UNIVERSITY OF CALIFORNIA

Los Angeles

Communication and Computationally Efficient Learning Algorithms

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical and Computer Engineering

by

Osama Ashraf Hanna Habib

2024

ABSTRACT OF THE DISSERTATION

Communication and Computationally Efficient Learning Algorithms

by

Osama Ashraf Hanna Habib
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2024
Professor Christina Panagio Fragouli, Chair

The growing availability of data and rapid advancements in machine learning are revolutionizing decision-making. Often, these data come from distributed devices with low computational capabilities, connected to a central learner through communication constrained channels. As a result, developing learning algorithms with low communication and computational demands is increasingly attracting attention in the literature and in practice. In this thesis, we solve open problems within this framework for three popular learning setups: Contextual Bandits, Multi-Modal Representation Learning, and Classification. Contextual Bandits is an online learning problem where an agent makes decisions based on contextual information and receives feedback (rewards) for these decisions. The agent learns from past interactions to optimize future rewards. This problem models various practical applications such as recommendation systems, clinical trials, and resource allocations. This thesis offers the following contributions to this area. First, we propose a method that uses only 3 bits to send each reward, even if the rewards take values from an infinite set, while achieving (nearly) optimal learning performance. Second, we prove a surprising reduction from contextual to non-contextual bandits (where only a single context is available for all users). This allows to solve contextual bandits without the need to communicate the context, which can be communication heavy in many practical setups. The reduction provides a framework for developing efficient contextual bandit algorithms by using the simpler algorithms proposed for non-contextual bandits, leading to improved performance bounds in a number of setups such as contextual bandits with sparse unknown parameters, misspecification, privacy constraints, adversarial corruption, and many others. The reduction takes a step into solving multiple open problems in these setups. Lastly, we introduce the first computationally efficient algorithm for contextual bandits with limited policy switches, which is highly relevant to most practical scenarios. In most setups, the learner cannot frequently update the policy due to communication or computation limits, or because of high response rates, as in the case of clinical trials or recommendation systems. In the second setup, Multi-Modal Representation Learning, the focus is in extracting common and private features from multi-modal data, which is required for various applications such as vehicle tracking and medical diagnosis from multi-modal sources, e.g., images, and audio. This thesis introduces the notion of Common Information Dimension (CID), which quantifies continuous common information between sources in terms of number of dimensions. Unlike existing bit-based measures that cannot deal with continuous common information, CID provides a fundamental limit on the dimension of common latent variables in multi-modal learning applications. We develop methods to compute CID for distributions of interest to practical scenarios. Lastly, the Classification setup aims to compress features that are collected at distributed nodes and sent to a central entity for classification. This scenario is encountered by many machine learning applications including wireless cyberphysical systems, immersive environments and supported health. This thesis proposes a compression scheme tailored to classification tasks that allows the central entity to use an existing classifier that operates transparently to the feature compression and maintains the (uncompressed) performance. Although we prove the NP-hardness of finding the optimal compression scheme, we introduce a computationally efficient compression algorithms to approximately solve the problem. We experiment our schemes in practical setups showing significant savings in both the communication and computation costs while achieving state-of-the-art performance.

The dissertation of Osama Ashraf Hanna Habib is approved.

Lieven Vandenberghe

Suhas N. Diggavi

Lin Yang

Christina Panagio Fragouli, Committee Chair

University of California, Los Angeles

2024

To my aunt and my mother . . .

who loved me unconditionally

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This section is dedicated to the story of the PhD journey of Osama Hanna, an ordinary person who dreamed to be a pilot, a policeman, a lawyer, a doctor, a scientist, and a mathematician, but became an engineer. He wants to share with you a part of his journey, which, unlike his results, you cannot find in published papers. He also wants to avoid losing it due to his one-week memory; he wants to read it later to encourage himself, give himself hope during hard times, and make himself happy during sad times. Throughout the story, he wants to introduce important people who have contributed to this thesis more than he did.

He remembers his elementary and primary school days when he struggled to pass his exams. Every year, he dreamed of being ranked 1st in school, yet the results were too embarrassing to reveal to his family and friends. He dreamed of traveling abroad, even for a day, but deep down believed it impossible. At some point, his family, being realistic, tried to find him a job that did not require a college degree. He thought that he did not have the energy and strength to work hard and achieve academic success, but he never lost hope. He wants his journey to provide you, the reader, hope no matter how unrealistic your dreams may be. He hopes, by reading this thesis, to give you the joy that comes with understanding a new idea—the joy that keeps him thinking and motivates him to do research. He wants you to be part of his journey and hopes that you give this thesis life, even after he passes away.

His learning ability was hindered by the inability to memorize results, spend long hours studying, and lack of energy and concentration. However, his hope began in 7th grade when his science teacher, **Mr. Emad Halim**, recognized his ability to process information quickly and understand deductions, despite his poor memory. **Emad**'s encouragement and thorough explanations of the reasoning allowed him to master the subject without studying at home, topping his class for the first time. Even though it was only one subject, and he barely passed other subjects, the hope that came with it was most important.

In high school, his mathematics teacher, **Mr. Refaat Qaldes**, played a pivotal role. Unlike

most high school math teachers, **Refaat** explained the proof of every result he came across, even when beyond the curriculum, which greatly enhanced his understanding. **Refaat** encouraged him, engaging in proof discussions and even urging him to suggest alternative proofs. This approach enabled him to prove results he had not yet studied, such as formulas for the summation of powers of natural numbers. It impacted his ability to think, enabled him to master math-based results in different subjects in high school, and gave him time to dedicate his limited energy to studying other subjects. These experiences were crucial in his journey towards his PhD.

His PhD journey was shaped by several key figures, most importantly his advisor, **Prof. Christina Fragouli**. **Christina** taught him many ways to approach research problems; he learned from her how to simplify complicated problems to simple examples that can later generalize. She was always engaged in the details of the problems, discussing the mathematical proofs in details and offering diverse approaches. He enjoyed these inspiring technical discussions with **Christina** which often left him wondering "Why didn't I think of that?". **Christina** taught him how to abstract / extract problems from applications, and connect them to broader contexts. Despite **Christina**'s extensive knowledge and senior position, she is always willing and eager to learn and is very humble—qualities that he always admired. She encouraged him to explore various problems and areas, promoting his curiosity and growth. She always shows excitement for her students' successes, and generously attributes all credit to them.

But she did not only teach him to think about research problems. He remembers the first day he met **Christina** and how welcoming and cheerful she was; he felt that he knew **Christina** from before. **Christina** guided him on how to interact with others, public speaking, prepare presentations, and connect with others; he continues to learn from her how to balance kindness and fairness (he even adopted her semicolon style). She always believed in him, allocated a significant amount of time to help him, connect him with others and did everything in his best interest. More than academically, through her success, kindness, respect, and welcoming attitude to everyone, she has been a role model for him. He will always cherish her words, "I am proud of you." **Christina** is and will always be one of his biggest sources of support, and he wonders if she would consider him a

current PhD for life.

His co-advisor **Prof. Lin Yang**: He started working with **Lin** in the third year of his PhD, and once he did, he fell in love with the problems **Lin** was working on as they fulfilled his interest in math. He very much enjoyed the research discussions with **Lin**, which stimulated his mind and made him look forward to the next meeting. It was very easy and efficient to exchange ideas with **Lin**; they said: "we think you and **Lin** have a common efficient codebook." **Lin** treats everyone with respect and kindness. **Lin** genuinely cares about his future, encouraged him, helped to connect him with other researchers and professors, and introduced him to multiple job opportunities. He will always be grateful for **Lin**'s guidance, support, and the invaluable role played in shaping his academic journey and future career.

Prof. Suhas Diggavi: **Suhas** always considered him his own student, teaching and helping him, and offering support both before and after graduation. **Suhas**'s breadth of knowledge and diverse interests greatly helped him expand his understanding of various areas and feel more freedom regarding possible research directions. Beyond information theory classes, probability classes, and research collaboration, he also learned from **Suhas** (and is still learning) how to be organized and act professionally. He appreciates **Suhas**'s advice on how to be a successful researcher, as well as guidance on job-related matters. He is deeply grateful for **Suhas**'s continuing support and mentorship.

He extends his gratitude to committee member **Prof. Lieven Vandenberghe** for his insightful suggestions and stimulating discussions. **Lieven**'s classes on convex optimization were invaluable, significantly aiding him in various research problems and beyond.

Prof. Mohammed Nafie: During his undergraduate studies, **Mohammed Nafie** was one of the few professors who prioritized teaching ideas and ways of thinking over rote memorization. He felt a special connection with **Dr. Nafie**'s thought process and teaching style. He could grasp **Dr. Nafie**'s lectures more quickly than any others, finding it impossible to lose focus during class. **Dr. Nafie** soon became his favorite professor and instructor. He eagerly awaited these mind-stimulating and enjoyable lectures each week. After graduating, he longed to experience those lectures again, though he knew it would not be the same as the first time when **Dr. Nafie** excited students' minds with unfamiliar concepts. In his final undergraduate year, **Dr. Nafie** encouraged him to pursue postgraduate studies and later became his master's advisor. Discussing problems with **Dr. Nafie** was a great and inspiring experience. **Dr. Nafie** remains one of his most favorite professors, instructors, and personalities. He still wishes he could return to learn more from **Dr. Nafie**.

He wants to thank his advisors: **Christina**, **Lin**, **Suhas**, and **Mohammed**, for accepting him as their student and helping him succeed. They will always be on his mind, an important part of his journey, and a great support for him. Without their help and acceptance, it would not have been possible for him to achieve any success; for this, he owes them everything he has. Nothing he could do can repay even a small part of this debt, and no words can fully express his gratitude to them.

In the field of bandits and reinforcement learning, **Tor Lattimore** and **Csaba Szepesvári** became his inspirations. **Tor**'s stream of novel, mind-stimulating ideas in yearly publications impressed and inspired him. His book, with **Csaba**, on bandit algorithms has been the main reference that introduced him to the leading results in bandits. He met **Tor** for the first time at COLT 2023, and he was inspired once again by how humble, kind, and encouraging he was. He will always cherish **Tor**'s words: "I liked your result." **Tor** has kindly supported him with reference letters for his job applications, which he is very proud of.

He first encountered **Csaba**'s name through the bandit algorithms book and his numerous leading papers on bandits and reinforcement learning. Their first meeting was when **Csaba** chaired the 2022 ICML workshop on RL Theory where he presented. Not knowing **Csaba**'s appearance, he was charmed by the chair's funny, cheerful, and engaging demeanor. After the workshop, he searched until he realized it was **Csaba** himself! Following **Csaba**'s recorded lectures and talks, he quickly grew to admire him. He was impressed by **Csaba**'s close attention to technical details and mathematics of new results, even at a senior level. He admired **Csaba**'s love of science and knowledge, and his dedication to helping others. Upon meeting **Csaba** in person at NeurIPS 2023, he wanted to jump up saying "I love you **Csaba**!" but he restrained himself out of embarrassment. He wants to thank **Csaba** and **Tor** for the inspirations, discussions, help, advice, encouragement, and making the RL theory community accepting and accessible for junior researchers.

He also acknowledges the reinforcement learning theory community, especially **Julia Olkhovskaya**, **Gergely Neu**, **Karthik Sridharan**, **Ayush Sekhari**, **Shinji Ito**, **Andras Gyorgy**, and **Dylan Foster**. He deeply appreciated their acceptance, humility, kindness, and inclusiveness, which encouraged him to engage in discussions, contribute, and aspire to be part of the community. He thoroughly enjoyed their inspiring conversations, which left him eagerly waiting for the next conference to reconnect and discuss ideas with them.

He also wants to thank his collaborator **Prof. Bhaskar Krishnamachari**, who promptly helped him with his job applications and gave him precious advice.

Prof. Petar Popovski: for hosting him at Aalborg University during his master's, providing support, advice, and inspiring discussions. **Petar**'s kindness and support left a lasting impression.

Prof. Tamer El-Batt, **Amr El-Sherif**, and **Amr El-Keyi**: for their mentorship during his master's, help, support, advice, and encouragement.

An important and highly enjoyable part of his PhD journey was his math classes. He devoted significant time during the second and third years to these courses, completing a total of 12. For this rewarding experience, he deeply thanks his math professors, especially **Prof. Monica Visan**, **Artem Chernikov**, **Yiannis Moschovakis**, and **Patrick Lutz**.

Yahya Ezzeldin was a mentor during the first part of his PhD, guiding him and helping convey his ideas. He always admired **Yahya**'s honesty, helpfulness, and dedication.

Antonious Girgis is his collaborator, roommate, and lifelong friend. They have shared countless experiences and explored many things together. He will always cherish their cooking times, short walks (ending up to be 10 miles), breaking things and attempting (often failing) to fix them, as well as their research and general discussions.

His lab collaborators, **Xinlin Li** and **Merve Karakas**: he very much enjoyed collaborating with them and learned from them more than they learned from him. He always felt happy and rewarded for their achievements as his own. They often helped him with proofreading, job applications, and offered valuable advice. They even helped improve his presentation slide contents, design, and colors when he only used black and yellow (his favorite childhood color).

He wants to acknowledge his lab colleagues **Chaorui Yao** and **Mine Dogan** for their kindness and humorous presence, as well as his senior lab colleagues **Mohammed Karmoose**, **Gaurav Agarwal**, **Sundara Rajan Srinivasavaradhan**, **Martina Cardone**, and **Linqi Song**. He deeply appreciates their help, support, and the fun moments they shared, including restaurant explorations and tea nights. His friends at UCLA—**Navjot Singh**, **Mohamad Rida Rammal**, **Ruida Zhou**, **Dhaivat Joshi**, **Kaan Ozkara**, and **Samer Hanna**—made his time there enjoyable with their walks, discussions, and games. He always felt he could rely on them in times of need.

His MSc friends—**Mohamed Abd-Elmagid**, **Abdulmoneam Ali**, **Ahmed Elkordy**, and **Belal ElDiwany**—hold a special place in his memories of that time.

He is grateful to **Moustafa Alzantot** for being a supportive friend and offering valuable advice about jobs, career, and life in general. He thanks **Mahmoud Elhebeary** for his accepting personality and the fun discussions during Fortnite games, which provided moments of joy and levity.

His lifelong friend **Osama Fawzi** has always been a source of support and encouragement. He cherishes their shared experiences, from undergraduate projects to last-minute exam preparations, discussions, trips, walks, and PlayStation days. **Osama**'s constant presence has been a comfort throughout his journey.

Finally, he expresses deep gratitude to his family: his **aunt**, **mom**, **dad**, siblings (**Mariam**, **Sara**, and **Abanoub**), and **Burhan Turgut**. They supported him through thick and thin, tolerating his disorganization and poor memory. He deeply appreciates their sacrifices and support for his choices. Their help extended to even the smallest challenges, such as waking him from his heavy sleeps. They are not just family but also his closest friends and confidants. They celebrated his successes, no matter how small, and provided comfort during setbacks. Their humor and lightheartedness often lifted his spirits during the most stressful times. Their love, patience, and unconditional support have been the bedrock of his journey. His achievements are as much theirs as they are his own.

As he does not have a partner, he thanks his future partner for allowing him time to focus on his PhD, and (humorously) takes this opportunity to hint that he is available.

He concludes by emphasizing that without all these individuals, he would never have been able to achieve any success. His journey stands as a testament to the power of hope, perseverance, and the impact of supportive relationships in academic pursuits.

VITA

2014	BSc in Electrical and Computer Engineering, Cairo University, Egypt.
2016-2018	Research Assistant, Nile University, Egypt.
2018	MSc in Communications and Information Technology, Nile University, Egypt.
2018-2023	Graduate Student Researcher, UCLA.
2019, 2020	Teaching Assistant, UCLA.
2024	Research Scientist, Meta AI.

SELECTED PUBLICATIONS

- [HYF24] Hanna, Osama, Lin Yang, and Christina Fragouli. "Efficient batched algorithm for contextual linear bandits with large action space via soft elimination." Advances in Neural Information Processing Systems 36 (2024).
- [HYF23] Hanna, Osama A., Lin Yang, and Christina Fragouli. "Contexts can be cheap: Solving stochastic contextual bandits with linear bandit algorithms." The Thirty Sixth Annual Conference on Learning Theory. PMLR, 2023.
- [HYF22a] Hanna, Osama, Lin Yang, and Christina Fragouli. "Learning from distributed users in contextual linear bandits without sharing the context." Advances in Neural Information Processing Systems 35 (2022): 11049-11062.

- [HYF22b] Hanna, Osama A., Lin Yang, and Christina Fragouli. "Solving multi-arm bandit using a few bits of communication." International Conference on Artificial Intelligence and Statistics. PMLR, 2022.

- [HYF22c] Hanna, Osama A., Lin F. Yang, and Christina Fragouli. "Compression for multi-arm bandits." IEEE Journal on Selected Areas in Information Theory 3.4 (2022): 773-788.

- [HGF24] Hanna, Osama, et al. "Differentially private stochastic linear bandits:(almost) for free." IEEE Journal on Selected Areas in Information Theory (2024).

- [HKY24, HKY23] Hanna, Osama A., et al. "Multi-Agent Bandit Learning through Heterogeneous Action Erasure Channels." International Conference on Artificial Intelligence and Statistics. PMLR, 2024.

- [HLD23] Hanna, Osama A., et al. "Common information dimension." 2023 IEEE International Symposium on Information Theory (ISIT). IEEE, 2023.

- [HLD24] Hanna, Osama A., et al. "On the Relation Between the Common Information Dimension and Wyner Common Information." 2024 IEEE International Symposium on Information Theory (ISIT). IEEE, 2024.

- [HEF21] Hanna, Osama A., et al. "Quantization of distributed data for learning." IEEE Journal on Selected Areas in Information Theory 2.3 (2021): 987-1001.

- [HES20] Hanna, Osama A., et al. "On distributed quantization for classification." IEEE Journal on Selected Areas in Information Theory 1.1 (2020): 237-249.

CHAPTER 1

Introduction

Today, we have more data than ever, leading to big advancements in machine learning. In most cases, this data comes from distributed small devices with low computational capabilities, connected to a central learner through communication constrained channels. As a result, there is growing interest in the literature in developing learning algorithms that require less communication and computation. This setup is relevant in a variety of applications, such as recommendation systems, supported health, clinical trials, a traffic management system directing drones, brain-to-computer interface applications capturing brain signals via multiple electrodes, and many others. This thesis addresses open problems within this framework for three popular learning setups: Contextual Bandits, Multi-Modal Representation Learning, and Classification.

