$

where  $\mathbb{I}$  is the indicator function. Let  $P_Z$  be the corresponding distribution.

We then construct the weight distribution  $\Pi$ . Given any  $\varepsilon > 0$ , we now provide a restricted

set of vectors  $Z \in \{0, 1\}^m$  such that the distribution  $P_Z$  has at least  $1 - \varepsilon$  total variation distance to the uniform distribution  $\lambda$ . We pick a parameter  $\beta \in (0, 1)$  depending on  $\varepsilon$  and  $m$ . Then, we define a set of vectors  $Z$  with  $A_Z$  covering approximately  $\beta$  fraction of  $[0, 1]$ ,

$$\mathcal{Z} = \left\{ Z \in \{0, 1\}^m : \left| \frac{1}{m} \sum_{j=1}^m Z_j - \beta \right| \leq \left( \frac{\beta}{m} \right)^{1/3} \right\}.$$

For any  $Z \in \mathcal{Z}$ , we have the total variation distance

$$\text{TV}(P_Z, \lambda) \geq \lambda(A_Z^c) = 1 - \frac{1}{m} \sum_{j=1}^m Z_j \geq 1 - \beta - \left( \frac{\beta}{m} \right)^{1/3}.$$

Therefore, as long as  $\beta + \left( \frac{\beta}{m} \right)^{1/3} \leq \varepsilon$ , we have  $P_Z \in \mathcal{P}_\varepsilon$  for all  $Z \in \mathcal{Z}$ . We construct the weight distribution  $\Pi$  supported on the  $\{P_Z : Z \in \mathcal{Z}\}$ . Let  $\tilde{\Pi} = \otimes_{j=1}^m \text{Bernoulli}(\beta)$  be the product distribution on  $\{0, 1\}^m$  such that each coordinate is 1 with probability  $\beta$ . Then, we define  $\Pi$  to be the distribution  $\tilde{\Pi}$  conditioning on  $\mathcal{Z}$ ,  $\Pi(Z) = \frac{\tilde{\Pi}(Z \cap \mathcal{Z})}{\tilde{\Pi}(\mathcal{Z})}$ .

We bound  $\text{TV}(\lambda^n, \int P_Z^n d\Pi(Z))$  by chi-squared divergence,

$$\frac{1}{2} \text{TV} \left( \lambda^n, \int P_Z^n d\Pi(Z) \right)^2 \leq \int_{[0,1]^n} \left( \int_Z p_Z^n d\Pi(Z) \right)^2 dx - 1.$$

Since  $\int P_Z^n d\Pi(Z)$  is a mixture of product distributions over  $\{P_Z^n : Z \in \mathcal{Z}\}$  with weights  $\Pi$ , we can expand the first term in the right-hand side as

$$\int_{[0,1]^n} \left( \int_Z p_Z^n d\Pi(Z) \right)^2 dx = \mathbb{E}_{Z, Z' \stackrel{iid}{\sim} \Pi} \left( \int p_Z(y) p_{Z'}(y) dy \right)^n.$$

By taking the density  $p_Z(y) = \frac{\mathbb{I}\{y \in \mathcal{Z}\}}{\frac{1}{m} \sum_{j=1}^m Z_j}$  into the equation, we have

$$\mathbb{E}_{Z, Z' \sim \Pi} \left( \int p_Z(y) p_{Z'}(y) dx \right)^n = \mathbb{E}_{Z, Z' \sim \Pi} \left( \frac{1}{\frac{1}{m} \sum_{j=1}^m Z_j} \cdot \frac{1}{\frac{1}{m} \sum_{j=1}^m Z'_j} \cdot \frac{1}{m} \sum_{j=1}^m Z_j Z'_j \right)^n.$$

According to the construction of  $\mathcal{Z}$ , for any vector  $Z \in \mathcal{Z}$ , we have  $\frac{1}{m} \sum_{j=1}^m Z_j \geq \beta - \left(\frac{\beta}{m}\right)^{1/3}$ . Since  $\tilde{\Pi}$  is the product of Bernoulli distribution with probability  $\beta$ , by the Chernoff bound, we have  $\tilde{\Pi}(Z \in \mathcal{Z}) \geq 1 - (\beta/m)^{1/3}$ . Thus, we have

$$\begin{aligned} & \mathbb{E}_{Z, Z' \sim \Pi} \left( \int p_Z(y) p_{Z'}(y) dx \right)^n \\ & \leq \left( \beta - \left(\frac{\beta}{m}\right)^{1/3} \right)^{-2n} \mathbb{E}_{Z, Z' \sim \Pi} \left( \frac{1}{m} \sum_{j=1}^m Z_j Z'_j \right)^n \\ & = \left( \beta - \left(\frac{\beta}{m}\right)^{1/3} \right)^{-2n} \frac{\mathbb{E}_{Z, Z' \sim \tilde{\Pi}} \left( \frac{1}{m} \sum_{j=1}^m Z_j Z'_j \right)^n \mathbb{I}\{Z \in \mathcal{Z}\} \mathbb{I}\{Z' \in \mathcal{Z}\}}{\tilde{\Pi}(Z \in \mathcal{Z})^2} \\ & \leq \left( \beta - \left(\frac{\beta}{m}\right)^{1/3} \right)^{-2n} \left( 1 - \left(\frac{\beta}{m}\right)^{1/3} \right)^{-2} \mathbb{E}_{Z, Z' \sim \tilde{\Pi}} \left( \frac{1}{m} \sum_{j=1}^m Z_j Z'_j \right)^n. \end{aligned}$$

The last term on the right-hand side can be bounded by

$$\begin{aligned} \mathbb{E}_{Z, Z' \sim \tilde{\Pi}} \left( \frac{1}{m} \sum_{j=1}^m Z_j Z'_j \right)^n & \leq \left( \beta^2 + \left(\frac{\beta^2}{m}\right)^{1/3} \right)^n + \tilde{\Pi} \left( \frac{1}{m} \sum_{j=1}^m Z_j Z'_j > \beta^2 + \left(\frac{\beta^2}{m}\right)^{1/3} \right) \\ & \leq \left( \beta^2 + \left(\frac{\beta^2}{m}\right)^{1/3} \right)^n + \left(\frac{\beta^2}{m}\right)^{1/3}, \end{aligned}$$

where the first inequality is using  $\frac{1}{m} \sum_{j=1}^m Z_j Z'_j \leq 1$  for  $\frac{1}{m} \sum_{j=1}^m Z_j Z'_j > \beta^2 + \left(\frac{\beta^2}{m}\right)^{1/3}$  and the second inequality is from the Chernoff bound on the sum of  $m$  Bernoulli variables with

probability  $\beta^2$ . Combining all bounds above, we have

$$\frac{1}{2} \text{TV} \left( \lambda^n, \int P_Z^n d\Pi(Z) \right)^2 \leq \frac{\left( \beta^2 + \left( \frac{\beta^2}{m} \right)^{1/3} \right)^n + \left( \frac{\beta^2}{m} \right)^{1/3}}{\left( \beta - \left( \frac{\beta}{m} \right)^{1/3} \right)^{2n} \left( 1 - \left( \frac{\beta}{m} \right)^{1/3} \right)^2} - 1.$$

It is clear that the above bound tends to zero when  $m \rightarrow \infty$ . Therefore, for any  $\delta > 0$ , we have  $\text{TV} \left( \lambda^n, \int P_Z^n d\Pi(Z) \right) \leq \delta$  for a sufficiently large  $m$ .  $\square$

*Proof of [Lemma B.3.2](#).* The result is a direct consequence of the nested property of  $\{S_j(x)\}_{j \in [m]}$ .  $\square$



# Appendix C

## Details for High-dimensional Ellipsoids

### C.1 Computational Hardness of Obtaining Non-trivial Volume Approximation Guarantees

In this section, we show evidence of computational intractability for obtaining any non-trivial approximations for the volume of the ellipsoid. In we will show that it is computationally hard to determine whether there exists a  $1 - \alpha$ -fraction of points that lie on a subspace. Note that the volume of the minimum volume ellipsoid containing a set of points  $S$  is zero exactly when  $S$  belongs to a subspace of dimension at most  $d - 1$ .

We start with a simple NP-hardness result due to [Khachiyan \[1995\]](#).

**Proposition C.1.1.** *There exists a constant  $\alpha_0 > 0$ , such that for any  $\alpha \geq \alpha_0(1 - \frac{1}{d}) + \frac{1}{d}$  given any set of  $m$  points in  $d$  dimensions, assuming  $P \neq NP$ , there is no algorithm running*

in polynomial time (in  $m, d$ ) that determines if there exists an ellipsoid of volume 0 that contains at least  $(1 - \alpha)m$  points.