Part I of the thesis focuses on contextual bandits, where we propose reward and context compression approaches while achieving the same learning performance as the uncompressed algorithms. Additionally, we propose the first computationally efficient algorithm for contextual bandits with limited policy switches. Towards achieving our results, we develop a reduction from contextual to non-contextual bandits (where only a single context is available for all users), which leads to improved regret bounds for contextual bandits in many setups.

Part II of the thesis focuses on multi-modal representation learning where the goal is to extract common and private features from multi-modal data. The thesis introduces the notion of Common Information Dimension (CID), which quantifies continuous common information between sources in terms of number of dimensions. Unlike existing bit-based measures that cannot deal with continuous common information, CID provides a fundamental limit on the dimension of common latent variables in multi-modal learning applications. We develop methods to compute CID for distributions of interest to practical scenarios.

Part III of the thesis considers a classification setup aiming to compress features that are collected at distributed nodes and sent to a central entity for classification. We propose a compression scheme tailored to classification tasks that allows the central entity to use an existing classifier that operates transparently to the feature compression and maintains the (uncompressed) performance. Although we prove the NP-hardness of finding the optimal compression scheme, we introduce a computationally efficient compression algorithms to approximately solve the problem.

Overview of Part I (Chapters 2-4): Contextual Bandits

Contextual Bandits is an online learning problem where an agent makes decisions based on contextual information and receives feedback (rewards) for these decisions. The agent learns from past interactions to optimize future rewards. This problem models various practical applications such as recommendation systems, clinical trials, and resource allocations.

Chapter 2 of the thesis presents a reduction from contextual to non-contextual bandits. This allows to solve contextual bandits without the need to communicate the context (communication heavy) in many practical cases. The reduction provides a recipe for developing efficient contextual bandit algorithms by using the simpler algorithms proposed for non-contextual bandits, leading to improved performance bounds in a number of setups such as contextual bandits with sparse unknown parameters, misspecification, privacy constraints, adversarial corruption, and many others. The reduction takes steps into solving open problems in these setups. This chapter serves as a basis for some techniques used in the following chapters. This work in this chapter is based on the following publications:

- [HYF23] Hanna, Osama A., Lin Yang, and Christina Fragouli. "Contexts can be cheap: Solving stochastic contextual bandits with linear bandit algorithms." The Thirty Sixth Annual Conference on Learning Theory. PMLR, 2023.
- [HYF22a] Hanna, Osama, Lin Yang, and Christina Fragouli. "Learning from distributed users

in contextual linear bandits without sharing the context." Advances in Neural Information Processing Systems 35 (2022): 11049-11062.

Chapters 3 studies contextual bandits with limited policy switches. This is a restriction for many real-world use cases due to computation or communication considerations, or may be imposed by the nature of the application, as is the case in multi-stage clinical trials or online marketing campaigns with high response rates, where it is not feasible to update the policy after each response. Despite continuous effort in the literature, existing algorithms require exponential complexity. We provide the first computationally efficient algorithm in this setup that achieves (nearly) optimal regret and uses the smallest number of policy switches (up to a small constant factor). This work in this chapter is based on the following publication:

- [HYF24] Hanna, Osama, Lin Yang, and Christina Fragouli. "Efficient batched algorithm for contextual linear bandits with large action space via soft elimination." Advances in Neural Information Processing Systems 36 (2024).

In chapter 4, we propose a reward compression scheme for distributed bandit applications. We propose a method that uses only 3 bits to send each reward, even if the rewards take values from an infinite set. The method can achieve the same learning performance as the best algorithm that uses uncompressed rewards. This work in this chapter is based on the following publications:

- [HYF22b] Hanna, Osama A., Lin Yang, and Christina Fragouli. "Solving multi-arm bandit using a few bits of communication." International Conference on Artificial Intelligence and Statistics. PMLR, 2022.

- [HYF22c] Hanna, Osama A., Lin F. Yang, and Christina Fragouli. "Compression for multi-arm bandits." IEEE Journal on Selected Areas in Information Theory 3.4 (2022): 773-788.

Overview of Part II (Chapter 5): Multi-Modal Representation Learning

The common randomness between dependent random variables is a fundamental problem in information theory and has ubiquitous applications in a number of areas, such as key generation in cryptography, hypothesis testing in statistical inference, and multi-modal representation learning in machine learning. Existing notions of common information only characterize the common information in terms of bits which can be infinite whenever continuous random variables are involved. Chapter 5 introduces the new notion of Common Information Dimension (CID) which characterizes the amount of common information between random variables in terms of number of dimensions. Our new metric sets a fundamental limit on the dimension of common latent variables in multi-modal learning applications. We propose a method to compute the common information dimension for a special class of distributions, that are of interest in practical applications. Additionally, we show that the common information dimension characterizes the amount of randomness (in bits) required to (approximately) distributedly simulate a pair of random variables with infinite common information. The work in this chapter is based on the following publications:

- [HLD23] Hanna, Osama A., et al. "Common information dimension." 2023 IEEE International Symposium on Information Theory (ISIT). IEEE, 2023.
- [HLD24] Hanna, Osama A., et al. "On the Relation Between the Common Information Dimension and Wyner Common Information." 2024 IEEE International Symposium on Information Theory (ISIT). IEEE, 2024.

Overview of Part III (Chapter 6): Distributed Classification

Chapter 6 studies distributed feature quantization for data that is going to be used for classification. This problem is often encountered in applications such as wireless cyberphysical systems, immersive environments, and supported health. The goal is to enable a pretrained classifier at a central node to carry out its classification on features that are gathered from distributed nodes through communication constrained channels. Our main contributions in this chapter include: we prove NP-hardness of finding optimal quantizers in the general case; we design an optimal scheme for a special case; we propose quantization algorithms, that leverage discrete neural representations and training data, and can be designed in polynomial-time for any number of features, any number of classes, and arbitrary division of features across the distributed nodes. We find that tailoring the quantizers to the classification task can offer significant savings: as compared to alternatives, we can achieve more than a factor of two reduction in terms of the number of bits communicated, for the same classification accuracy. This work in this chapter is based on the following publication:

- [HES20] Hanna, Osama A., et al. "On distributed quantization for classification." IEEE Journal on Selected Areas in Information Theory 1.1 (2020): 237-249.

CHAPTER 2

A Reduction from Contextual to Linear Bandits

2.1 Summary

In this chapter, we address the stochastic contextual linear bandit problem, where a decision maker is provided a context (a random set of actions drawn from a distribution). The expected reward of each action is specified by the inner product of the action and an unknown parameter. The goal is to design an algorithm that learns to play as close as possible to the unknown optimal policy after a number of action plays. This problem is considered more challenging than the linear bandit problem, which can be viewed as a contextual bandit problem with a *fixed* context. Surprisingly, in this chapter, we show that the stochastic contextual problem can be solved as if it is a linear bandit problem. In particular, we establish a novel reduction framework that converts every stochastic contextual linear bandit instance to a linear bandit instance, when the context distribution is known. When the context distribution is unknown, we establish an algorithm that reduces the stochastic contextual instance to a sequence of linear bandit instances with small misspecifications and achieves nearly the same worst-case regret bound as the algorithm that solves the misspecified linear bandit instances. As a consequence, our results imply a $O(d\sqrt{T \log T})$ high-probability regret bound for contextual linear bandits, making progress in resolving an open problem in [LWZ19, LWC21]. Our reduction framework opens up a new way to approach stochastic contextual linear bandit problems, and enables improved regret bounds in a number of instances including the batch setting, contextual bandits with misspecifications, contextual bandits with sparse unknown parameters, and contextual bandits with adversarial corruption.

Algorithm	Regret Bound	Type	Assumption/restriction
[APS11]	$O(d\sqrt{T}\log T)$	w.h.p.	
[LWC21]	$O(d\sqrt{T\log T}poly(\log\log T))$	exp	
[LWC21]	$O(d\sqrt{T}\log T poly(\log\log T))$	w.h.p.	
Ours	$O(d\sqrt{T\log T})$	w.h.p.	stochastic context
[RYZ21]	$O(d\sqrt{T\log(d)\log(T)}\log\log T)$	exp	batch learning with
Ours	$O(d\sqrt{T\log(T)}\log\log T)$	w.h.p.	$O(\log\log T)$ batches
[FGM20]	$O(d\sqrt{T}\log T+\epsilon\sqrt{d}T)$	exp	misspecified
Ours	$O(d\sqrt{T\log T}+\epsilon\sqrt{d}T\log T)$	w.h.p.	
[FGM20]	$\tilde{O}(d^{4.5}\sqrt{T}+d^4C)$	w.h.p.	adversarial corruption
Ours	$\tilde{O}(d\sqrt{T}+d^{3/2}C)$	w.h.p.	
[APS12]	$O(\sqrt{dsT}\log T)$	w.h.p.	sparse
Ours	$O(\sqrt{dsT\log T})$	w.h.p.	
Ours	$O(\sqrt{dsT\log T}\log\log T)$	w.h.p.	sparse with
			$O(\log\log T)$ batches

Table 2.1: Comparison of best known in literature vs. our approach regret bounds. Here, d is the model dimension, T is the time horizon, ϵ is an upper bound on the amount of misspecification, C limits the power of adversary, s is an upper bound on the number of non-zero elements in the unknown parameter, exp indicates a regret bound in expectation, w.h.p. indicates a regret bound that holds with probability at least $1-1/T$, and $\tilde{O}$ hides $\log$ factors.

2.2 Introduction

Linear bandit and contextual linear bandit problems are attracting extensive attention - for example, more than $17,000$ papers appear when searching for “linear contextual bandit” on Google Scholar during the last 5 years - as they enable to support impactful active learning applications through elegant formulations. In linear bandits, a learner at each time $t \in [T]$, where T is the time horizon, pulls an arm a_t from a fixed action space $\mathcal{A}$ (that may be continuous or discrete), and receives a reward $r_t = \langle a_t, \theta_\star \rangle + \eta_t$, where $\theta_\star$ is an unknown d-dimensional vector of parameters and η_t is random noise. Contextual linear bandits add another layer of complexity by enabling at each round the action space to be different to capture context; in this case, the learner at time t observes an action space (context) $\mathcal{A}_t$. That is, we can think of linear bandits as single-context contextual bandits, observing $\mathcal{A}_t = \mathcal{A}$ for all t. For example, while linear bandits are used in recommendation systems where the set of actions is fixed and oblivious to the individual the recommendation is addressed to, contextual linear bandits are used in personalized recommendations, where the action space gets tailored to context attributes such as age, gender, income and interests of each individual.

It is not surprising that, although more limited in applications, linear bandits are much better understood in theory than contextual linear bandits. Indeed, algorithms for linear bandits often leverage the fixed action space property, and cannot be easily extended to contextual linear bandits. To give a concrete example, the algorithm Phased Elimination (PE) [LSW20, VMK14] leverages the fixed action space by exploring a (small) core set of actions to achieve good estimates of the rewards for all actions. This algorithm achieves a high probability regret bound of $O(d\sqrt{T \log T})$. Nevertheless, despite several attempts over the last decade [APS11,LWC21,LWZ19], the best known regret upper bounds for contextual linear bandits have a $\log$ (or iterated $\log$) multiplicative gap over the $O(d\sqrt{T \log T})$ bound both in high probability and in expectation. Similarly, the best known algorithms for several linear bandits problems (e.g., with misspecification, adversarial corruption, and others [LSW20, FGM20, BLK21, WDZ22, RYZ21]), perform better (in the worst-case) than the corresponding algorithms for contextual linear bandits.

2.2.1 Our Results

The Reduction

We show in this chapter the surprising result that, provided the context comes from a distribution $\mathcal{D}$ (stochastic context), contextual linear bandit problems can be reduced to solving (single context) linear bandit problem when the context distribution $\mathcal{D}$ is known, and to linear bandits with $\tilde{O}(1/\sqrt{T})$ misspecification when the distribution $\mathcal{D}$ is unknown. These results are presented in the following informal theorems and their exact statements are given in Theorems 1, 2, and 21 in App. A.1.

Informal Statement of Theorem 1. For any contextual linear bandit instance I with known context distribution $\mathcal{D}$, there exists (constructively) a linear bandit instance L with the same action dimension, and any algorithm solving L solves I with the same worst-case regret bound as L.

Informal Statement of Theorem 2. For any contextual linear bandit instance I with unknown context distribution $\mathcal{D}$, there exist (constructively) $\log T$ misspecified linear bandit instances $L_1, ..., L_{\log T}$, where L_i operates on part of the horizon of length T_i, has $\tilde{O}(1/\sqrt{T_i})$-misspecification and the same action dimension, and any algorithm solving $L_1, ..., L_{\log T}$ solves I with the same worst-case regret bound as $L_1, ..., L_{\log T}$.

Informal Statement of Theorem 21. For any contextual linear bandit instance I with unknown context distribution $\mathcal{D}$ but where now the action space has a component-wise product structure, there exists (constructively) a linear bandit instance L with double the action dimension of I, and any algorithm solving L solves I with the same worst-case regret bound as L.

Stochastic contextual bandits encompass practical cases where the context is not selected adversarially; in our example of personalized recommendations, the age, gender, income, come from distributions. Our framework simplifies the contextual linear bandit problem and enables to use any existing (or future) linear bandit algorithms to solve the contextual case. Moreover, our results separate the case of stochastic contexts from the harder case of adversarial contexts and explain why good results, which are not achievable for adversarial contexts, are possible for stochastic contexts.

Implications and Related Work

The equivalence we proved opens up a new way to approach stochastic contextual linear bandit problems, and results in a number of new results or recovery of existing results in a simpler manner; we next present some of these implications (summarized in Table 2.1) and their positioning with respect to related work. In the discussion next, d is the model dimension and T is the time horizon. These implications are stated formally and proved in Section 2.5.

• **Tighter Regret Bounds.** To the best of our knowledge, there is a gap in the regret bounds of contextual linear bandits: the state-of-the-art lower bounds are, $\Omega(d\sqrt{T})$ for linear bandits [LS20] and $\Omega(d\sqrt{T\log T})$ for linear contextual bandit with adversarial contexts [LWZ19].

Our contribution [Corollary 2 in Section 2.5]. Our approach achieves a regret upper bound $O(d\sqrt{T\log T})$ with high probability even when the action set is infinite. While it is not known if a $\Omega(d\sqrt{T\log T})$ lower bound holds for stochastic contexts, our result improves over state of the art high probability bounds by at least a factor of $\sqrt{\log T}$ and matches the best known upper bound for linear bandits.

Related Work. The best attempts of upper bounds are [APS11, LWC21, LWZ19]. In particular, [APS11] achieves a regret bound of $O(d\sqrt{T}\log T)$ with high probability; [LWC21] achieves $O(d\sqrt{T\log T}poly(\log\log T))$ in expectation and $O(d\sqrt{T}\log T poly(\log\log T))$ with high probability; and [LWZ19] achieves

$O(d\sqrt{T\log T}poly(\log\log(T)))$ *in expectation* and only when the number of actions is *finite and bounded by* $2^{d/2}$.

• **Batch Learning.** In batch learning, instead of observing the reward at the end of each round to decide what action to play next, the learning agent is constrained to split the rounds into a fixed number of M batches, use a predetermined policy within each batch, and it can only observe the action outcomes and switch the policy at the end of each batch. This is a central problem in online learning [GHR19, PRC16, RYZ21, HZZ20] as limited policy adaptivity enables parallelism and facilitates deployment of learning algorithms for large-scale models.

Our Contribution [Theorem 3 in Section 2.5]. Our batch algorithm for contextual linear bandits achieves with *high probability* $O(d\sqrt{T\log(T)}\log\log T)$ regret bound for $O(\log\log T)$ batches, a $\sqrt{\log d}$ better than the *in expectation* regret bound in [RYZ21].

Related Work. A number of works have explored batched contextual linear bandits, both in adversarial [APS11, HZZ20] and non-adversarial settings [RYZ21, ZJZ21]. The breakthrough work in [RYZ21] achieved a nearly optimal *in expectation* regret upper bound $O(d\sqrt{T\log d\log(T)}\log\log T)$ using $O(\log\log T)$ batches for the stochastic contexts setting. The near-optimality follows from the result in [GHR19] which shows that $\Omega(\log\log T)$ batches are required to achieve a $O(\sqrt{T})$ worst-case regret bound for multi-armed bandits with a finite number of arms (this is a special case of contextual linear bandits). If the contexts are chosen adversarially, $\Omega(\sqrt{T})$ batches are required to achieve a $O(\sqrt{T})$ regret bound [HZZ20].

• **Misspecified Bandits.** Linear bandit algorithms are designed under the assumption that the expected rewards are perfectly linear functions of the actions; misspecified bandits relax this assumption by considering perturbations of the linear model measured by the amount of deviation in the expected rewards (we call the case ϵ-misspecification if the deviation is upper bounded by ϵ). The non-linearity in the model enables to better capture real-world environments and is of high interest in the literature [DKW19, CG13, LSW20, FGM20, GCG17, FR20].

Our Contribution [Theorem 4, 5 in Section 2.5]. We provide a regret bound of $O(d\sqrt{T\log T}+\epsilon\sqrt{d}T\log T)$ with high probability for contextual bandits with unknown misspecification, and $O(d\sqrt{T\log T}+\epsilon\sqrt{d}T)$ with high probability for ϵ known. To the best of our knowledge, these results offer the first *optimal regret bounds* for ϵ-misspecified contextual linear bandits, and improve over existing literature for unknown ϵ by providing high probability bounds and improved $\log$ factors. We also present the first nearly optimal algorithm for misspecified contextual linear bandits with $O(\log T)$ batches.