*Proof.* Khachiyan shows that it is NP-hard to determine if there exists a  $d - 1$  dimensional subspace containing at least  $(1 - \alpha_0)(1 - 1/d)$  fraction of points [Khachiyan, 1995]. The volume of the minimum volume ellipsoid containing a set of points  $S$  is zero if and only if  $S$  belongs to a subspace of dimension at most  $d - 1$ . Moreover, we can add additional random points in  $\mathbb{R}^d$  to get the same hardness for any coverage  $1 - \alpha \leq (1 - \alpha_0)(1 - 1/d)$ .  $\square$

## C.2 Proofs of Claims in Section 5.4

*Proof of Claim 5.4.2.* Let  $S := N_B(F) \subseteq V$  be the set of vertices incident to edges in  $F$ , and let  $H := (S, F)$  be the subgraph of  $G$  induced by  $F$  on the vertex set  $S$ .

We first prove the second inequality. Since for any edge  $e = (i, j) \in F$ , the vector  $u'_e = (e_i + e_j)/\sqrt{2}$ , all vectors in  $P$  lie in the coordinate subspace spanned by  $e_i$  for all  $i \in S$ . Therefore

$$\dim(\text{span } P) \leq |S| = |N_B(F)|.$$

We now prove the first inequality. Let the connected components of  $H$  be  $H_1, \dots, H_r$  with vertex sets  $S_1, \dots, S_r$  (a partition of  $S$ ) and edge sets  $F_1, \dots, F_r$  (a partition of  $F$ ). Since the supports of the vectors  $\{u'_e : e \in F_c\}$  lie in disjoint coordinate blocks  $S_c$ , we have

$$\dim(\text{span } P) = \sum_{c=1}^r \dim(\text{span } \{u_e : e \in F_c\}).$$

Fix a component  $H_c$ . Choose a spanning tree  $T_c \subseteq F_c$ ; then  $|T_c| = |S_c| - 1$ . We claim that the vectors  $\{u'_e : e \in T_c\}$  are linearly independent. Indeed, suppose  $\sum_{e \in T_c} \lambda_e u'_e = 0$ . Pick a

leaf  $v$  of the tree  $T_c$ , and let  $e_v = \{v, w\}$  be its unique incident edge in  $T_c$ . Looking at the  $v$ -th coordinate of the preceding relation yields  $\lambda_{e_v} = 0$ , because  $u'_{e_v}$  has one in coordinate  $v$  while all other  $u'_e$  for  $e \in T_c \setminus \{e_v\}$  have zero in that coordinate. Removing  $e_v$  and  $v$  from the tree and repeating this argument inductively forces all coefficients to be zero. Thus

$$\dim(\text{span} \{u_e : e \in F_c\}) = |T_c| = |S_c| - 1.$$

Summing over components gives

$$\dim(\text{span } P) = \sum_{c=1}^r (|S_c| - 1) = |S| - r.$$

Finally, since  $S$  consists of vertices incident to edges of  $F$ , each component  $H_c$  has at least two vertices, so  $r \leq |S|/2$ . Therefore

$$\dim(\text{span } P) \geq |S| - r \geq \frac{|S|}{2} = \frac{|N_B(F)|}{2}.$$

Combining with the upper bound proves the claim.  $\square$

## C.3 Algorithm Details

### C.3.1 Preliminaries

The Schur complement is a useful way to compute the determinant and eigenvalues of a block matrix.

**Fact C.3.1** (Schur complement [Bhatia, 1997]). *Consider a  $(p + q) \times (p + q)$  block matrix*

$M$  where the submatrix  $X \in \mathbb{R}^{p \times p}$  is invertible.

$$M = \left[ \begin{array}{c|c} X & Y \\ \hline W & Z \end{array} \right]$$

Then the Schur complement for the block  $X$  is given by  $Z - WX^{-1}Y$ , and

$$\det(M) = \det(X) \cdot \det(Z - WX^{-1}Y).$$

### C.3.2 Bounding Ball

The first step of our algorithm is to restrict our attention to a bounding ball that has radius proportional to the longest axis of the optimal ellipsoid. This can be done via a simple search procedure.

**Lemma C.3.2** (Bounding ball). *For the set of points  $A = \{a_1, \dots, a_n\} \subseteq \mathbb{R}^d$ , create a set of balls*

$$\mathfrak{B} = \{B(a_i, \|a_i - a_j\|) : i, j \in [n]\}.$$

*Remove any  $B \in \mathfrak{B}$  such that  $|B \cap A| \leq (1 - \alpha)n$ . We have that  $|\mathfrak{B}| \leq n^2$ .*

*Let  $E^*$  be the minimum volume ellipsoid of condition number at most  $\beta$  that contains at least  $(1 - \alpha)$ -fraction of the points. Then, there exists a ball  $\hat{B} \in \mathfrak{B}$  such that  $\hat{B}$  contains all of the points in  $E^*$ , and the radius of  $\hat{B}$  is at most twice the longest axis length of  $E^*$ .*

*Proof.* The bound  $|\mathfrak{B}| \leq n^2$  is immediate from the construction since there are at most  $n^2$  pairs of  $i, j \in [n]$ .

Let  $c$  be the center of the ellipsoid  $E^*$ , and let  $R = \max_{x \in E^*} \|x - c\|$  be the length of its

longest semi-axis. Let  $a_i, a_j \in A \cap E^\star$  be the pair of centers that maximize the distance among all pairs of points in  $A \cap E^\star$ , i.e.,

$$\|a_i - a_j\| = \max\{\|a - a'\| : a, a' \in A \cap E^\star\}.$$

Set the ball  $\widehat{B} := B(a_i, \|a_i - a_j\|)$ .

By construction, for every  $a_\ell \in A \cap E^\star$  we have  $\|a_\ell - a_i\| \leq \|a_j - a_i\|$ , so  $A \cap E^\star \subseteq \widehat{B}$ ; that is,  $\widehat{B}$  contains all points of  $A$  that lie in  $E^\star$ . Thus, we have the ball  $\widehat{B} \in \mathfrak{B}$ . Moreover, since  $a_i, a_j \in E^\star$ ,

$$\|a_i - a_j\| \leq \|a_i - c\| + \|a_j - c\| \leq R + R = 2R,$$

so the radius of  $\widehat{B}$  is at most  $2R$ , i.e., at most twice the longest semi-axis length of  $E^\star$ . This proves the claim.  $\square$

### C.3.3 Iterative Outlier Removal via Dual

We analyze the number of points that are removed in our iterative outlier removal process. We use Chebyshev's inequality with the observation that our removal process forms a martingale.

**Proposition C.3.3** (Point removal martingale). *Let the random variable  $w_{(nj)+i}$  be the weight that is assigned by the SDP to point  $i$  in iteration  $j \in [R]$  ( $j$ th time the SDP is solved) in [Algorithm 5.1](#), where we define  $w_{(nj)+i} = 0$  for points that have already been deleted by iteration  $j$ .*

*Let  $X_{(nj)+i} \sim \text{Bern}(w_{(nj)+i})$  be the indicator random variable of whether point  $i$  was removed*

in iteration  $j$ . Define  $Z_{(nj)+i}$  as the deviation of  $X_{(nj)+i}$  from its expectation,

$$Z_{(nj)+i} = X_{(nj)+i} - w_{(nj)+i}.$$

We have that

$$\mathbb{E}[Z_k | Z_0, \dots, Z_{k-1}] = 0,$$

$$\mathbb{E}[Z_k^2] = \mathbb{E}[w_k(1 - w_k)] \leq \mathbb{E}[w_k].$$

Thus, for any subset of the variables  $S \subseteq [nR]$ , we have

$$\begin{aligned} \text{Var} \left[ \sum_{k \in S} Z_k \right] &= \mathbb{E} \left[ \left( \sum_{k \in S} Z_k \right)^2 \right] = \sum_{k \in S} \sum_{k' \in S} \mathbb{E}[Z_k Z_{k'}] \\ &= \sum_{k \in S} \mathbb{E}[Z_k^2] \leq \mathbb{E} \left[ \sum_{k \in S} w_k \right] = \mathbb{E} \left[ \sum_{k \in S} X_k \right]. \end{aligned}$$

This allows us to conclude the two statements we need about the outlier removal process.

First we argue that over  $R$  rounds, we will make progress removing the outliers.

**Lemma C.3.4** (Outlier removal progress). *Assume  $\alpha n$  is at least a sufficiently large constant. There exists a constant  $c$  such that if we run the iterative outlier removal procedure for  $R \geq c \frac{\alpha n}{\gamma d}$  iterations, then with probability at least 0.99, there exists an iteration on which less than  $\gamma d$  mass was placed on outliers.*

*Proof.* Consider the random variables as defined in [Proposition C.3.3](#). Let  $S \subseteq [nR]$  be the subset of indices that correspond to outliers,

$$S = \{(nj) + i : a_i \notin E^*\},$$

where  $E^*$  is the optimal solution.