Related Work. The work in [LSW20] shows that PE with modified confidence intervals achieves the optimal regret bound of $O(d\sqrt{T\log T}+\epsilon\sqrt{d}T)$ with high probability (matching the $\Omega(\epsilon\sqrt{d}T)$ lower bound [LSW20]) for linear bandits with known ϵ misspecification. If ϵ is unknown, the same algorithm was shown to achieve a regret bound of $O(d\sqrt{T\log T}+\epsilon\sqrt{d}T\log T)$ with high probability. The work in [ZLK20, LSW20] proposed variants of LinUCB that achieve a regret bound of $\tilde{O}(d\sqrt{T}+\epsilon\sqrt{d}T)$[1] with high probability for contextual linear bandits with known ϵ. However, changing action sets with unknown ϵ was left as an open problem. The works in [PPA20, FGM20] made progress in answering this question by providing a regret bound of $O(d\sqrt{T}\log T+\epsilon\sqrt{d}T)$ in expectation [FGM20]. However, improving the $\log$ factors in the first term and strengthening the result to a high probability bound was unresolved. Due to the challenge that changing action sets impose on the analysis, the techniques of [FGM20] cannot be directly extended to provide high probability bounds on the regret. Our result of $O(d\sqrt{T\log T}+\epsilon\sqrt{d}T\log T)$ regret removes the $\sqrt{\log T}$ factor from the first term and adds a $\log T$ factor in the second term; depending on the value of ϵ this can lead to a tighter or looser bound.

• **Bandits with Adversarial Corruption.** Linear bandits with adversarial corruption recently attracted significant interest [LSS21, CDJ21, JHL21, LLS19, BKS20, BLK21, LLW21, WDZ22] due to the vulnerability of online learning applications to attacks. There are multiple adversarial models that are proposed in the literature; here we consider a widely used model that assumes the adversary knows the policy, and observes the history, but does not observe the current action before corrupting the rewards.

Our Contribution [Theorem 6 in Section 2.5]. Our work provides the first algorithm for contextual linear bandits that achieves a regret bound of $\tilde{O}(d\sqrt{T}+d^{3/2}C)$ with high probability for unknown C, which upper bounds the total amount of corruption from the adversary. This improves over the best known $\tilde{O}(d^{4.5}\sqrt{T}+d^{4}C)$ bound for linear bandits with changing action sets in [WDZ22]. We note that in our regret bound, while the dependency on d in the first term is nearly

[1]Here $\tilde{O}$ hides log factors.

optimal, the dependency on $d^{3/2}$ in the second term is not. However, we simplify the problem of improving this dependency, as any algorithm that improves it for linear bandits will imply the same improvement for contextual bandits with our reduction.

Related Work. The work in [BLK21] considers linear bandits with adversarial corruption and achieves a regret bound of $\tilde{O}(d\sqrt{T}+d^{3/2}C)$ with high probability for known corruption level C and a regret bound of $\tilde{O}(d\sqrt{T}+d^{3/2}C+C^2)$ with high probability for a stronger adversary that observes the current action and unknown corruption level C, while the work in [WDZ22] achieves a regret bound of $\tilde{O}(d\sqrt{T}+d^{3/2}C)$ with high probability for unknown corruption C. These algorithms have optimal dependency on T, C but it is not known if the $d^{3/2}$ dependency is tight or not. The work in [LLW21] improves the dependency on d for linearized corruption, by achieving a regret bound of $\tilde{O}(d\sqrt{T}+C)$ with high probability which is also nearly optimal [LLW21]. The proposed algorithms and analysis rely on the assumption that the action set is fixed, and as far as we know there are no known generalizations for changing action sets - beyond the work in [BLK21] that considers changing action sets but imposes a strong assumption on the context distribution. The work in [HZZ22] considers a stronger adversary, that observes the current action, and achieves an $\tilde{O}(d\sqrt{T})$ regret bound for unknown $C \leq \sqrt{T}$ and linear regret otherwise. The first paper to prove a regret bound with nearly optimal dependency on T, C for linear bandits with changing action sets (linear contextual bandits) is [WDZ22] which achieves a regret bound of $\tilde{O}(d^{4.5}\sqrt{T}+d^4C)$ with high probability for unknown C. While the dependency on T, C is nearly optimal, the dependency on d is clearly not - improving this was left as an open problem. Our results take a step in this direction by removing a factor of $d^{3.5}$ from the first term, by removing a factor of $d^{2.5}$ from the second term and by reducing the problem of further improving the dependency of d in the second term to achieving this improvement over a linear bandit setup.

• **Sparsity.** High dimensional linear bandits with sparsity capture practical cases such as, when there exist a large number of candidate features and limited information on which of them are useful; use cases include personalized medicine and online advertising [BB20]. This setup results in sparsity in the unknown linear bandit parameters, which can be leveraged for more efficient learning.

Our Contribution [Theorem 7 in Section 2.5]. Our work provides the first $O(\sqrt{dsT \log T})$ regret bound with high probability for contextual linear bandits, improving a factor of $\sqrt{\log T}$ on the state of the art, where s is an upper bound of the nonzero elements in the model.

Related Work. Due to its practical significance, a number of works have examined this setup [LCS15, APS12, Ger11, CM12, CWF22, DWD22, JZJ22, HLD21, HLW20]. To the best of our knowledge, the best known regret bound is $O(\sqrt{dsT} \log T)$ with high probability [APS12]. While this is shown to be nearly optimal [LS20], improving the $\log T$ factor was left as an open problem. Our work resolves this.

2.2.2 Technical Overview

Our major technique innovation is in the conception and execution of a reduction from a stochastic contextual linear bandit instance to a linear bandit instance. This reduction is made possible by establishing a linear bandit action for each possible parameter θ of the contextual bandit instance. In particular, for a given θ, we establish a new action $g(\theta)$ that is the expected best action (under the distribution of the context $\mathcal{A}_t$) with respect to the parameter θ, i.e., $g(\theta) = \mathbb{E}_{\mathcal{D}}(\arg\max_{\tilde{a} \in \mathcal{A}_t} \langle \tilde{a}, \theta \rangle)$. Note that $g(\theta)$ may not correspond to any valid action for the present context in the corresponding contextual bandit instance. Yet, we show that if one plays an action $\arg\max_{\tilde{a} \in \mathcal{A}_t} \langle \tilde{a}, \theta \rangle$ that is optimal with respect to θ, then the contextual bandit instance generates a *linear reward* with respect to $g(\theta)$. Moreover, the linear bandit instances share the same optimal parameter $\theta_\star$ as in the contextual instance. Under standard boundedness assumptions of the contexts and actions, the reward noise in the reduced linear instance and the contextual bandit instance also share a similar sub-Gaussian tail. By mapping the linear bandit action $g(\theta)$ to the contextual bandit $\arg\max_{\tilde{a} \in \mathcal{A}_t} \langle \tilde{a}, \theta \rangle$ for any context $\mathcal{A}_t$ and θ, any algorithm for the linear bandit problem can be immediately applied to solve for the contextual bandit problem and suffer no additional regret in the worst-case.

The reduction becomes more challenging when the context distribution is unknown. One idea is to estimate $g(\theta)$ for all possible θ. Unfortunately, doing so would require a large number of samples resulting in unbounded regret. We resolve this issue by a batched approach where the batches provide increasingly better estimates of $g(\theta)$. In each batch, we estimate $g(\theta)$ using all the contexts generated in the previous batch. Note that this inevitably introduces error which ruins the linearity of the collected reward from the contextual bandit instance. Hence, we can only apply algorithms that are designed for misspecified linear bandits. Luckily, with a carefully designed batch sequence, we show that a linear bandit algorithm that works for all misspecification levels $\epsilon \in [1/\sqrt{T}, 1]$ can be applied to solve the contextual bandit instance. As it is hard to guarantee a good estimate of g for all θ, we restrict our attention to a finite subset of the unknown parameter set Θ (discretization) that is guaranteed to contain a good action. The amount of misspecification is bounded using a union bound argument over the discretization of Θ. While the discretization of Θ may eliminate the optimal arm and the function g is shown to be non-smooth, we show that the function $r(\theta) = \langle g(\theta), \theta_\star \rangle$ is smooth on a neighborhood of $\theta_\star$. This is sufficient to show that if discretized finely, the discrete set will contain a good arm. The final worst-case regret bound can be controlled by the regret bound of the linear bandit algorithm and the batch lengths.

We next provide a high level explanation on why our reduction enables **tighter regret bounds** for contextual linear bandits with stochastic contexts. Using the Phased Elimination algorithm (PE) [LSW20] with modified confidence intervals to solve the misspecified linear bandit instance, we prove a high probability bound of $O(d\sqrt{T \log T})$ on the regret of the contextual bandit problem using our reduction. Existing analysis techniques for changing action sets suffer $\log$ factors due to several reasons, such as bounding the radius of an ellipsoid that contains the unknown parameter with high probability, bounding the regret by the radius of the ellipsoid, or summing the regret over multiple episodes; in contrast, the $\log$ factors in our analysis appear only from a union bound used to prove concentration of the estimated arm means with high probability.

The reduction also implies improved regret bounds for **misspecified contextual linear bandits** by leveraging the optimal and nearly optimal regret bounds proved in the literature [LSW20] for fixed action sets. This results in high probability bounds; as opposed to the best known results for changing action sets that add $\log$ factors and only hold in expectation. As the reduction introduces a small $\tilde{O}(1/\sqrt{T})$ misspecification, this adds to the possibly unknown misspecification ϵ. There is a little subtlety here; some algorithms adapt to known ϵ better than the unknown case. To avoid adding extra factors to the regret bounds by sub-optimal adaptation to the known $\tilde{O}(1/\sqrt{T})$ part of misspecification, we slightly modify existing algorithms to account for this. This is done by increasing the number of times an arm is explored by a constant factor. Similarly, our results allow to carry over the better regret bounds for linear bandits with **adversarial corruption** to the contextual setting. This is achieved by modifying the algorithms for linear bandits with adversarial corruption to account for the known misspecification added by our reduction.

Our reduction has a byproduct; it limits the size of available actions in each round to the minimum between $|\mathcal{A}|$ and the size of the parameter set Θ (recall that we construct an action $g(\theta)$ for each $\theta \in \Theta$). While in general a discretization of size $T^{\Omega(d)}$ is required to guarantee a good action in the discrete set, if the unknown parameter follows some structure, a discretization of smaller size can be sufficient. For example, for contextual linear bandits with s-**sparse unknown parameter**, we show that a discretization of size $T^{O(s)}$ is sufficient. This directly implies a high probability $O(\sqrt{dsT\log T})$ regret bound as opposed to the best known $O(\sqrt{dsT}\log T)$ bound [APS12] that loses extra $\log$ factors due to solving a linear regression problem over the space of sparse unknown parameters.

As it is enough to make only batch updates to our estimates of the actions $g(\theta)$, our algorithm can be modified to provide improved regret bounds for contextual linear bandits with $O(\log\log T)$ **batch**es. We use batch lengths that were introduced in [GHR19] which grow as $T^{1-2^{-m}}$. In each batch, as action set is fixed, we utilize elimination algorithms with the G-optimal design. However, at batch m, the gaps of sub-optimal actions depend on the confidence of our estimates in batch $m-1$, which rely on the G-optimal design using the estimates of g from batch $m-2$. As a result, the regret in batch m can be at most the ratio between the length of batch m and batch $m-2$. Considering the growth rate of the batch lengths, this ratio can be large, especially in the first few batches. To fix this, we modify the batch lengths to grow in length only at the batches with odd index, while even batches use the same length as the previous batch. Our algorithm results in an $O(d\sqrt{T\log T}\log\log T)$ regret bound using $O(\log\log T)$ batches, improving a $\sqrt{\log d}$ factor (that appeared due to the distributional G-optimal design proposed in [RYZ21]) in the regret bound over the best known result [RYZ21]. Our result also provides the first high probability bound under the $O(\log\log T)$ batches limitation.

As a consequence of the batch learning result and the fact that our reduction can limit the action set based on the size of Θ, we provide the first algorithm with $O(\log\log T)$ batches for the sparse setting with a regret bounded by $O(\sqrt{dsT\log T}\log\log T)$ with high probability, where s is the sparsity parameter.

Chapter Organization

Section 2.3 describes our setup and reviews notation; Section 2.4 describes our reduction and main theorems; App. A.1 describes a spacial case of the reduction; and Section 2.5 uses our reduction to prove improved regret bounds for a number of stochastic linear bandit problems.

2.3 Model and Notations

Notation. We use the following notation throughout the chapter. For a vector X we use X_i to denote the i-th element of the vector X. The set $\{1, ..., i\}$ for $i \in \mathbb{N}, i > 0$ is denoted by $[i]$, where $\mathbb{N}$ is the set of natural numbers. We say that $y = O(f(x))$ if there is x_0 and a constant c such that $y \leq cf(x)\ \forall x > x_0$; we also use $\tilde{O}(f(x))$ to omit $\log$ factors. A δ-net (with respect to norm-2) of a set $\mathcal{A} \subseteq \mathbb{R}^d$ for $\delta > 0$ is any set $\mathcal{B} \subseteq \mathbb{R}^d$ such that for every $a \in \mathcal{A}$, there exists $b \in \mathcal{B}$ with $\|a - b\|_2 \leq \delta$, where $\mathbb{R}$ is the set of real numbers. For a set of sets $\mathcal{S}$, $\cup\mathcal{S}$ denotes the union of all elements in $\mathcal{S}$. For a family of sets $\{\mathcal{A}_i\}_{i=1}^n$ we use $\prod_{i=1}^n \mathcal{A}_i = \{(a_1, ..., a_n)|a_i \in \mathcal{A}_i,\ \forall i \in [n]\}$ to denote the product set.

Contextual Linear Bandits. We consider a contextual linear bandit problem, where a learner interacts with an environment over a time horizon of length T. At time $t \in [T]$, the learner observes a set of admissible arms $\mathcal{A}_t$ representing the context, pulls an arm $a_t \in \mathcal{A}_t$, and receives a reward

$$r_t = \langle a_t, \theta_\star \rangle + \eta_t, \tag{2.1}$$

where the context $\mathcal{A}_t$ is generated from a distribution $\mathcal{D}$ (independently from other iterations), a_t is a function of the history $\mathcal{H}_t = \{\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_t\}$, $\theta_\star$ is an unknown parameter vector of dimension d, and η_t is a random noise. Here, the noise η_t follows an unknown distribution that satisfies $\mathbb{E}[\eta_t|\mathcal{F}_t] = 0$, $\mathbb{E}[\exp(\lambda\eta_t)|\mathcal{F}_t] \leq \exp(\lambda^2/2)\ \forall\lambda \in \mathbb{R}$ (sub-Gaussian), where $\mathcal{F}_t = \sigma\{\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_t, a_t\}$ is the filtration of all historic information up to time t, and $\sigma(X)$ is the σ-algebra generated by X. We follow the standard assumptions that $\theta_\star \in \Theta \subseteq \{\theta|\ \|\theta\|_2 \leq 1\}$, $\|a\|_2 \leq 1\ \forall a \in \mathcal{A}_t$ and $\forall t \in [T]$ almost surely. The learner adopts a policy π that maps the history up to time t, $(\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_{t-1}, a_{t-1}, r_{t-1}, \mathcal{A}_t)$, to a probability distribution over $\mathcal{A}_t$; we denote the policy $\pi(\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_{t-1}, a_{t-1}, r_{t-1}, \mathcal{A}_t)$ at time t by $\pi_t(\mathcal{A}_t)$. The goal of the learner is to minimize the regret defined as

$$R_T = \sum_{t=1}^{T} \max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle - \langle a_t, \theta_\star \rangle. \tag{2.2}$$

In the next sections we assume for simplicity that for each $\theta \in \Theta$, there is a unique $a_t \in \mathcal{A}_t$ that satisfies $\langle a_t, \theta \rangle = \sup_{a \in \mathcal{A}_t} \langle a, \theta \rangle$ almost surely. This is to avoid dealing with approximations and choice functions (if there are multiple optimal actions) in the description of our algorithms. However, our results do not need this assumption, please see App. A.2.1.

Batch Learning. In this setting, the learner is allowed to change the policy π_t only at M pre-specified time slots $1 \leq t^{(1)}, ..., t^{(M)} \leq T$, where M is the number of batches.

Misspecified Linear Bandits. Here pulling an action $a \in \mathcal{A}_t$ generates a reward perturbed as

$$r_t = \langle a, \theta_\star \rangle + \eta_t + f(a), \tag{2.3}$$

where f is unknown perturbation function, $\theta_\star$ is the unknown parameter vector, and η_t is a zero-mean noise that is 1-subgaussian conditioned on the history. The amount of deviation in the rewards is upper bounded by $\epsilon = \sup_{a \in \cup \text{Supp}(\mathcal{D})} |f(a)|$, and ϵ is called the amount of misspecification, where $\mathcal{D}$ is the context distribution and Supp($\mathcal{D}$) is the support set of $\mathcal{D}$.

Adversarial Corruption. We assume an adversary that operates as follows at each time t:

- The adversary observes the history of all past contexts $\mathcal{A}_1, ..., \mathcal{A}_{t-1}$, actions $a_1, ..., a_{t-1}$, rewards $r_1, ..., r_{t-1}$, and previously corrupted rewards $\tilde{r}_1, ..., \tilde{r}_{t-1}$, together with the current context $\mathcal{A}_t$.
- The adversary decides on a corruption function $c_t : \mathcal{A}_t \to \mathbb{R}$ that determines the amount of corruption for each action.
- The learner observes the history of contexts $\mathcal{A}_1, ..., \mathcal{A}_t$, actions $a_1, ..., a_{t-1}$, and previously corrupted rewards $\tilde{r}_1, ..., \tilde{r}_{t-1}$.
- The learner pulls arm a_t and observes reward $\tilde{r}_t = r_t + c_t(a_t)$, where $c_t(a_t)$ is the corruption provided by the adversary.

Note that the true reward r_t follows the linear bandits model, while the corrupted rewards do not need to. We assume that the amount of corruption the adversary can inflict is bounded as

$$\sum_{t=1}^{T} \sup_{a \in \mathcal{A}_t} |c_t(a)| \leq C, \tag{2.4}$$

where C is the maximum amount of corruption.

Sparsity. We here assume that the d-dimensional parameter vector $\theta_\star$ in (2.1) is sparse, namely $||\theta_\star||_0 \leq s$ for some known $s \in [d]$, where $||\theta_\star||_0$ denotes the norm-0 or cardinality of the vector $\theta_\star$.

2.4 Reduction from Stochastic Contextual to Linear Bandits

2.4.1 Reduction for Known Context Distribution $\mathcal{D}$

We construct a contextual linear bandit algorithm $\mathcal{M}$ that operates at a high level as follows. At each time t, the learner:

Step 1 (plays): observes a set of actions $\mathcal{A}_t$, uses θ_t (i.e., the current estimate of θ^*) to decide which action $a_t \in \mathcal{A}_t$ to play, and observes the associated reward r_t;

Step 2 (learns): calculates θ_{t+1}, i.e., an updated estimate of θ^*.