Let

$$Z_S = \sum_{k \in S} Z_k$$

be the deviation of the number of outliers removed from its expectation.

By [Proposition C.3.3](#) we have

$$\text{Var}(Z_S) \leq \mathbb{E} \left[ \sum_{k \in S} X_k \right] \leq \alpha n,$$

since there are at most  $\alpha n$  outliers that can be removed in any run of the algorithm. By Chebyshev's inequality we have

$$\mathbb{P}[|Z_S| \geq (c-1)\alpha n] \leq \frac{\alpha n}{(c-1)^2(\alpha n)^2} = \frac{1}{(c-1)^2\alpha n}.$$

Let  $\mathcal{E}$  be the event that on all  $R$  iterations, the SDP placed at least  $\gamma d$  mass on outliers. We have that in the event  $\mathcal{E}$ ,

$$\begin{aligned} \sum_{k \in S} Z_k &= \sum_{k \in S} X_k - \sum_{k \in S} w_k \\ &\leq \alpha n - R \cdot \gamma d \\ &\leq -(c-1)\alpha n. \end{aligned}$$

Thus  $\mathcal{E}$  is a subset of the event that  $|Z_S| \geq (c-1)\alpha n$ , and the probability of event  $\mathcal{E}$  is bounded by  $\frac{1}{(c-1)^2\alpha n}$ . Taking  $c = 2$  and  $\alpha n \geq 100$  proves the claim.  $\square$

We also must show that over  $R$  rounds we do not remove too many points total (outliers

and inliers together).

**Lemma C.3.5** (Bound on number of points removed). *Assume  $\frac{\alpha n}{\gamma}$  is at least a sufficiently large constant. Let  $c \in \mathbb{R}$  be a constant. Then there exists a constant  $c_2$  such that, if we run the iterative outlier removal procedure for  $R = c \frac{\alpha n}{\gamma d}$  iterations, then with probability at least 0.99, we remove a total of at most*

$$c_2 \frac{\alpha n}{\gamma} \text{ points.}$$

*Proof.* Let

$$Z = \sum_{k \in [nR]} Z_k$$

be the deviation of the number of points removed from its expectation. By [Proposition C.3.3](#) we have

$$\text{Var}(Z) \leq \mathbb{E} \left[ \sum_{k \in [nR]} X_k \right] \leq Rd,$$

since the SDP assigns weight  $d$  in each round. By Chebyshev's inequality, we have

$$\mathbb{P}[|Z| \geq c_3 Rd] \leq \frac{Rd}{c_3^2 R^2 d^2} = \frac{1}{c_3^2 Rd}.$$

Thus,

$$\mathbb{P} \left[ \sum_{k \in [nR]} X_k \geq c_2 \frac{\alpha n}{\gamma} \right] \leq \mathbb{P} \left[ \sum_{k \in [nR]} Z_k \geq \left( \frac{c_2}{c} - 1 \right) Rd \right] \leq \frac{1}{(c_2/c - 1)^2 Rd}.$$

Taking  $c_2$  such that  $c_2/c - 1 \geq 1$ , and assuming  $Rd = \frac{\alpha n}{\gamma} \geq 100$  proves the claim.  $\square$



### C.3.4 Dual Equivalence

**Proposition C.3.6** (Dual equivalence under the lift). *Let  $a_1, \dots, a_n \in \mathbb{R}^d$  and  $\tilde{a}_i := (a_i, 1) \in \mathbb{R}^{d+1}$ . Let  $w$  be any optimal dual solution for the free-center MVEE (Proposition 5.2.2) in  $\mathbb{R}^d$ . Then  $\tilde{w} = \frac{d+1}{d}w$  is an optimal dual solution for the origin-centered MVEE (Proposition 5.2.4) of the lifted points  $(\tilde{a}_i)$  in  $\mathbb{R}^{d+1}$ .*

*Conversely, let  $\tilde{w}$  be any optimal dual solution for the origin-centered MVEE (Proposition 5.2.4) of the lifted points in  $\mathbb{R}^{d+1}$ . Then  $w = \frac{d}{d+1}\tilde{w}$  is an optimal dual solution for the free-center MVEE (Proposition 5.2.2) in  $\mathbb{R}^d$ .*

*Proof.* We use KKT conditions for both problems.