In our reduction, we use a single-context algorithm Λ for learning the parameter $\theta_\star$ in step 2, i.e., choose θ_{t+1}, and prove that we can achieve the same worst-case regret bound as Λ.

Fixed Action Space. We provide to the linear bandit algorithm Λ the fixed action space

$$\mathcal{X} = \{g(\theta) | \theta \in \Theta\}, \text{ where } g(\theta) = \mathbb{E}_{\mathcal{A}_t \sim \mathcal{D}}[\arg\max_{a \in \mathcal{A}_t} \langle a, \theta \rangle | \theta]. \tag{2.5}$$

That is, for each θ, we create an action $g(\theta)$ that is the expected best action (under the distribution of the context $\mathcal{D}$) with respect to the parameter θ. We illustrate using a simple example.

Example 1. Assume that we may observe one out of the two following action sets $\{[1], [-1]\}, \{[1]\}$ randomly with probability $1/2$. The function $g(\theta) : \mathbb{R} \to \mathbb{R}$ can be calculated as follows

$$g(\theta) = \begin{cases} 1 & \text{if } \theta \geq 0 \\ 0 & \text{if } \theta < 0 \end{cases} \tag{2.6}$$

and thus $\mathcal{X} = \{[0], [1]\}$.

Reduction Algorithm. The algorithm $\mathcal{M}$ proceeds at each time t as follows:

1. The single context algorithm Λ decides to play an action $x_t \in \mathcal{X}$, using the history Λ observed. This action is never actually played. Instead, $\mathcal{M}$ observes what action x_t that Λ selected, and uses a θ_t with $x_t = g(\theta_t)$ as its current estimate of $\theta_\star$, where ties are broken arbitrarily.

2. $\mathcal{M}$ observes $\mathcal{A}_t$, plays the action $a_t = \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle$ and receives reward r_t. It provides this reward to Λ.

3. Λ assumes that the reward r_t it received was generated according to the linear bandit model $r_t = \langle g(\theta_t), \theta_\star \rangle + \eta'_t$, and adds the action-reward pair $(g(\theta_t), r_t)$ to its history.

Note that the set of actions $\mathcal{X}$ we created contains actions that may not be part of the original sets $\mathcal{A}_t$'s; this is fine, since these actions are actually never played - they are used to simulate an environment that enables Λ to correctly update its estimate of $\theta_\star$. That is, although all actions played come from the eligible sets $\mathcal{A}_t$, all learning (updates on θ_t) is derived from the single context algorithm Λ that never explicitly learns $\mathcal{A}_t$. We also highlight that

Theorem 1. *Let Λ be any algorithm for linear bandits and I be a contextual linear bandit instance with stochastic contexts, unknown parameter $\theta_\star$ and rewards r_t generated as described in the reduction algorithm described above. It holds that*

- *The reward r_t is generated, by pulling the arm $g(\theta_t)$, from a linear bandit instance L with action set $\mathcal{X}$, and unknown parameter $\theta_\star$.*
- *The reduction results in an algorithm $\mathcal{M}$ for contextual linear bandits such that with probability at least $1-\delta$ we have*

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| \leq c\sqrt{T\log(1/\delta)}, \tag{2.7}$$

  where $R_T^{\Lambda}(L)$ is the regret of Λ over the constructed linear bandit instance L, $R_T^{\mathcal{M}}(I)$ is the regret of algorithm $\mathcal{M}$ over the instance I and c is a universal constant.

Proof Outline. The complete proof is provided in App. A; we here provide a brief outline. The basic idea is to show that the action taken by algorithm $\mathcal{M}$ at time t is an unbiased estimate of $g(\theta_t)$, then decompose $R_T^{\mathcal{M}}(I)$ as

$$\begin{aligned} R_T^{\mathcal{M}}(I) = \sum_{t=1}^{T} &\langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta_\star\rangle, \theta_\star\rangle - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta_\star\rangle, \theta_\star\rangle] \\ &+ \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta_t\rangle, \theta_\star\rangle|\theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta_t\rangle, \theta_\star\rangle + \langle g(\theta_\star),\theta_\star\rangle - \langle g(\theta_t),\theta_\star\rangle, \end{aligned} \tag{2.8}$$

where the expectation is with respect to the randomness in the context generation $\mathcal{A}_t$. We then prove and use results such as $\langle g(\theta'), \theta' \rangle = \max_{\theta\in\Theta} \langle g(\theta), \theta' \rangle, \ \forall\, \theta' \in \Theta$, to arrive at

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| \leq |\sum_{t=1}^{T} \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle]| \\ + |\sum_{t=1}^{T} \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle | \theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle|, \tag{2.9}$$

Next, we show that the quantity

$$\Sigma_T := \sum_{t=1}^{T} \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle | \theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle$$

is a martingale (using that a_t is unbiased estimate of $g(\theta_t)$) with a bounded difference (by boundedness of $\Theta, \mathcal{A}_t$) and apply the Azuma–Hoeffding inequality to bound these terms with high probability. To prove that the rewards r_t come from a linear bandit instance L, we finally show that the reward r_t can be expressed as $r_t = \langle g(\theta_t), \theta_\star \rangle + \eta'_t$, where η'_t is a zero mean 1-sub-Gaussian noise conditioned on the filtration of historic information $\theta_1, ..., \theta_t$ and rewards $r_1, ..., r_{t-1}$ of the instance L. $\square$

Algorithm 1 Reduction from stochastic contexts to no context

```
1: Input: confidence parameter $\delta$, phase lengths $\{t^{(m)}\}_{m=1}^{M+1}$, and algorithm $\Lambda_\epsilon$ for linear contextual
   bandits with $\epsilon$ misspecification.
2: Initialize: $g^{(1)} : \Theta' \to \mathbb{R}^d$ randomly, $\epsilon_1 = 1$, and let $\mathcal{X}_1 = \{g^{(1)}(\theta) | \theta \in \Theta'\}$.
3:
4: for $m = 1 : M$ do
5:     for $t = t^{(m)} + 1, ..., t^{(m+1)}$ do
6:         Let $g^{(m)}(\theta_t) \in \mathcal{X}_m$ be the arm selected by $\Lambda_{\epsilon_m}$ after observing rewards $r_{t^{(m)}+1}, ..., r_{t-1}$.
7:         Play arm $a_t = \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t \rangle$ and receive reward $r_t$. Provide $r_t$ to $\Lambda_{\epsilon_m}$.
8:     Update: $g^{(m+1)}(\theta) = \frac{1}{t^{(m+1)}} \sum_{t=1}^{t^{(m+1)}} \arg\max_{a\in\mathcal{A}_t} \langle a, \theta \rangle$, $\mathcal{X}_{m+1} = \{g^{(m+1)}(\theta) | \theta \in \Theta'\}$, and
   $\epsilon_{m+1} = 2\sqrt{\log(M|\Theta'|/\delta)/t^{(m+1)}}$.
```

2.4.2 Reduction for Unknown Context Distribution

As described in (2.5), calculating the function $g(\theta)$ requires knowledge of the distribution $\mathcal{D}$. We do not really need this knowledge; we prove it is sufficient to use empirical estimation of $g(\theta)$. As a result, we prove that any stochastic linear contextual bandit instance, even for unknown context distributions, can be reduced to a linear bandit instance albeit with a small misspecification.

Our basic approach follows the reduction in Section 2.4.1 but uses a sequence of functions $g^{(m)}$ that approximate g increasingly well (as m increases). To do so, as it is hard to guarantee a good estimate of g for all θ, we restrict our attention to a finite subset of Θ that is large enough to include a good action. In particular, instead of considering actions in a continuous space Θ, we only consider a finite subset of actions $\Theta' \subseteq \Theta$, where Θ' is an $1/T$-net for Θ according to the norm-2 distance. We divide the time horizon T into M epochs, each of duration $T_m = t^{(m+1)} - t^{(m)}$, $m = 1 \dots M$. For each epoch, we construct an empirical estimate of g as $g^{(m)}(\theta) = \frac{1}{t^{(m)}} \sum_{t=1}^{t^{(m)}} (\arg\max_{a \in \mathcal{A}_t} \langle a, \theta \rangle)$, and calculate the set of actions $\mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}$. We then use at each epoch a single context algorithm as before, but we now provide at epoch m the fixed set of actions $\mathcal{X}_m$ for the algorithm to choose from. As a result, the regret of the linear bandit problem is defined as

$$R_T^{\Lambda_\epsilon} = \sum_{t=1}^{T} \max_{\theta \in \Theta'} \langle g(\theta), \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle. \tag{2.10}$$

Our algorithm relies on Λ_ϵ, an algorithm for linear bandits with ϵ misspecification. The misspecification reflects our confidence in our estimate of the function g, hence, decreases each epoch. We start with a large value of the misspecification parameter $\epsilon_1 = 1$ and a random initialization of the function g (which we cal $g^{(1)}$), and hence random initialization of $\mathcal{X}$ denoted $\mathcal{X}_1$. At time slot t of epoch m, the algorithm asks Λ_{ϵ_m} for an action to play $g^{(m)}(\theta_t) \in \mathcal{X}_m$ given the history of action and rewards $\{(g^{(m)}(\theta_i), r_i)\}_{i=t^{(m)}+1}^{t-1}$ in epoch m only. The algorithm pulls the action $a_t = \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle$ and observes r_t. The reward r_t is then passed to the algorithm Λ_{ϵ_m}. At the end of each epoch, the misspecification parameter, estimates $g^{(m)}$, and action set $\mathcal{X}_m$ are updated. The pseudo-code of our reduction is provided in Algorithm 1, and the proof in App. B. To achieve nearly optimal regret bounds, the misspecification ϵ need to be $\tilde{O}(1/\sqrt{T})$

(recall the $\Omega(\sqrt{d}\epsilon T)$ regret lower bound [LSW20]). Attempting to use $M = 2$ to first estimate g and then learn in the second epoch, would require the length of the first epoch to be $\Omega(T)$, to ensure the $\tilde{O}(1/\sqrt{T})$ misspecification, resulting in linear regret. Instead, as we clarify next, we use exponentially increasing epoch lengths to mix the learning with a gradual estimation of g resulting in misspecification that is effectively $\tilde{O}(1/\sqrt{T})$.

Theorem 2. *Let Λ_ϵ be an algorithm for linear bandits with ϵ misspecification and I be a contextual linear bandit instance with stochastic contexts, unknown parameter $\theta_\star$ and rewards r_t are generated as described in Algorithm 1. The following holds:*

- *Conditioned on $\mathcal{H}_{t^{(m)}} = \sigma(\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_{t^{(m)}}, a_{t^{(m)}}, r_{t^{(m)}})$: in epoch m, the rewards r_t are generated, by pulling arm $g^{(m)}(\theta_t)$, from a misspecified linear bandit instance L_m for $t = t^{(m)} + 1, ..., t^{(m+1)}$, action set $\mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}$, unknown parameter $\theta_\star$, mean rewards $\langle g(\theta), \theta_\star \rangle$, and unknown misspecification ϵ'_m.*
- *The misspecification ϵ'_m is bounded by the known quantity ϵ_m with probability at least $1 - c\delta/M$.*
- *With probability at least $1 - \delta$ we have that $|R_T(I) - \sum_{m=1}^{M} R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)| \leq c\sqrt{T \log(1/\delta)}$, where $R_T(I)$ is the regret of Algorithm 1 over the instance I, $R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$ is the regret of algorithm Λ_{ϵ_m} over the bandit instance L_m in phase m, $T_m = t^{(m+1)} - t^{(m)}$, and c is a universal constant.*

As a consequence, we prove the following corollary in Section 2.5.1.

Corollary 1. *For Algorithm 1 with $t^{(m)} = 2^{m-1}$ and Λ_ϵ to be PE with modified confidence intervals [LSW20], it holds that with probability at least $1 - c\delta$ we have that $R_T = O(d\sqrt{T \log(T/\delta)})$.*

In App. A.1 we describe an important special case where the reduction can be performed without approximating g.

2.4.3 Computational Complexity

In this subsection, we show that our reduction can be made *computationally-efficient* with the help of optimization oracles and linear regression oracles [AK08, ZFL22], which are standard assumptions when computation is a concern in bandit problems.

In particular, we are able to

- Efficiently construct a linear optimization oracle for the linear bandit instance L_m given a linear optimization oracle for the original contextual bandit instance I. Together with the linear regression oracle, this suffices to construct the linear bandit policy [AK08, DHK08, BCK12, HK16, IHS19].
- Efficiently compute an inverse of the function $g^{(m)}$, which is performed in our reduction to translate the linear bandit action to an action in the original instance.

Recall that a linear optimization oracle of the contextual bandit instance states that $\arg\max_{a\in\mathcal{A}_t}\langle a,\theta\rangle$ can be efficiently solved for any context $\mathcal{A}_t$. However, in the reduced bandit instance, we are solving an optimization problem related to the reduced arms, $\mathcal{X}_m = \{g^{(m)}(\theta)|\theta\in\Theta'\}$. In Lemma 1 in the following subsection, we provide an efficient (approximate) optimization oracle for solving $\arg\max_{x\in\mathcal{X}_m}\langle x,\theta\rangle$ by applying a careful discretization of the set Θ. By storing $g^{(m)}(\theta)$ for all the arms pulled by the linear bandit algorithm, we can then apply the (efficient) regression oracle for the original contextual bandit instance to get a regression oracle in the reduced instance.

If all actions played by the linear bandit algorithm are the output of the linear optimization oracle for some θ [AK08, DHK08, IHS19], then from Lemma 1 in the following subsection, the inversion of $g^{(m)}(\theta)$ for the actions pulled by the linear bandit algorithm can be performed by storing θ whenever the corresponding action $g^{(m)}(\theta)$ is stored. This increases both the space and time complexity only by a constant factor.

2.4.4 Constructing the Optimization Oracle

In this subsection, we construct an efficient approximate linear optimization oracle over the set $\mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}$, defined in Algorithm 1. This construction relies on a linear optimization oracle over the sets $\mathcal{A}_t$, which we define next.

Definition 1. A **linear optimization oracle** for a set $\mathcal{A}$ is a function $\mathcal{O}(\mathcal{A}; .)$ which takes as input $\theta \in \{a \in \mathbb{R}^d | \|a\|_2 \leq 1\}$ and outputs $\mathcal{O}(\mathcal{A}; \theta) \in \mathcal{A}$ with $\langle \mathcal{O}(\mathcal{A}; \theta), \theta \rangle = \sup_{a \in \mathcal{A}} \langle a, \theta \rangle$.

We consider the following discretization of the set Θ that simplifies the construction of the linear optimization oracle over $\mathcal{X}_m$. The set Θ is discretized to $\Theta' = \{[\theta]_q | \theta \in \Theta\}$, where $[\theta]_q = q\lfloor \theta\sqrt{d}/q \rfloor / \sqrt{d}$ and q is the discretization parameter. Note that for $\Theta \subseteq \{a \in \mathbb{R}^d | \|a\|_2 \leq 1\}$, the size of the set Θ' can be bounded as $|\Theta'| \leq Cd(1/q)^{4d+2}$, where C is a universal constant. The following lemma shows that $g^{(m)}([\theta/\|\theta\|_2]_q)$ for sufficiently small q can be used as an approximate linear optimization oracle over the set $\mathcal{X}_m$. We observe that $g^{(m)}$ can be calculated by invoking the linear optimization oracle of the contextual bandit instance at most T times.

Lemma 1. Consider a given $m \in [M]$ and let $g^{(m)}(\theta) = \frac{1}{t^{(m)}} \sum_{t=1}^{t^{(m)}} \arg\max_{a \in \mathcal{A}_t} \langle a, \theta \rangle$, $\mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}$, where $t^{(m)}$ is the length of phase m and $\Theta' = \{[\theta]_q | \theta \in \Theta\}$ is a discretization of Θ, $[\theta]_q = q\lfloor \theta\sqrt{d}/q \rfloor / \sqrt{d}$ and q is the discretization parameter. For any $\theta \in \{a \in \mathbb{R}^d | \|a\|_2 \leq 1\}$, $\epsilon \in \mathbb{R}^+$, if $q \leq \epsilon/2$, we have that

$$\langle g^{(m)}([\theta/\|\theta\|_2]_q), \theta \rangle \geq \sup_{a \in \mathcal{X}_m} \langle a, \theta \rangle - \epsilon \|\theta\|_2. \tag{2.11}$$

The proof is provided in App. A.4

2.5 Implications

2.5.1 Tighter Regret Bound for Contextual Linear Bandits

We here show that our reduction leads to the first $O(d\sqrt{T \log T})$ high probability upper bound for linear bandits with changing action sets. We recall that in all our results, we only consider contextual linear bandits with unknown context distribution as described in Section 2.3. We rely on the Phased Elimination (PE) [LSW20] as our linear bandit algorithm Λ_ϵ. PE is the same as Algorithm 2, that we will use next in the batched setting, except that $\mathcal{X}, g$ are fixed (recall that we apply PE within an epoch that fixes the estimate of g). The parameter γ_m in Algorithm 2 is called the confidence interval; we will specify its value in our theorems. In Algorithm 1, setting $t^{(m)} = 2^{m-1}$ and Λ_ϵ to be PE with modified confidence intervals [LSW20] to account for the misspecification we get the following corollary.