**KKT (free-center in  $\mathbb{R}^d$ ).** For any optimal primal-dual pair  $(Q, c, w)$  in  $\mathbb{R}^d$ ,

$$Q^{-1} = M = \sum_i w_i (a_i - c)(a_i - c)^\top, \quad c = \frac{\sum_i w_i a_i}{\sum_i w_i}, \quad \sum_i w_i = d, \quad (\text{C.1})$$

and complementary slackness:  $w_i > 0 \Rightarrow (a_i - c)^\top Q (a_i - c) = 1$ .

**KKT (origin-centered on the lift in  $\mathbb{R}^{d+1}$ ).** For any optimal primal-dual pair  $(\hat{Q}, \tilde{w})$  on the lifted points  $\tilde{a}_i$ ,

$$\hat{Q}^{-1} = \hat{M} = \sum_i \tilde{w}_i \tilde{a}_i \tilde{a}_i^\top, \quad \sum_i \tilde{w}_i = d + 1, \quad (\text{C.2})$$

and complementary slackness:  $\tilde{w}_i > 0 \Rightarrow \tilde{a}_i^\top \hat{Q} \tilde{a}_i = 1$ .

**From  $d$  to  $d + 1$ .** Let  $w$  be optimal in  $\mathbb{R}^d$  and set  $c$  and  $Q, M$  as in (C.1). Define the lifted matrix

$$\hat{A} := \sum_i w_i \tilde{a}_i \tilde{a}_i^\top = \sum_{i=1}^n w_i \tilde{a}_i \tilde{a}_i^\top = \begin{pmatrix} \sum_i w_i a_i a_i^\top & \sum_i w_i a_i \\ \sum_i w_i a_i^\top & \sum_i w_i \end{pmatrix}.$$

The Schur complement using (C.1) gives

$$\hat{A}^{-1} = \begin{pmatrix} M^{-1} & -M^{-1}c \\ -c^\top M^{-1} & c^\top M^{-1}c + \frac{1}{d} \end{pmatrix} \implies \tilde{a}_i^\top \hat{A}^{-1} \tilde{a}_i = (a_i - c)^\top M^{-1}(a_i - c) + \frac{1}{d}. \quad (\text{C.3})$$

Then, we set

$$\hat{Q} := \frac{d}{d+1} \hat{A}^{-1}, \quad \tilde{w} := \frac{d+1}{d} w.$$

Thus, we have  $\sum_i \tilde{w}_i = (d+1)/d \sum_i w_i = d+1$  and

$$\hat{Q}^{-1} = \frac{d+1}{d} \hat{A} = \sum_i \frac{d+1}{d} w_i \tilde{a}_i \tilde{a}_i^\top = \sum_i \tilde{w}_i \tilde{a}_i \tilde{a}_i^\top.$$

By (C.3) and feasibility of  $(Q, c, w)$ , we have for any point  $a_i$  with  $(a_i - c)^\top Q(a_i - c) \leq 1$ ,

$$\tilde{a}_i^\top \hat{Q} \tilde{a}_i = \frac{d}{d+1} \left( (a_i - c)^\top M^{-1}(a_i - c) + \frac{1}{d} \right) \leq 1,$$

with equality whenever  $w_i > 0$  (since then  $(a_i - c)^\top Q(a_i - c) = 1$ ). Thus, primal feasibility and complementary slackness hold for  $(\hat{Q}, \tilde{w})$ , which implies optimality.

**From  $d+1$  to  $d$ .** Let  $(\hat{Q}, \tilde{w})$  be optimal in  $\mathbb{R}^{d+1}$  and set

$$w := \frac{d}{d+1} \tilde{w}, \quad c := \frac{\sum_i \tilde{w}_i a_i}{\sum_i \tilde{w}_i} = \frac{\sum_i w_i a_i}{\sum_i w_i}.$$

Define the matrix in  $\mathbb{R}^{d \times d}$

$$M := \sum_i w_i (a_i - c)(a_i - c)^\top \quad \text{and} \quad Q := M^{-1}.$$

Let  $\hat{A} := \sum_i w_i \tilde{a}_i \tilde{a}_i^\top$ ; then

$$\hat{Q}^{-1} = \sum_i \tilde{w}_i \tilde{a}_i \tilde{a}_i^\top = \frac{d+1}{d} \sum_i w_i \tilde{a}_i \tilde{a}_i^\top = \frac{d+1}{d} \hat{A} \implies \hat{Q} = \frac{d}{d+1} \hat{A}^{-1}.$$