Corollary 2. *For Algorithm 1 with $t^{(m)} = 2^{m-1}$ and Λ_{ϵ_m} to be PE with $\gamma_m = 6\sqrt{d\log(T|\Theta'|/\delta)/t^{(m)}}$ it holds that with probability at least $1 - c\delta$ we have that $R_T \leq c\sqrt{dT\log(T|\Theta'|/\delta)}$.*

Proof. Let the length of phase m be $T_m = t^{(m+1)} - t^{(m)} = 2^{m-1}$. Conditioned on the event that the misspecification in phase m is bounded by ϵ_m, PE with modified confidence intervals achieves a regret $R_{T_m}^{\Lambda_{\epsilon_m}}$ that is upper bounded by $O(\sqrt{dT_m\log(T_mM|\Theta'|/\delta)} + \sqrt{d}T_m\epsilon_m)$ with probability at least $1 - \delta/M$. Hence, by Proposition 3 and the union bound we have that it holds with probability at least $1 - c\delta$ that

$$R_{T_m}^{\Lambda_{\epsilon_m}} \leq c(\sqrt{dT_m\log(T|\Theta'|/\delta)} + \sqrt{d}T_m\epsilon_m)\forall m \in [M], \tag{2.12}$$

where we used the fact that $T_m \leq T, M \leq T$. Substituting the value of ϵ_m we get that the following holds with probability at least $1 - c\delta$

$$\begin{aligned} R_{T_m}^{\Lambda_{\epsilon_m}} &\leq c(\sqrt{dT_m\log(T|\Theta'|/\delta)} + \sqrt{d}T_m\sqrt{\frac{\log(M|\Theta'|/\delta)}{t^{(m)}}}) \\ &= c(\sqrt{dT_m\log(T|\Theta'|/\delta)} + \sqrt{dT_m\log(M|\Theta'|/\delta)})\forall m \in [M]. \end{aligned} \tag{2.13}$$

Hence, using $M \leq T$, we get that with probability at least $1 - c\delta$ we have

$$R_{T_m}^{\Lambda_{\epsilon_m}} \leq c\sqrt{dT_m\log(T|\Theta'|/\delta)} \tag{2.14}$$

Substituting in Theorem 2, we get that with probability at least $1 - c\delta$ it holds that

$$\begin{aligned}
R_T &\le c\sqrt{T\log T} + c\sqrt{d\log(T|\Theta'|/\delta)}\sum_{m=1}^{\log T}\sqrt{T_m} \\
&\le c'\sqrt{dT\log(T|\Theta'|/\delta)}\sum_{m=1}^{\log T}\sqrt{2^{m-\log T}} \\
&\le c'\sqrt{dT\log(T|\Theta'|/\delta)}\sum_{i=0}^{\infty}\sqrt{2^{-i}} \\
&\le \frac{c'}{1-1/\sqrt{2}}\sqrt{dT\log(T|\Theta'|/\delta)}. \qquad (2.15)
\end{aligned}$$

□

It is well known that if $\Theta \subseteq \{a \in \mathbb{R}^d | \|a\|_2 \le 1\}$, then there is $1/T$-net of Θ, Θ', such that $|\Theta'| \le (6T)^d, \Theta' \subseteq \Theta$. This directly implies that $R_T = O(d\sqrt{T\log(T)})$ with probability at least $1 - 1/T$. This improves a factor of $\sqrt{\log T}\log\log T$ over [LWC21] and a factor of $\sqrt{\log T}$ over [APS11].

2.5.2 Batch Learning

We here show that our reduction can be applied to improve the result of [RYZ21] for contextual linear bandits with stochastic contexts by providing a regret upper bound of $O(d\sqrt{T\log T}\log\log T)$ with high probability as opposed to the *in expectation* $O(d\sqrt{T\log d\log T}\log\log T)$ regret bound in [RYZ21]. This can be achieved by replacing Λ_{ϵ_m} with the G-optimal design policy constructed using $\mathcal{X}_m$. To compute $t^{(i)}$ we first define

$$u_m = T^{1-2^{-m}}, m = 1, ..., M/2. \qquad (2.16)$$

We let

$$t^{(m)} = \lfloor u_{m//2+1} \rfloor \forall m \in [M], t^{(M+1)} = T. \qquad (2.17)$$

where // denotes integer division. For completeness, we include the pseudo-code in Algorithm 2. The following result follows using Theorem 2.

Theorem 3. *For Algorithm 1 with $M = 2\log\log T$ (corresponds to $2\log\log T + 1$ batches), $t^{(m)}$ given in* (2.17) *and Λ_{ϵ_m} replaced by the G-optimal design policy constructed using $\mathcal{X}_m$, it holds that there is a constant c such that with probability at least $1 - c\delta$, we have*

$$R_T \leq c\sqrt{dT\log(\frac{M|\Theta'|}{\delta})}\log\log T. \tag{2.18}$$

Proof. We first notice that for $M = \log\log T$, we have that $u_{m+1}/\sqrt{u_m} = \sqrt{T}\forall m \in [M-1]$, $u_{M+1}/\sqrt{u_M} = \sqrt{2T}$. Hence, we have that

$$u_{m+1}/\sqrt{u_m} \leq \sqrt{2T}\forall m \in [M]. \tag{2.19}$$

We also have that $u_{m+1} \geq u_m \;\forall m \geq 1$ (since $u_m \leq T$). The proof follows from the properties of the G-optimal design together with the properties of $g^{(m)}$ in the proof of Theorem 2. The G-optimal design ensures that for any $\theta \in \Theta'$ the following holds with probability at least $1 - \delta$

$$|\langle g^{(m)}(\theta), \hat{\theta}_m - \theta_\star\rangle| \leq 2\epsilon'_m\sqrt{d} + \sqrt{\frac{4d}{T_m}\log 1/\delta}, \tag{2.20}$$

where $\epsilon'_m = \sup_{\theta\in\Theta'}|\langle g^{(m)}(\theta) - g(\theta), \theta_\star\rangle|$. By the triangle inequality, we have that

$$\begin{aligned}|\langle g^{(m)}(\theta), \hat{\theta}_m\rangle - \langle g(\theta), \theta_\star\rangle| &\leq 2\epsilon'_m\sqrt{d} + \sqrt{\frac{4d}{T_m}\log 1/\delta} + |\langle g^{(m)}(\theta) - g(\theta), \theta_\star\rangle| \\ &\leq \epsilon'_m(2\sqrt{d}+1) + \sqrt{\frac{4d}{T_m}\log 1/\delta} \end{aligned} \tag{2.21}$$

By Proposition 3 we have that $\epsilon'_m \leq \epsilon_m\forall m \in [M]$ with probability at least $1-\delta$. Hence, by the union bound we have that the following holds with probability at least $1 - \delta$

$$|\langle g^{(m)}(\theta), \hat{\theta}_m\rangle - \langle g(\theta), \theta_\star\rangle| \leq \epsilon_m(2\sqrt{d}+1) + \sqrt{\frac{4d}{T_m}\log(\frac{M|\Theta'|}{\delta})}\forall\theta \in \Theta'\forall m \in [M]. \tag{2.22}$$

Hence, with probability at least $1-\delta$, the best arm is not eliminated and the arms that are not eliminated at the end of batch m, will have a gap that is at most twice the confidence interval in (2.22), otherwise, they satisfy the elimination criterion with $\theta_\star$. Hence, the sum regret $\sum_{m=1}^{M} R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$ is bounded as follows with probability at least $1-\delta$

$$\begin{aligned}
\sum_{m=1}^{M} R_{T_m}^{\Lambda_{\epsilon_m}}(L_m) &\leq \sqrt{T} + c'\sum_{m=1}^{M}\epsilon_m(2\sqrt{d}+1)T_{m+1} + c'\sqrt{\frac{4d}{T_m}\log(\frac{M|\Theta'|}{\delta})}T_{m+1} \\
&= \sqrt{T} + c\sqrt{d\log(\frac{M|\Theta'|}{\delta})}\sum_{m=1}^{M}\frac{T_{m+1}}{\sqrt{T_{m-1}}} + \frac{T_{m+1}}{\sqrt{T_m}} \\
&\overset{(i)}{\leq} \sqrt{T} + 2c\sqrt{d\log(\frac{M|\Theta'|}{\delta})}\sum_{m=1}^{M}\frac{T_{m+1}}{\sqrt{T_{m-1}}} + \sqrt{T} \\
&\overset{(ii)}{\leq} \sqrt{T} + 2c\sqrt{d\log(\frac{M|\Theta'|}{\delta})}\sum_{m=1}^{M}\sqrt{T} + \sqrt{T} \\
&\leq c''\sqrt{dT\log(\frac{M|\Theta'|}{\delta})}\log\log T, \qquad (2.23)
\end{aligned}$$

where (i) uses (2.19) and the fact that either $T_{m+1} \leq \sqrt{2T_m}$ or $T_{m+1} = T_m \leq \sqrt{2TT_m}$, and (ii) follows from the fact that either ($T_m = T_{m-1}$ and $T_{m+1} \leq \sqrt{2TT_m}$) or ($T_m \leq \sqrt{2TT_{m-1}}$ and $T_{m+1} = T_m$), hence, in both cases we have $T_{m+1} \leq \sqrt{2TT_{m-1}}$. The results follow by Theorem 2 and the union bound. $\square$

As in Section 2.5.1, this implies that $R_T = O(d\sqrt{T\log(T)}\log\log T)$ with probability at least $1 - 1/T$, which strengthens the result in [RYZ21] to high probability result and also improves a factor of $\log d$ in the regret bound. Moreover, as will be seen in the following appendices, if $\theta_\star$ is s-sparse, Theorem 3 implies a regret bound of $O(\sqrt{dsT\log T}\log\log T)$ with probability at least $1 - 1/T$. To the best of our knowledge, this is the first nearly optimal algorithm for sparse unknown parameters that uses $O(\log\log T)$ batches.

2.5.3 Misspecified Contextual Linear Bandits

Our reduction framework can be used to improve regret bounds for misspecified contextual linear bandits. By noticing that for ϵ misspecified contextual linear bandits, the total amount of misspecification in epoch m is bounded by $\epsilon_m + \epsilon$ with high probability, the following result follows directly from [LSW20].

Algorithm 2 Batched algorithm for linear bandits with stochastic context

```
1: Input: confidence parameter $\delta$, and phase lengths $\{t^{(m)}\}_{m=1}^{M}$.
2: Initialize: $g^{(1)} : \Theta' \to \mathbb{R}^d$ randomly, $\Theta_1 = \Theta'$, and let $\mathcal{X}_1 = \{g^{(1)}(\theta)|\theta \in \Theta'\}$.
3: for $m = 1 : M$ do
4:     Find design $\rho : \mathcal{X}_m \to [0,1]$ with $\max_{a\in\text{Supp}(\rho)} \|a\|^2_{G^{-1}(\rho)} \le 2d$, $|\text{Supp}(\rho)| \le 4d\log\log d + 16$, where $G(\rho) = \sum_{a\in\text{Supp}(\rho)} \rho(a)aa^T$.
5:     Compute $u(x) = \lceil \rho(x)T_m \rceil$ and $u = \sum_{x\in\text{Supp}(\rho)} u(x)$.
6:     Use the policy described by $a_t = \arg\max_{a\in\mathcal{A}_t} \langle a, \theta\rangle$ for $u(g^{(m)}(\theta))$ times for each $\theta \in \Theta_m$ with $g^{(m)}(\theta) \in \text{Supp}(\rho)$.
7:     $\hat{\theta}_m = (\sum_{g\in\text{Supp}(\rho)} u(g)gg^T)^{-1} \sum_{i=1}^{u} r_i g^{(m)}(\theta_i)$.
8:     Update: $\Theta_{m+1} = \left\{\theta \in \Theta_m | \max_{\theta'\in\Theta_m} \langle \hat{\theta}_m, g^{(m)}(\theta') - g^{(m)}(\theta)\rangle \le \gamma_m \right\}$, $\gamma_m = 10\sqrt{\frac{d}{T_{m-1}} \log(M|\Theta'|/\delta)}$.
9:     Update: $g^{(m+1)}(\theta) = \frac{1}{t^{(m+1)}} \sum_{t=1}^{t^{(m+1)}} \arg\max_{a\in\mathcal{A}_t} \langle a, \theta\rangle$, and $\mathcal{X}_{m+1} = \{g^{(m+1)}(\theta)|\theta \in \Theta'\}$.
```

Theorem 4. *For contextual linear bandits with known misspecification that is bounded by ϵ, Algorithm 1 with Λ_{ϵ_m} being PE with $\gamma_m = 6d\sqrt{\log(T)/T_m} + \epsilon\sqrt{d}$ achieves a regret bound*

$$R_T \le c(d\sqrt{T\log(T/\delta)} + \epsilon\sqrt{d}T) \tag{2.24}$$

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

If ϵ is unknown, [LSW20] showed that PE achieves a regret upper bounded by $O(d\sqrt{T\log(T)} + \epsilon\sqrt{d}T\log T$ with high probability. We notice that the dependency on ϵ has an extra $\log T$ factor as compared to the known misspecification result. To avoid adding the $\log T$ factor in the term containing ϵ_m, we exploit the knowledge of ϵ_m to modify the confidence intervals in PE. We note that following the same analysis in [LSW20] by replacing ϵ with $\epsilon + \epsilon_m$ results in the same concentration of estimated means except for constant factors, which implies the following result.

Theorem 5. *For contextual linear bandits with unknown misspecification that is bounded by ϵ,* *Algorithm 1 with Λ_{ϵ_m} being PE with $\gamma_m = 6d\sqrt{\log(T)/T_m}$ achieves a regret bound*

$$R_T \leq c(d\sqrt{T\log(T/\delta)} + \epsilon\sqrt{d}T\log T) \tag{2.25}$$

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

Proof. The proof follows similar steps as in the proof of Proposition 5.1 in [LSW20] with different constants due to the change in γ_m. □

We note that as PE is performed in each epoch, the number of batches is $\Omega(\log^2 T)$. However, we constructed the algorithm this way only for simplicity. It is possible to perform PE once and update the estimates of g at the end of each batch similar to Algorithm 2.

2.5.4 Bandits with Adversarial Corruption

Our reduction directly extends the results for linear bandits with adversarial corruption to the contextual setting while maintaining the same regret bounds up to $\log$ factors. This leads to $\tilde{O}(d\sqrt{T} + d^{1.5}C)$ high probability regret bound as opposed to the best known $\tilde{O}(d^{4.5}\sqrt{T} + d^4C)$ regret bound. It was shown in [BLK21] that PE with modified confidence intervals achieves an $\tilde{O}(d\sqrt{T} + d^{1.5}C)$ high probability regret bound for unknown corruption C. A model selection based approach in [FGM20] that uses PE as a subroutine is shown to generalize this result for unknown C without changing the regret bound except for $\log$ factors. By adapting the confidence intervals of PE to account for the known ϵ_m misspecification the $\tilde{O}(d\sqrt{T} + d^{1.5}C)$ high probability regret bound extends for a model with $O(\sqrt{d\log T/T})$ misspecification and known corruption C. We will modify the PE confidence interval to be

$$\gamma_m = 8d\sqrt{\log(T)/t^{(m)}} + \frac{2C(4d\log\log d + 18)}{T_m}\sqrt{8d}. \tag{2.26}$$

The model selection approach in [FGM20] implies the following result.

Theorem 6. *For contextual linear bandits with unknown corruption that is bounded by C, Algorithm 1 with $t^{(m)} = 2^{m-1}$ and Λ_{ϵ_m} being G-COBE in [FGM20] with PE, that uses γ_m in* (2.26)*, as* *a subroutine, achieves*

$$R_T = \tilde{O}(d\sqrt{T} + d^{1.5}C) \tag{2.27}$$

with probability at least $1 - c/T$.

Proof. The proof follows by proving a regret bound for PE with the chosen γ_m for known C. Then use the model selection result in [FGM20] to prove a regret bound for unknown C and finally apply our main theorem to bound the final regret.

We notice that

$$|\langle g^{(m)}(\theta), \hat{\theta}_m\rangle - \langle g(\theta), \theta_\star\rangle| \le |\langle g^{(m)}(\theta), \hat{\theta}_m - \theta_\star\rangle| + |\langle g^{(m)}(\theta) - g(\theta), \theta_\star\rangle|. \tag{2.28}$$

Using Lemma 1 in [BLK21] and equation (2.20) to bound the first term and Proposition 3 to bound the second term, we obtain that $|\langle g^{(m)}(\theta), \hat{\theta}_m\rangle - \langle g(\theta), \theta_\star\rangle| \le \gamma_m/2$ with probability at least $1-c/T$. A standard calculation in Theorem 1 in [BLK21] implies a regret bound of $\tilde{O}(d\sqrt{T} + d^{1.5}C)$ with high probability. Hence, Theorem 4 in [WDZ22] implies a regret bound of $\tilde{O}(d\sqrt{T} + d^{1.5}C)$ with high probability for G-COBE with unknown C. Applying Theorem 2 and proceeding as in equation (2.15) concludes the proof. □

2.5.5 Sparsity

For linear contextual bandits with s-sparse unknown parameter, our reduction can be used to prove $O(\sqrt{dsT\log(T)})$ regret bound with high probability as opposed to the best known $O(\sqrt{dsT}\log^2 T)$ regret bound. It is not hard to show that there is $1/T$ cover with size at most $(6T)^{2s+1}$ in that case; also proved below for completeness. The following result directly follows from Corollary 2.

Theorem 7. *For Algorithm 1 with* $t^{(m)} = 2^{m-1}$ *and* Λ_{ϵ_m} *to be PE with* $\gamma_m = 6\sqrt{2ds\log(T/\delta)/t^{(m)}}$ *it holds that with probability at least* $1 - c\delta$ *we have that* $R_T = O(\sqrt{dsT\log(T/\delta)})$.

Proof. By Corollary 2, we only need to show that Θ contains an $1/T$-net with size at most $(6T)^{2s+1}$. We have that there is $1/2T$-net for $\{\phi \in \mathbb{R}^s | \|\phi\|_2 \le 1\}$ with size at most $(6T)^s$. To construct

s-sparse vectors in $\mathbb{R}^d$, there is $\sum_{i=1}^{s}\binom{d}{i}$ ways to choose the non-zero entries. This implies that the set $\{\theta \in \mathbb{R}^d | \|\theta\|_2 \leq 1, \|\theta\|_0 \leq s\}$ has $1/2T$-net $\mathcal{N}$ of size at most

$$|\mathcal{N}| \leq (6T)^s \sum_{i=1}^{s} \binom{d}{i} \leq (6T)^s \sum_{i=1}^{s} d^i \leq (6T)^s d^{s+1} \leq (6T)^{2s+1}. \tag{2.29}$$

We next construct an $1/T$-net that is subset of Θ (recall that this is required for Theorem 2). For every $x \in \mathcal{N}$, let $\mathcal{N}_x = \{\theta \in \Theta | \|\theta - x\| \leq 1/2T\}$. Let $\alpha : \mathcal{N} \to |\mathcal{N}|$ be an ordering of the set $\mathcal{N}$. And let $\mathcal{N}' \subseteq \mathcal{N}$ be defined as $\mathcal{N}' = \{x \in \mathcal{N} | \mathcal{N}_x \not\subseteq \cup_{y:\alpha(y) \leq \alpha(x)} \mathcal{N}_y\}$. Hence, $\{\mathcal{N}_x | x \in \mathcal{N}'\}$ is a set of pairwise disjoint sets. By the axiom of choice, there is a set $\mathcal{N}''$ such that $\mathcal{N}''$ contains exactly one element of each set $\mathcal{N}_x \forall x \in \mathcal{N}'$. By construction of $\mathcal{N}''$ we have that $\mathcal{N}'' \subseteq \Theta$.

We also have that for each $\theta \in \Theta$, there is $x \in \mathcal{N}$ with $\|x - \theta\|_2 \leq 1/2T$. Then for each $y \in \mathcal{N}_x$, we have by the triangle inequality that $\|y - \theta\|_2 \leq 1/T$. By the definition of $\mathcal{N}'$, we have that $\cup \mathcal{N}' = \cup_{z \in \mathcal{N}} \mathcal{N}_z$. Then, by construction of $\mathcal{N}''$, there is $z \in \mathcal{N}''$ such that $z \in \mathcal{N}_x$. Hence, $\|z - \theta\|_2 \leq 1/T$. This implies that $\mathcal{N}'' \subseteq \Theta$ is an $1/T$-net of Θ. We also have by construction that also that $|\mathcal{N}''| \leq |\mathcal{N}| \leq (6T)^{2s+1}$. $\square$

2.5.6 Structured Unknown Parameters

In some cases, the dimension d can be large but the unknown parameter is mapped from a small space of dimension s. In particular let $f : \mathbb{R}^s \to \mathbb{R}^d$ be such that $\Theta \subseteq \{f(\phi) | \|\phi\|_2 \leq 1\}$ and

$$\|f(\phi) - f(\phi')\|_2 \leq c\|\phi - \phi'\|_2, \tag{2.30}$$

where c is a universal constant. Finding an $1/T$-net for $\Phi = \{\phi | f(\phi) \in \Theta\}$ implies a c/T-net for Θ with the same size. Since $\Phi \subseteq \{\phi \in \mathbb{R}^s | \|\phi\|_2 \leq 1\}$, we have that there is an $1/T$-net for Φ that is contained in Φ with size at most $(6T)^s$. The following result immediately follows from Corollary 2.

Corollary 3. *Under the considered structured unknown parameters assumption, Algorithm 1 with $t^{(m)} = 2^{m-1}$ and Λ_{ϵ_m} to be PE with $\gamma_m = 6\sqrt{ds\log(T/\delta)/t^{(m)}}$ satisfies that with probability at least $1 - c\delta$ we have that $R_T = O(\sqrt{dsT\log(T/\delta)})$.*

2.6 Context Compression in Distributed Contextual Bandits

We consider a setup where a central learner wishes to solve a contextual linear bandit problem with the help of transient agents. That is, we assume that the agents are distributed and may not be present for the whole duration of learning. The central learner, through the information it keeps, could help passing by devices decide how to perform an action, for example: passing by drones decide how to perform a manoeuver; agricultural robots decide what amounts of substances such as pesticids to release; and passing by mobile devices decide which local restaurants to recommend.

A main challenge in these applications is the efficient communication of the context the agents experience. More specifically, each time an agent joins, she receives from the central learner information on the system, such as current estimates of the system parameters; she observes her current context, selects and plays an action and collects the corresponding reward. Although the distributed agent knows her context, the action she plays and the observed reward, the central learner does not - and needs this information to update its estimate of the system parameters. The context in particular can be communication heavy - in the examples we mentioned before, for drones the context could be their navigation capabilities, physical attributes, and enviromental factors such as wind speed; for agricultural robots, it could be images that indicate state of plants and sensor measurements such as of soil consistency; for restaurant recommendations, it could be the personal dietary preferences and restrictions, budget, and emotional state. Moreover, because of geographical separation, the central learner may not have any other way to learn the context beyond communication. Unlike the reward, that is usually a single scalar value, the context can be represented by a vector of a large dimension d from an infinite alphabet, and thus, communicating the context efficiently is heavily nontrivial.

The reduction algorithm in Section 2.4.1 and Theorem 1, imply that if the context distribution is known, then the agent does not need to share any information about the context to the learner.

Remark 1. Knowledge of the context distribution is possible in practice, since many times the context may be capturing well studied statistics (e.g., male or female, age, weight, income, race, dietary restrictions, emotional state, etc) - the advent of large data has made and will continue to make such distributions available. Similarly, actions may be finite (eg., restaurants to visit) or well described (e.g., released amounts of substances), and thus the distribution could be derived.

We developed a context compression scheme for the case of unknown distribution in [HYF22a]. However, the number of bits required is $\Theta(d)$, as opposed to zero bits for the known distribution case.

In addition to communication constraints, not sharing information about the context provides privacy benefits, as we have shown in [HGF24].

2.7 Conclusion and Open Research Directions

In this chapter, we presented a novel reduction from stochastic contextual linear bandit problems to (fixed-context) linear bandit problems; our reduction explains why results, that are not achievable for adversarial contexts, are possible for stochastic contextual bandits, and offers a framework that can be leveraged to gain new algorithms and bounds for contextual linear bandit problems. We illustrate the power of our approach by applying it to achieve improved bounds over a number of cases; this is not an exhaustive list, and we hope that our approach will be a useful tool to researchers in this field.

We leave the following as open problems:

- Improving the regret bound of the reduction from $\tilde{O}(d\sqrt{T})$ to $\tilde{O}(\sqrt{dT \log K})$ when the number of actions at each iteration $|\mathcal{A}_t| \leq K$ is small. The challenge is that the number of actions of the reduced linear bandit problem $|\mathcal{X}|$ may be large, while $|\mathcal{A}_t|$ is small for all t.
- Extending the reduction to slowly varying context distributions, and to the linear Markov decision processes (MDPs).
- Extending the reduction to non-linear contextual bandits.

CHAPTER 3

A Computationally-Efficient Algorithm for Batched Linear Contextual Bandits

3.1 Summary

In this chapter, we provide the first efficient batched algorithm for contextual linear bandits with large action spaces. Unlike existing batched algorithms that rely on action elimination, which are not implementable for large action sets, our algorithm only uses a linear optimization oracle over the action set to design the policy. The proposed algorithm achieves a regret upper bound $\tilde{O}(\sqrt{T})$ with high probability, and uses $O(\log\log T)$ batches, matching the lower bound on the number of batches [GHR19]. When specialized to linear bandits, our algorithm can achieve a high probability gap-dependent regret bound of $\tilde{O}(1/\Delta_{\min})$ with the optimal $\log T$ number of batches, where $\Delta_{\min}$ is the minimum reward gap between a suboptimal arm and the optimal. Our result is achieved via the reduction introduced in Chapter 2, and a novel soft elimination approach, that entails “shaping" the action sets at each batch so that we can efficiently identify (near) optimal actions.

3.2 Introduction

In numerous real-world use cases, the learner may be restricted to change the policy a limited (small) number of times.This constraint may stem from factors such as computation or communication considerations, or may be imposed by the nature of the application, as is the case in multi-stage clinical trials or online marketing campaigns with high response rates, where it is not feasible to update the policy after each response. Similarly, the use of crowdsourcing platforms or the need to conduct time-consuming simulations in reinforcement learning may require policies with limited adaptivity. As a result, there has been significant interest in designing algorithms that can achieve the optimal regret with limited policy switches [PRC16, AAA17, PAS18, GHR19, EKM21, JXX21, RYZ21, HZZ20]. This setup is known as the batched contextual linear bandit problem: the T rounds are partitioned into batches, and the learner can collect rewards and update the action selection policy only at the end of each batch.

Large Action Space. Contextual linear bandit applications frequently need to explore an extremely large (even continuous) set of actions, e.g., millions of products to be recommended. As other examples, in the classical bandit problem of clinical trials, each decision involves selecting a treatment option from a potentially infinite set of mixed treatments [Gro97]. In manufacturing problems, the goal is often to maximize revenue by selecting from a very large set of decisions, with the revenue associated with each decision being unknown [TVV04]. Additionally, in applications where actions correspond to images in a database or high-dimensional embeddings of complex documents like webpages, the set of actions can be vast [LCL10, ACE08]. As a result, there is a strong interest in algorithms that can be efficiently implemented when the action space is large or infinite [DHK08, AK08, HK16, YRS22, ZFL22].

While computationally-efficient batched algorithms exist for contextual linear bandits with small action sets, and efficient ones that are not batched exist for contextual linear bandits with large action sets, to date, *there are no efficient batched algorithms* that can handle large action spaces. Existing batched algorithms for contextual linear bandits [RYZ21, HZZ20] rely on *action elimination* that requires a linear scan of the action set; while efficient non-batched algorithms for large action spaces do not extend to the batched setting [RYZ21, HZZ20] (see related work in the following for more details).

3.2.1 Our Contributions

In this chapter, we provide the first efficient batched algorithm for contextual linear bandits with nearly optimal regret upper bound of $\tilde{O}(d^{3/2}\sqrt{T})$ with high probability, while using $O(\log\log T)$ batches, which matches the lower bound on the number of batches required to achieve $\sqrt{T}$-type regret bounds [GHR19]. For linear bandits, our algorithm can attain a high probability gap-dependent regret bound of $\tilde{O}(d^3/\Delta_{\min})$ with the optimal $\log T$ number of batches [GHR19], where $\Delta_{\min}$ represents the minimum reward gap between a suboptimal arm and the optimal.

Our algorithm for linear bandits, that we term SoftBatch, builds on a form of "soft elimination". Our observation is that, a good algorithm should be able to approximate the gap $\Delta(a)$ between each action $a \in \mathcal{A}$ and the optimal one with $O(\Delta(a))$ accuracy; and if we can do that, then we can use this knowledge to limit the number of times we play suboptimal actions, as well as use this knowledge to select which actions we want to play at all. As essentially all batched algorithms do, at each batch we select and play (a small number of) actions that enable to estimate well the unknown parameter vector without incurring large regret. In particular, for each batch, we choose a set of well-behaved basis actions (e.g., a barycentric spanner [AK08]), established by calling an optimization oracle polynomial times. However, instead of selecting at batch m, vectors from the "true" action set $\mathcal{A}$, we consider virtual "weighted" sets $\tilde{\mathcal{A}}_m$, where each action's magnitude is weighted inversely proportional to the estimated gap $\Delta(a)$, and select vectors guided by these weighted action sets. Then we play each basis action a a number of times inversely proportional to the square of the estimated gap $\Delta(a)$ to preserve small regret. This in return provides us an accurate estimator for the optimal parameter by the benign properties of the basis actions. Thus our approach implements a form of soft elimination (shaping) of the action set, where the actions closest to the optimal become increasingly dominant. A crucial part in our design is that we never actually calculate the gaps $\Delta(a)$ for all actions $a \in \mathcal{A}$ (only for the basis actions). The exploration policy we propose uses solely a linear optimization oracle applied to the original action set.

Our contextual bandit algorithm utilizes the reduction technique in Chapter 2 to transform the problem into a linear bandit problem. We incorporate the reduction into our batched linear bandit algorithm, by constructing an efficient linear optimization oracle for the exponentially large action set in the reduced problem using a linear optimization oracle for the original action sets (contexts).

Our proof techniques may be of independent interest. We develop a novel approach to bound regret in linear bandits, we design an efficient exploration policy using inverse squared gap weighting, and a simple method to handle the case where the action set does not span $\mathbb{R}^d$, where d is the problem dimension. Our approach avoids the necessity of imposing assumptions, such as the one in [ZFL22], which entails having a subset of d actions forming a matrix with determinant at least r^d for a constant r. These assumptions can be strong, particularly when dealing with changing action sets, and may not hold after modifying the action set, for instance, by eliminating or weighting actions.

3.2.2 Related Work

Contextual linear and linear bandits have had significant impact both in theory and practice [APS11, DHK08, LWZ19, LWC21, LS20, LCL10, TM17, ABC16, BBG12, CGL21]). The best performing algorithms achieve a regret bound $\tilde{O}(d\sqrt{T})$[1] [APS11, DHK08], matching the regret lower bound $\Omega(d\sqrt{T})$ [LS20]. The same algorithms achieve a nearly optimal regret upper bound $\tilde{O}(\frac{d^2}{\Delta_{\min}})$ if the minimum gap of suboptimal arms is lower bounded by $\Delta_{\min}$. However, the resulting policies require updates at every time step and involve solving a non-convex optimization problem, which is not practical for large action spaces [Sah74, DHK08].

Batched algorithms. Existing batched algorithms for contextual linear bandits [RYZ21, HZZ20, HYF23] have achieved nearly optimal regret upper bounds of $\tilde{O}(d\sqrt{T})$. However, these algorithms rely on action elimination, which involves either performing a linear scan on the action set or solving an optimization problem over the non-convex set of good (not eliminated) actions to design and implement the policy at each time step. Similarly, batched algorithms for linear bandits [LSW20, EKM21] also rely on action elimination. Although, unlike contextual bandits, the

[1] $\tilde{O}$ hides log factors.

elimination constraint in linear bandits can be linear, which can be exploited to efficiently compute the policy (under certain assumptions) [AK08], resulting in an $\tilde{O}(d^{3/2}\sqrt{T})$ regret upper bound, it requires solving an optimization problem over the action set with an elimination constraint. This can be much harder than solving the optimization problem over the action set without additional constraints for some sets, such as the non-convex set resulting from the reduction of contextual to linear bandits in Chapter 2.

Efficient algorithms for large action spaces. There is a long line of work on efficient algorithms for linear bandits that only rely on a linear optimization oracle over the action set [AK08, DHK08, BCK12, HK16, IHS19]. However, these algorithms cannot be extended to the batched setting without extra assumptions on the action set, and more importantly, they do not extend to the batched contextual setting. Existing efficient algorithms for contextual linear bandits [AG13b, ZFL22, DHK08] can achieve $\tilde{O}(d^{3/2}\sqrt{T})$ regret bound, but it remains unclear if they can be extended to the batched setting, particularly given the challenge posed by changing action sets. Another line of work attempts to design efficient algorithms using hashing-based methods to approximate the maximum inner product [YRS22, JBN17], but these methods result in complexity that is sublinear but still polynomial in the number of actions.

Table 3.1 summarizes how our results position w.r.t. related work.

3.3 Model and Notation

Notation. We use $[n]$ for a natural number n to denote the set $\{1, \cdots, n\}$; $\mathbf{1}(E)$, for an event E, to denote the indicator function which returns 1 if E holds and 0 otherwise; $\mathcal{B}_r = \{a \in \mathbb{R}^d | \|a\|_2 \leq r\}$ to denote the ball of center 0 and radius r; $\mathcal{S}_r = \{a \in \mathbb{R}^d | \|a\|_2 = r\}$ to denote the sphere of center 0 and radius r; and $\|a\|_{\boldsymbol{V}} = \sqrt{a^\top \boldsymbol{V} a}$ to denote the matrix norm of a vector $a \in \mathbb{R}^d$ with respect to a positive semi-definite matrix $\boldsymbol{V}$. Table B.1 in App. B.1 summarizes our notation.

Algorithm	Regret Bound	Context	Efficient	Number of batches
[LSW20, EKM21]	$\tilde{O}(d\sqrt{T})$	✗	requires assumptions	$O(\log T)$
[AK08, DHK08, BCK12, HK16, IHS19]	$\tilde{O}(d\sqrt{T})$	✗	✓	T
[APS11, DHK08]	$\tilde{O}(d\sqrt{T})$	✓	✗	T
[RYZ21, HZZ20, HYF23]	$\tilde{O}(d\sqrt{T})$	✓	✗	$O(\log\log T)$
[AG13b, ZFL22, DHK08]	$\tilde{O}(d^{3/2}\sqrt{T})$	✓	✓	T
Chapter 3	$\tilde{O}(d^{3/2}\sqrt{T})$	✓	✓	$O(\log\log T)$

Table 3.1: Comparison with related work

Contextual Linear Bandits. We consider a contextual linear bandit problem over an horizon of length T, where at each round $t \in [T]$, the learner receives a set of actions $\mathcal{A}_t \subseteq \mathbb{R}^d$ sampled from an unknown distribution $\mathcal{D}$ independently from other rounds. The learner plays an action $a_t \in \mathcal{A}_t$ and receives a reward $r_t = \langle a_t, \theta_\star \rangle + \eta_t$, where $\theta_\star$ is an unknown system parameter vector with $\theta_\star \in \mathbb{R}^d$, and η_t is noise that is zero mean conditioned on the filtration of historic information $(\mathcal{A}_1, a_1, r_1, \cdots, \mathcal{A}_t, a_t)$. The learner adopts a **policy** that maps the history $(\mathcal{A}_1, a_1, r_1, \cdots, \mathcal{A}_t)$ to a distribution over the action set $\mathcal{A}_t$, with the objective of minimizing the pseudo regret defined as

$$R_T = \sum_{t=1}^{T} \sup_{a \in \mathcal{A}_t} \langle a - a_t, \theta_\star \rangle. \tag{3.1}$$

For simplicity, we assume that $\mathcal{A}_t$ is compact for all $t \in [T]$ almost surely, which ensures the existence of an action $a_\theta \in \mathcal{A}_t$ that attains the supremum $\sup_{a \in \mathcal{A}_t} \langle a, \theta \rangle$. Non-compact sets can be handled using sufficiently small approximations. We also adopt the following standard assumption [LS20].

Assumption 1. *(Boundedness.) $\theta_\star \in \mathcal{B}_1$, $\mathcal{A}_t \subseteq \mathcal{B}_1$, and $|r_t| \leq 1$ almost surely $\forall t \in [T]$.*

Linear Bandits. Recall from Chapter 2 that changing the action set over time enables to model contextual information. If the action space is fixed, namely, $\mathcal{A}_t = \mathcal{A}$ for for all $t \in [T]$, the problem is known as Linear Bandits. For Linear Bandits, we denote an optimal action by $a^\star = \arg\max_{a\in\mathcal{A}} \langle a, \theta_\star \rangle$ and define the gap $\Delta_a = \langle a^\star - a, \theta_\star \rangle$ for all actions $a \in \mathcal{A}$.