Writing  $\hat{A}^{-1}$  in the block form (C.3) yields

$$\tilde{a}_i^\top \hat{Q} \tilde{a}_i = \frac{d}{d+1} \left( (a_i - c)^\top M^{-1} (a_i - c) + \frac{1}{d} \right).$$

Thus, since  $\tilde{a}_i^\top \hat{Q} \tilde{a}_i \leq 1$ , we have  $(a_i - c)^\top M^{-1} (a_i - c) \leq 1$ . So,  $(Q, c)$  is primal feasible in  $\mathbb{R}^d$ . If  $\tilde{w}_i > 0$ , then  $\tilde{a}_i^\top \hat{Q} \tilde{a}_i = 1$  by the complementary slackness in  $\mathbb{R}^{d+1}$ , hence  $(a_i - c)^\top M^{-1} (a_i - c) = 1$ , which implies complementary slackness for  $(Q, c, w)$  as well. Thus,  $(Q, c, w)$  is an optimal solution in  $\mathbb{R}^d$ .  $\square$

# Vita

## Vaidehi Srinivas

|                  |                                                      |
|------------------|------------------------------------------------------|
| <b>2020</b>      | B.S. in Computer Science, Carnegie Mellon University |
| <b>2020-2021</b> | Fulbright Combined Award for Austria                 |
| <b>2026</b>      | Ph.D. in Computer Science, Northwestern University   |

## Field of study

Theoretical computer science, with a focus on the foundations of machine learning

## Publications

Vaidehi Srinivas, David P. Woodruff, Ziyu Xu, and Samson Zhou. Memory bounds for the experts problem. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, page 1158–1171, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392648. doi: 10.1145/3519935.3520069. URL <https://doi.org/10.1145/3519935.3520069>

Liam O’Carroll, Vaidehi Srinivas, and Aravindan Vijayaraghavan. The burer-monteiro SDP method can fail even above the barvinok-pataki bound. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL <https://openreview.net/forum?id=C2Mikd2Wp0c>

Aditya Bhaskara, Eric Evert, Vaidehi Srinivas, and Aravindan Vijayaraghavan. New tools for smoothed analysis: Least singular value bounds for random matrices with dependent entries. In *Proceedings of the 56th Annual ACM Symposium on Theory of Com-*

- puting, STOC 2024, page 375–386, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703836. doi: 10.1145/3618260.3649765. URL <https://doi.org/10.1145/3618260.3649765>
- Quanquan C. Liu and Vaidehi Srinivas. The predicted-updates dynamic model: Offline, incremental, and decremental to fully dynamic transformations. In Shipra Agrawal and Aaron Roth, editors, *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pages 3582–3641. PMLR, 30 Jun–03 Jul 2024. URL <https://proceedings.mlr.press/v247/liu24c.html>
- Avrim Blum and Vaidehi Srinivas. *Competitive strategies to use “warm start” algorithms with predictions*, pages 3775–3801. 2025. doi: 10.1137/1.9781611978322.127. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611978322.127>
- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Volume optimality in conformal prediction with structured prediction sets. In *Forty-second International Conference on Machine Learning*, 2025b. URL <https://openreview.net/forum?id=oNDhnGrD51>
- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Computing high-dimensional confidence sets for arbitrary distributions. In Nika Haghtalab and Ankur Moitra, editors, *Proceedings of Thirty Eighth Conference on Learning Theory*, volume 291 of *Proceedings of Machine Learning Research*, pages 2227–2269. PMLR, 30 Jun–04 Jul 2025a. URL <https://proceedings.mlr.press/v291/gao25b.html>
- Dionysis Arvanitakis, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Guarantees for alternating least squares in overparameterized tensor decompositions. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=9FDErIfoVE>
- Vaidehi Srinivas. Online conformal prediction with efficiency guarantees. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2026. URL <https://arxiv.org/abs/2507.02496>
- Chao Gao, Liren Shan, Vaidehi Srinivas, and Aravindan Vijayaraghavan. Learning confidence ellipsoids and applications to robust subspace recovery. In *Proceedings of 67th Symposium on the Foundations of Computer Science*, 2026. URL <https://arxiv.org/abs/2512.16875>