Batched Setting. In a batched setting, the learner is only allowed to change the policy at M pre-chosen rounds, where M is the number of batches. Batch m includes T_m rounds, $m \in [M]$, with $\sum_{m=1}^{M} T_m = T$. In each batch, the learner adopts a policy π that takes as input the action set $\mathcal{A}_t$ along with all the previous history except for rewards observed in the current batch, and outputs a distribution over the action set $\mathcal{A}_t$. In particular, the rewards of the actions pulled in the current batch are utilized solely to update the policy at the end of the batch.

Regularized least squares. Let $\{a_i, r_i\}_{i=1}^{n}$ be a sequence of n pulled actions and observed rewards over n rounds. The regularized least squares estimate $\hat{\theta}$ of $\theta_\star$ based on this action-reward sequence can be calculated as

$$\hat{\theta} = \boldsymbol{V}^{-1} \sum_{i=1}^{n} r_i a_i, \tag{3.2}$$

where $\boldsymbol{V} = \lambda \mathbf{I} + \sum_{i=1}^{n} a_i a_i^\top$, and λ is the regularization parameter.

Goal. Our goal is to design efficient batched algorithms for Contextual Linear and Linear Bandits with large (even infinite) action spaces that achieve (nearly) optimal regret.

We will do so by making use of the linear optimization oracles defined next.

Definition 2. A **linear optimization oracle** for a set $\mathcal{A}$ is a function $\mathcal{O}(\mathcal{A}; .)$ which takes as input $\theta \in \mathcal{B}_1$ and outputs $\mathcal{O}(\mathcal{A}; \theta) \in \mathcal{A}$ with $\langle \mathcal{O}(\mathcal{A}; \theta), \theta \rangle = \sup_{a\in\mathcal{A}} \langle a, \theta \rangle$. An **approximate linear optimization oracle with additive error** at most ϵ for the set $\mathcal{A}$ is a function $\mathcal{O}_\epsilon^+(\mathcal{A}; .) : \mathcal{B}_1 \to \mathcal{A}$ that satisfies $\langle \mathcal{O}_\epsilon^+(\mathcal{A}; \theta), \theta \rangle \geq \sup_{a\in\mathcal{A}} \langle a, \theta \rangle - \epsilon,\ \forall \theta \in \mathcal{B}_1$. An **approximate linear optimization oracle with multiplicative error** $0 < \alpha < 1$ for the set $\mathcal{A}$ is a function $\mathcal{O}_\alpha^\times(\mathcal{A}; .) : \mathcal{B}_1 \to \mathcal{A}$ that satisfies $\langle \mathcal{O}_\alpha^\times(\mathcal{A}; \theta), \theta \rangle \geq (1-\alpha) \sup_{a\in\mathcal{A}} \langle a, \theta \rangle,\ \forall \theta \in \mathcal{B}_1$.

Assumption 2. *(Linear optimization oracle.) We assume that we can access a linear optimization oracle $\mathcal{O}(\mathcal{A}_t; .)$ for each set of actions $\mathcal{A}_t$ with running time at most $\mathcal{T}_{opt}$ and space complexity $\mathcal{M}_{opt}$.*

We note that assuming a linear optimization oracle over $\mathcal{A}_t$ is natural [AK08, DHK08, BCK12, HK16, IHS19, ZFL22] since even if the learner perfectly learns the unknown parameter vector $\theta_\star$, the learner still needs to solve $\sup_{a\in\mathcal{A}_t} \langle a, \theta_\star \rangle$ to minimize the regret in (3.1).

3.4 Efficient Soft Elimination Algorithm for Linear Bandits

In this section we propose and analyze an algorithm (which we call SoftBach and describe in Algorithm 3) for linear bandits, that is, when $\mathcal{A}_t = \mathcal{A}$.

3.4.1 Main Result

The following two theorems, proved in App. B.4 and B.5, respectively, formally state that Algorithm 3 achieves (nearly) optimal regret using $M = \lceil \log\log T \rceil + 1$ batches with sample and time complexities polynomial in d and linear in T. We provide the algorithm description in Section 3.4.2 and a proof outline in Section 3.4.2.

Theorem 8. *Consider a linear bandit instance with action set $\mathcal{A} \subseteq \mathbb{R}^d$ and horizon T. There exists a universal constant C and a choice for the batch lengths such that Algorithm 3 finishes in at most $M = \lceil \log\log T \rceil + 1$ batches with regret bounded as*

$$R_T \leq C\gamma\sqrt{T}\log\log T \text{ with probability at least } 1-\delta, \tag{3.3}$$

where $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$, $C_L = e^8 d$ and δ is a parameter. Moreover, if $\forall a \in \mathcal{A}$ with $\Delta_a > 0$ we have $\Delta_a \geq \Delta_{\min}$, then there exists a choice of batch lengths so that Algorithm 3 finishes in at most $M = \log_4 T$ batches with regret bounded as

$$R_T \leq C\frac{\gamma^2}{\Delta_{\min}}\log T \text{ with probability at least } 1-\delta. \tag{3.4}$$

Our regret bounds achieve nearly optimal dependency on T, and match the best known regret bounds of $\tilde{O}(d^{3/2}\sqrt{T})$ for (unbatched) efficient contextual linear bandit algorithms [AG13b, ZFL22, DHK08], while losing a $\sqrt{d}$ factor when compared to the $\Omega(d\sqrt{T})$ lower bound [LS20]. This extra $\sqrt{d}$ factor is due to relying on the best known method to design a notion of spanner of the set of actions (as we explain in section 3.4.2) with radius $\sqrt{C_L} = O(\sqrt{d})$ using linear optimization oracles. Any future improvement that reduces the radius from $O(\sqrt{d})$ to $O(1)$ will immediately result in nearly optimal regret bounds for Algorithm 3. The following result upper bounds the time and space complexity.

Theorem 9. *Algorithm 3 finishes in $\tilde{O}(Td^2 + d^4M + \mathcal{T}_{opt}d^3M)$ runtime and uses $\tilde{O}(d^2 + \mathcal{M}_{opt})$ memory, where $\mathcal{T}_{opt}, \mathcal{M}_{opt}$ are the time and space complexity of the linear optimization oracle.*

We observe that unlike algorithms that require a linear scan on the action set, our space and time complexities are polynomial in the parameters d, T, and $\mathcal{T}_{\text{opt}}$.

3.4.2 SoftBatch (Algorithm 3) Description

Intuition. The main intuition behind SoftBatch is that, we do not need to necessarily eliminate suboptimal actions; it suffices to be able to select and play a small set of unique actions $\mathcal{C}_m$ in each batch m, that allows to estimate increasingly well the parameter vector θ_* and the best action $a^\star$ while playing suboptimal actions for a small number of times. Our algorithm proposes a novel way to select such sets $\mathcal{C}_m$ efficiently, through a form of "action set shaping" that we will describe in this section. Additionally, to learn θ_* while achieving a (nearly) optimal regret, SoftBatch plays each action $a \in \mathcal{C}_m$ a number of times $\propto 1/\Delta_a^2$, where $\Delta_a = \langle a_\star - a, \theta_\star \rangle$ is the gap for action a (i.e., we play the suboptimal actions in $\mathcal{C}_m$ for a small number of times so as not to accumulate regret). SoftBatch enables to estimate the gap Δ_a within a constant factor for any action a (yet only does so for a limited number of actions in each batch), and essentially uses the gaps Δ_a as a guide on which actions to play and for how many rounds each.

Steps. SoftBatch (Algorithm 1) takes as input the action set $\mathcal{A} \subseteq \mathbb{R}^d$, the horizon T, the number of

Algorithm 3 [SoftBatch] A Batched Algorithm for Linear Bandits

```
1: Input: action set $\mathcal{A} \subseteq \mathbb{R}^d$, horizon $T$, number of batches $M$, batch lengths $\{T_m\}_{m=1}^M$, confidence parameter $\delta$.
2: Let $\mathcal{A}' = \mathcal{A} \cup \mathcal{B}_{1/T}$,    $C_L = e^8 d$,    $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$.
3: Initialize: $\theta_1 = 0$, $a_1^\star$ is a random action in $\mathcal{A}$, $\Delta_1(a) = 1\ \forall a \in \mathcal{A}'$, and $T_0 = 1$.
4: for $m = 1 : M$ do
5:     Calculate $\{a_1, \dots, a_d\} = \text{LWS}(\mathcal{A}', \eta_m = \sqrt{T_{m-1}}/(8\gamma), a_m^\star, \theta_m)$.
6:     For the set $\{a_1, \dots, a_d\}$ assign $\pi(i) = \frac{1}{d}$, $\forall i \in [d]$.
7:     for $i = 1 : d$ do
8:         If $a_i \notin \mathcal{B}_{1/T}$, calculate $\Delta_m(a_i) = \langle a_m^\star - a_i, \theta_m \rangle$ and pull it $n_m(i) = \lceil \frac{\pi(i) T_m/8}{(1+\sqrt{T_{m-1}}\Delta_m(a_i)/(8\gamma))^2} \rceil$ times. go to step 10 if the number of pulls in the current batch reaches $T_m$. Terminate Algorithm 3 if the total number of pulls reaches $T$.
9:     Pull action $a_0 = a_m^\star$ for $\max\{0, T_m - \sum_{i=1}^d n_m(i)\}$ times.
10:    Compute the regularized (with $\lambda = 1$) least squares estimator $\boldsymbol{V}_m = \mathbf{I} + \sum_{i=1}^{T_m} \tilde{a}_i \tilde{a}_i^\top$ and $\theta_{m+1} = \boldsymbol{V}_m^{-1} \sum_{i=1}^{T_m} r_i \tilde{a}_i$, and $\tilde{a}_i$ is the action pulled in $i$-th round of the batch.
11:    Update $a_{m+1}^\star = \mathcal{O}^+_{\frac{1}{T}}(\mathcal{A}; \theta_{m+1})$.
```

batches M, and the batch lengths $\{T_m\}_{m=1}^M$, and operates as follows[2].

In batch m, the algorithm starts with a current estimate of the parameter vector $\theta_\star$, which we call θ_m, and an estimate of the optimal action $a^\star$ which we call $a_m^\star$; note that given these, we are able to estimate for any action $a \in \mathcal{A}$ the gap $\Delta_m(a) = \langle a_m^\star - a, \theta_m \rangle$ (but we will only do so for the actions the algorithm actually plays). The algorithm then calls LWS, a Linear Weighted Spanner subroutine (described in Algorithm 2), that it feeds with an augmented action space $\mathcal{A}' = \mathcal{A} \cup \mathcal{B}_{1/T}$ for reasons we will explain later. LWS selects d actions $\mathcal{C}_m = \{a_1, \cdots, a_d\}$ to play in batch m (note that some of these may belong to $\mathcal{B}_{1/T}$ and will in this case not be played). Each of these d actions a_i is pulled $n_m(i) \propto \frac{\pi(i)}{\Delta_m(a_i)^2}$ times, where $\pi(.)$ is a uniform exploration distribution with value $1/d$

[2]We discuss how to select M and $\{T_m\}$ in App. B.4.

for all the d actions. We show in the proof of Theorem 8 that $\sum_{i=1}^{d} n_m(i) \leq T_m, \forall m \in [M]$, with high probability. To guarantee that the length of the batch is T_m, the algorithm pulls $a_m^\star$ for the remaining rounds, if needed. At the end of the batch, the algorithm updates its estimate θ_{m+1} of the unknown parameter vector using regularized least squares.

The remaining core part of the algorithm to discuss is the subroutine LWS, and we do so next. We start by providing our reasoning behind the LWS design.

The LWS Algorithm. Recall that we want LWS at each batch m to select d vectors $\{a_i\} \subseteq \mathcal{A}'$ such that, by playing each $n_m(i)$ times, we can create a least-squares estimate θ_{m+1} of θ_* that allows an accurate estimate of the product $\langle a, \theta_\star \rangle$ for all $a \in \mathcal{A}$. It is well-known (see [LS20]) that the error in estimating $\langle a, \theta_\star \rangle$ is proportional to $\|a\|_{\boldsymbol{V}_m^{-1}}$, where $\boldsymbol{V}_m = \mathbf{I} + \sum_{i=1}^{T_m} a_i a_i^\top$ is the least squares matrix we used to estimate θ_{m+1}. Thus, essentially we want LWS to select d vectors $\{a_i\}$ that maintain a small $\|a\|_{\boldsymbol{V}_m^{-1}}$ for all actions $a \in \mathcal{A}$[3]. We can do so using what is called a G-optimal design [KW60].

Definition 3. (G-optimal design) For any set $\mathcal{A} \subseteq \mathbb{R}^d$, a subset $\mathcal{S} \subseteq \mathcal{A}$, together with a distribution π over $\mathcal{S}$ is said to be a C**-approximate optimal design** for $\mathcal{A}$ if for any $a \in \mathcal{A}$

$$\|a\|^2_{\boldsymbol{V}_\pi^{-1}} \leq Cd, \tag{3.5}$$

where $\boldsymbol{V}_\pi = \sum_{a_i \in \mathcal{S}} \pi(i) a_i a_i^\top$[4]. When $C = 1$ this is referred to as a **G-optimal design**.

Notice that if we were to play each action a_i for $n\pi(i)$ times, then $\boldsymbol{V}_\pi$ would be (approximately) a normalized least squares matrix since $\boldsymbol{V}_\pi + \mathbf{I}/n = \boldsymbol{V}/n$, and hence, $\|a\|^2_{\boldsymbol{V}^{-1}} \leq \|a\|^2_{\boldsymbol{V}_\pi^{-1}}/n$.

It is well-known that for any compact set, there exists a 1-approximate optimal design [KW60] with $|\mathcal{S}| = d$. However, computing an 1-approximate optimal design is NP-hard in general [GLS12, SEF14], even for small action sets. Computing a 2-approximate optimal design can be done in polynomial time [Tod16], but the complexity scales linearly with the size of the action

[3]Adding reward samples from the estimated best action $a_m^\star$ can only improve the least squares estimator.

[4]This summation assumes finiteness of the set $\mathcal{S}$ which suffices for our application.

set. Instead, we adopt an approach introduced in [AK08], which efficiently constructs an $O(\sqrt{d})$-approximate optimal design using only a linear optimization oracle. This relies on the concept of a barycentric spanner, which we define next.

Definition 4. (Barycentric spanner) For any set $\mathcal{A} \subseteq \mathbb{R}^d$, a subset $\mathcal{S} = \{a_1, \cdots, a_d\} \subseteq \mathcal{A}$ is said to be a C**-approximate barycentric spanner** for $\mathcal{A}$ if any $a \in \mathcal{A}$ can be expressed as a linear combination of vectors in $\mathcal{S}$ with coeficients in $[-C, C]$.

It is easy to see that a C**-approximate barycentric spanner** together with **a uniform distribution** $\pi(i) = 1/d$ results in a $C\sqrt{d}$-approximate optimal design [HK16, ZFL22]. And importantly, a C-approximate barycentric spanner for a set $\mathcal{A}$ can be constructed using at most $O(d^2 \log_C d)$ calls of a linear optimization oracle over the set $\mathcal{A}$ [HK16].

However, this is still not sufficient for us. Even though we can efficiently construct a $C\sqrt{d}$-approximate optimal design for $\mathcal{A}$, we do not want to pull these arms according to a uniform distribution; we want to pull action a_i with estimated gap $\Delta_m(a_i)$ for $n_m(i) = \lceil \pi(i)T_m/(1 + \sqrt{T_{m-1}}\Delta_m(a)/(8\gamma))^2 \rceil$ times to control the regret (which can be thought of as using a weighted distribution[5]). But if we do not use the uniform distribution, the resulting least squares matrix $\boldsymbol{V}_m$ may not satisfy that $\|a\|_{\boldsymbol{V}_m^{-1}}$ is sufficiently small for all actions a.

To account for this, instead of finding a C-approximate barycentric spanner for the set $\mathcal{A}$, at each batch m we consider a **virtual action set** $\tilde{\mathcal{A}}_m$, which we define as

$$\tilde{\mathcal{A}}_m = \{\phi_m(a) | a \in \mathcal{A}\}, \phi_m(a) = \frac{a}{1 + \eta_m \Delta_m(a)}, \tag{3.6}$$

where $\eta_m = \sqrt{T_{m-1}}/(8\gamma)$ and find actions $\{a_1, \cdots, a_d\} \in \mathcal{A}$ such that $\{\phi_m(a_i)\}_{i=1}^d$ forms a C-approximate barycentric spanner for $\tilde{\mathcal{A}}_m$. The least squares matrix at batch m can be bounded as

$$\boldsymbol{V}_m = \mathbf{I} + \sum_{i=0}^{d} \tilde{n}_m(i) a_i a_i^\top \geq \sum_{i=1}^{d} \frac{\pi(i)T_m/8}{(1 + \eta_m \Delta_m(a_i))^2} a_i a_i^\top = \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \phi_m(a_i) \phi_m(a_i)^\top \tag{3.7}$$

[5]The technique of inverse gap weighting was employed in [AL99, ZFL22], albeit with a different weighting approach using inverse gap instead of squared inverse gap, as utilized in our proposed schemes.

with high probability[6], where $\tilde{n}_m(i)$ is the number of times action a_i is played in batch m and $a_0 = a_m^\star$. That is, playing actions $\{a_1, \cdots, a_d\} \in \mathcal{A}$ for $n_m(i)$ times each, can equivalently be thought of as playing actions $\{\phi(a_1), \cdots, \phi(a_d)\} \in \tilde{\mathcal{A}}_m$ for $\pi(i)T_m$ times each; and since $\{\phi(a_1), \cdots, \phi(a_d)\}$ form an approximate optimal design (through a barycentric spanner) for the set $\tilde{\mathcal{A}}_m$, the resulting least squares matrix will lead to small $\|\phi_m(a)\|_{\boldsymbol{V}_m^{-1}}$ values. In our proofs we show that a small enough $\|\phi_m(a)\|_{\boldsymbol{V}_m^{-1}}$ implies $\|a\|_{\boldsymbol{V}_m^{-1}} = O(\Delta_a)$ as a result of the scaling in $\phi(a)$. We prove in Lemma 16 in App. B.4 that this allows to estimate Δ_a within a constant factor, which is all we need.

Intuitively, the virtual set $\tilde{\mathcal{A}}_m$ weighs the actions inversely proportional to the estimated gap $\Delta_m(a)$ and batch length $\sqrt{T_{m-1}}$: the larger the gap and T_{m-1}, the smaller magnitude the corresponding action has; this implements a form of soft elimination (shaping) of the action set, where the actions closest to the optimal become increasingly dominant as the batch length increases while the remaining fade out to zero. As a result, as m increases, the span of the optimal design focuses on the space where actions have small gaps, allowing to better distinguish among them.

To complete SoftBach (Algorithm 1), one last step is missing. LWS (Algorithm 4) follows standard steps (in Algorithm 4, see [AK08] for detailed explanation) to calculate the C-approximate barycentric spanner for $\tilde{\mathcal{A}}_m$. But to follow these steps, it requires the ability to solve the nonlinear optimization problem $\sup_{a\in\mathcal{A}} \langle \phi_m(a), \theta\rangle$, since $\phi_m(a) = a/(1 + \sqrt{T_{m-1}}\Delta_m(a)/(8\gamma))$ is nonlinear in a. To do so, we will use[7] an approximate oracle with multiplicative error, that we term **LW-ArgMax** and describe next.

LW-ArgMax Algorithm. LW-ArgMax (Algorithm 5) constructs an approximate oracle with $(1 - \alpha)$-multiplicative error for the optimization $\sup_{a\in\mathcal{A}} \langle \phi_m(a), \theta\rangle$. This is sufficient: we show

[6]We show in the proof of Theorem 8 that $\sum_{i=1}^{d} n_m(i) \le T_m \forall m \in [M]$ with high probability, hence, all the required $n_m(i)$ action pulls can be finished within the batch.

[7]A related problem was faced in [ZFL22], but with a different function hence, the resulting strategy does not apply in our case. Both [ZFL22] and our solution use the standard idea of line search, albeit with different steps and different number of iterations. The proof that our line search provides an approximate optimization oracle turns out to be much more involved than that of [ZFL22].

Algorithm 4 Linear Weighted Spanner (LWS) Algorithm

```
1:  Input: set of actions $\mathcal{A}$, parameter $\eta$, estimated best action $\hat{a}$, estimated parameter $\hat{\theta}$.
2:  Initialize: $\tilde{a}_i = e_i$, where $e_i$ is the $i$-th basis vector of dimension $d$. Let $\boldsymbol{A} = [\tilde{a}_1, \cdots, \tilde{a}_d]$.
3:  Let $C = \exp(1)$, $\Delta(a) = \langle \hat{a} - a, \hat{\theta} \rangle$, $\phi(a) = a/(1 + \eta\Delta(a))$.
4:  for $i = 1, \cdots, d$ do
5:      Find $\theta$ with $\langle \theta, \tilde{a} \rangle = \det(\tilde{a}, \boldsymbol{A}_{-i}),\ \forall \tilde{a} \in \mathbb{R}^d$.
6:      $a^+ = \text{LW-ArgMax}(\mathcal{A}; \frac{\theta}{\|\theta\|_2}, \eta, \hat{a}, \hat{\theta})$, $a^- = \text{LW-ArgMax}(\mathcal{A}; \frac{\theta}{\|\theta\|_2}, \eta, \hat{a}, -\hat{\theta})$.
7:      $a_i = \arg\max_{b \in \{a^+, a^-\}} |\langle \phi(b), \theta \rangle|$, $\tilde{a}_i = \phi(a_i)$.
8:
9:  for $i = 1, \cdots, d$ do
10:     Find $\theta$ with $\langle \theta, \tilde{a} \rangle = \det(\tilde{a}, \boldsymbol{A}_{-i}),\ \forall \tilde{a} \in \mathbb{R}^d$.
11:     $a^+ = \text{LW-ArgMax}(\mathcal{A}; \frac{\theta}{\|\theta\|_2}, \eta, \hat{a}, \hat{\theta})$, $a^- = \text{LW-ArgMax}(\mathcal{A}; \frac{\theta}{\|\theta\|_2}, \eta, \hat{a}, -\hat{\theta})$.
12:     $a = \arg\max_{b \in \{a^+, a^-\}} |\langle \phi(b), \theta \rangle|$.
13:     if $|\det((\phi(a), \boldsymbol{A}_{-i}))| \geq C|\det(\boldsymbol{A})|$ then
14:         $a_i = a, \tilde{a}_i = \phi(a)$.
15:         go to line 8.
16: Return: $a_1, \cdots, a_d$.
```

in Lemma 3 that Algorithm 4 can use LW-ArgMax to compute a C/α-approximate barycentric spanner for $\tilde{\mathcal{A}}_m$.

Recall that, before providing the action set $\mathcal{A}$ to Algorithm 4, SoftBatch extends to $\mathcal{A}' = \mathcal{A} \cup \mathcal{B}_{1/T}$[8]. This guarantees that: $\mathcal{A}'$ spans $\mathbb{R}^d$ (required to find a barycentric spanner [AK08]), and $\sup_{a \in \mathcal{A}'} \langle a, \theta \rangle \geq 1/T$ for all θ with $\|\theta\|_2 = 1$ which implies that any approximate optimization oracle with additive error less than $1/(2T)$ has multiplicative error of at most $1/2$. The extension of the set $\mathcal{A}$ results in the barycentric spanner possibly containing points not in $\mathcal{A}$. However, we show

[8]The linear optimization problem $\max_{a \in \mathcal{A}'} \langle a, \theta \rangle$ can be solved by comparing $\max_{a \in \mathcal{A}} \langle a, \theta \rangle$ and $\max_{a \in \mathcal{B}_{1/T}} \langle a, \theta \rangle = 1/T$.

Algorithm 5 LW-ArgMax Algorithm

```
1: Input: set of actions $\mathcal{A}$, $\theta \in \mathcal{S}_1$, parameter $\eta$, estimated best action $\hat{a}$, estimate $\hat{\theta}$, horizon T.
2: Let $\Delta(a) = \langle \hat{a} - a, \hat{\theta} \rangle$, $\phi(a) = a/(1 + \eta\Delta(a))$.
3: Let $W = 3\log T$, $N = 36W\log^2(T)$, $s = 1 - 1/6\log T$, $\epsilon' = (1 - \exp(-1))/(12T^{7+12\log T})$.
4: Initialize $z = 2^W$.
5: for $i = 1, \cdots, N+1$ do
6:     $\tilde{\theta} = (1 + 1/W)z\theta + z^{1+1/W}\eta\hat{\theta}$
7:     $a_i = \mathcal{O}^+_{\epsilon'}(\mathcal{A}; \tilde{\theta}/\|\tilde{\theta}\|_2)$.
8:     $z \leftarrow zs$.
9: Return: $\arg\max_{a \in \{a_i\}_{i=1}^N} \langle \phi(a), \theta \rangle$.
```

that removing these points only affects $\sup_{a\in\mathcal{A}} \|a\|_{\boldsymbol{V}^{-1}}$ by a constant factor, since $\mathcal{B}_{1/T}$ has a small radius. Extending the set $\mathcal{A}$ to $\mathcal{A}'$ also handles the case where the span of $\mathcal{A}$ is smaller than $\mathbb{R}^d$, that was typically handled in literature by constructing a basis of $\mathcal{A}$ which can be complicated for some sets.

LW-ArgMax then builds on the following observation (proved as part of the proof of Lemma 2):

$$\arg\max_{a\in\mathcal{A}'} \langle \phi(a), \theta \rangle (\langle a, \theta \rangle)^{1/W} = \arg\max_{a\in\mathcal{A}'} \sup_{z\geq 0} L_z(a), \tag{3.8}$$

where $L_z(a) = z \cdot (1+1/W) \cdot \langle a, \theta \rangle - z^{1+1/W}(1+\eta\Delta(a))$ and $\Delta(a) = \sup_{b\in\mathcal{A}} \langle b - a, \theta_\star \rangle \; \forall a \in \mathcal{A}'$.

By choosing W to be large enough, the left hand side of (3.8) becomes a good approximation for $\langle \phi(a), \theta \rangle$. For a fixed z, the supremum on the right hand side of (3.8) reduces to a linear optimization over the set $\mathcal{A}'$ (that we solve using an approximate linear optimization oracle). Although the optimal value of z is not known, it can be bounded (see equation (B.16) in App. B.2); thus LW-ArgMax scans between upper and lower bounds on the optimal z with a constant multiplicative step. The pseudo-code is provided in Algorithm 5.

Proof Outline for Theorem 8

We start by proving that LW-ArgMax is an approximate linear optimization oracle for the set $\tilde{\mathcal{A}}$ with $1-\exp(-3)$ multiplicative error. The result is stated in Lemma 2 and proved in App. B.2.

Lemma 2. Let $T \geq 3, \eta \in \mathbb{R}, \hat{a} \in \mathbb{R}^d, \hat{\theta} \in \mathcal{B}_T$ be given parameters, and $\mathcal{A}$ be a given set. Let $\Delta(a), \phi(a)$ denote $\Delta(a) = \langle \hat{a} - a, \hat{\theta} \rangle, \phi(a) = a/(1+\eta\Delta(a))$. If $\mathcal{B}_{1/T} \subseteq \mathcal{A} \subseteq \mathcal{B}_1, |\eta| \leq T$ and $1/2 \leq 1 + \eta\Delta(a) \leq T^2,\ \forall a \in \mathcal{A}$, then for any $\theta \in \mathcal{S}_1$, LW-ArgMax outputs an element $a \in \mathcal{A}$ such that

$$\langle \phi(a), \theta \rangle \geq \exp(-3) \sup_{b \in \{\phi(b') | b' \in \mathcal{A}\}} \langle b, \theta \rangle. \tag{3.9}$$

The conditions of Lemma 2 are easy to verify for all batches m; namely, $\mathcal{B}_{1/T} \subseteq \mathcal{A} \subseteq \mathcal{B}_1$ holds as we extend the set of actions by adding $\mathcal{B}_{1/T}$ before feeding it into Algorithm 5 and the condition $1/2 \leq 1 + \eta\Delta(a) \leq T^2,\ \forall a \in \mathcal{A}$ is proved in Theorem 8 for all the inputs fed into Algorithm 5.

Given the result of Lemma 2, we next show that Algorithm 4 finds a C/α-approximate barycentric spanner of the set $\tilde{\mathcal{A}}_m,\ \forall m \in [M]$. This is done by slightly adapting the proof of Proposition 2.5 in [AK08] to work with approximate linear optimization oracles instead of exact oracles. The result is stated in the following theorem and the proof is provided in App. B.3 for completeness.

Lemma 3. Let $\eta \in \mathbb{R}, \hat{a} \in \mathbb{R}^d, \hat{\theta} \in \mathbb{R}^d$ be given parameters, and $\mathcal{A}$ be a given set. Let $\Delta(a), \phi(a)$ denote $\Delta(a) = \langle \hat{a} - a, \hat{\theta} \rangle, \phi(a) = a/(1+\eta\Delta(a))$. Suppose that $\langle \phi(\text{LW-ArgMax}(\theta)), \theta \rangle \geq \alpha \sup_{a \in \mathcal{A}} \langle \phi(a), \theta \rangle$, then Algorithm 4 computes a C/α-approximate barycentric spanner for the set $\tilde{\mathcal{A}} = \{\phi(a) | a \in \mathcal{A}\}$ with at most $O(d^2 \log_C(d/\alpha))$ calls to LW-ArgMax.

To build our regret bounds, we essentially prove that a number of pulls of $\propto \frac{\pi(i)}{\Delta_m(a_i)^2}$ for action a_i enables to estimate the gap $\Delta_m(a_i)$ within a constant factor of the real gap Δ_{a_i}. To do so, we start by providing an error bound for estimating $\langle \phi_m(a), \theta_\star \rangle$ using standard sub-Gaussian concentration inequalities. Then, through mathematical induction, we extend this bound to the error of the action mean estimates $\langle a, \theta_m \rangle$. Intuitively, if we believe that $\langle a, \theta_m \rangle$ is a good estimate of $\langle a, \theta_\star \rangle$ for all actions, which implies $\Delta_m(a)$ is a good estimate of Δ_a at batch m, then even though the scale in

ϕ_{m+1} by $\Delta_m(a)$, this property will continue to hold at batch $m+1$. The constants multiplying $\Delta_m(a)$ in ϕ_m are carefully designed to enable this. Finally, we show that the inverse squared gap weighting of the distribution enables to tightly upper bound the regret. $\square$

3.5 Algorithms for Contextual Linear Bandits

Our algorithm for contextual linear bandits is based on a technique proposed in Chapter 2, which reduces the contextual linear to a linear bandit setting. However, we cannot directly apply the reduction from Chapter 2 in Algorithm 3, as the reduction is not necessarily computationally efficient. Instead, we build a new algorithm (see Algorithm 6) that incorporates reduction steps within Algorithm 3. One challenge we encounter is the introduction of a large, non-convex action set through the reduction process. To address this, we utilize the linear optimization oracle over the new action set, proposed in Chapter 3, to ensure the efficiency of Algorithm 6. Additionally, the reduction requires estimating the expected value of a function g, and we carefully design the batch lengths to perform this estimation effectively.

We modify Algorithm 3 by adapting the action set in each batch based on the estimate of g, i.e., the set $\mathcal{X}_m$ (note that we do not need to explicitly calculate the sets $\mathcal{X}_m$). However, an issue arises where the estimate of $\theta_\star$ at batch m depends on the approximate optimal design from batch $m-1$, which employs the action set $\mathcal{X}_{m-1}$ estimated from the contexts of batch $m-2$. In the proof of Theorem 10, we demonstrate that this leads to regret proportional to $T_m/\sqrt{T_{m-2}}$. If the batch lengths grow rapidly, significant regret may occur. To mitigate this, we reduce the growth rate of batch lengths by allowing them to increase only when m is odd (a similar technique was employed in Chapter 2).

The pseudo-code for our algorithm for linear contextual bandits is provided in Algorithm 6. The algorithm follows similar steps to Algorithm 3 with the following exceptions. The set of actions $\mathcal{X}_m$ is updated (see step 14) in every batch using contexts observed in the previous batch. It is important to note that these sets $\mathcal{X}_m$ (in steps 4 and 14) are never actually computed; the definitions are provided for notation purposes. We only need an approximate optimizer for the set $\mathcal{X}_m$ to construct the approximate barycentric spanner in Algorithm 4. As shown in Lemma 1 in Chapter 2, $g^{(m)}([\theta]_q)$ for sufficiently small q can serve as our approximate oracle. Furthermore, the computation of $g^{(m)}([\theta]_q)$ can be performed using $O(T)$ calls to the linear optimization oracle $\mathcal{O}(\mathcal{A}_t; .)$, hence, with complexity of $O(T\mathcal{T}_{\text{opt}})$.

Additionally, we assume that LW-ArgMax (and LWS) returns for each a_i the value $\theta^{(i)} = [\tilde{\theta}_i]_q$. Here, $\tilde{\theta}_i$ is the input to the approximate linear optimization oracle which yielded the output a_i. The final difference from Algorithm 3 is that we do not play action a_i for $n_m(i)$ times (note that a_i may not be in $\mathcal{A}_t$); instead we play policy $\mathcal{O}(\mathcal{A}_t; \theta^{(i)})$ for $n_m(i)$ times, where $\theta^{(i)}$ is the parameter associated with a_i returned by LWS (Algorithm 4) as described earlier.

The following theorem describes our main result.

Theorem 10. *Consider a contextual linear bandit instance with $\mathcal{A}_t$ generated from an unknown distribution $\mathcal{D}$. There exists a universal constant C and choice for batch lengths such that Algorithm 6 finishes in $O(\log\log T)$ batches with regret upper bounded as*

$$R_T \leq C\gamma\sqrt{T}\log\log T$$

with probability at least $1-\delta$, where $\gamma = 10\sqrt{C_L d(\log(16M/\delta) + 57d\log^2(6T))}$. Moreover, the running time and space complexity are $\tilde{O}(d^4 + \mathcal{T}_{opt}d^3T)$, $\tilde{O}(d^2 + \mathcal{M}_{opt})$ respectively.

To implement Algorithm 6 efficiently we need: (i) an **approximate linear optimization oracle for the set $\mathcal{X}_m$** with additive error at most $\epsilon = (1-\exp(-1))/(12T^{7+12\log T})$: it was shown in Lemma 1 in Chapter 2 that $g^{(m)}([\theta]_q)$ can be used as our approximate oracle for $q \leq \epsilon/2$; and (ii) an **inverse of the function** $g^{(m)}$ to find θ_t associated with $g^{(m)}(\theta_t)$ to play the action $a_t = \mathcal{O}(\mathcal{A}_t; \theta_t)$: we observe that all actions played by our algorithms (Algorithm 3 and 6) are the output of the approximate optimization oracle for some θ; namely, for Algorithm 6 any pulled action is of the

[9]Recall that in the contextual setting we assume that LW-ArgMax (and LWS) returns for each a_i the value $\theta^{(i)} = [\tilde{\theta}_i]_q$, where $\tilde{\theta}_i$ is the input to the approximate linear optimization oracle that resulted in the output a_i.

Algorithm 6 Efficient Batched Algorithm for contextual linear bandits

```
1: Input: number of batches $M$, batch lengths $\{T_m\}_{m=1}^M$, horizon $T$, confidence parameter $\delta$, set of unknown parameters $\Theta \subseteq \mathcal{B}_1$, discretization parameter $q$.
2: Select modified batch lengths $\{\tau_m\}_{m=1}^{2M}$ to $\tau_m = T_{m//2}$, where $//$ is the integer division.
3: Let $C_L = \exp(8)d, \quad \gamma = 10\sqrt{C_L d(\log(8M/\delta) + 57d\log^2(6T))}, \ \tau_{-1} = \tau_0 = 1, \ \Theta' = \{[\theta]_q = \lfloor \theta/(\sqrt{d}q) \rfloor \sqrt{d}q | \theta \in \Theta\}$.
4: Let $g^{(1)} : \Theta' \to \mathbb{R}^d$ be defined as $g^{(1)}(\theta) = 0, \ \forall \theta \in \Theta'$, and let $\mathcal{X}_1 = \{g^{(1)}(\theta) | \theta \in \Theta'\}, \mathcal{X}_1' = \mathcal{X}_1 \cup \mathcal{B}_{1/T}$, $\Delta_1(a) = 1 \ \forall a \in \mathcal{X}_1'$.
5: Initialize: $\theta_1 = 0$, $a_1^\star$ to be a random action in $\mathcal{X}_1$.
6: for $m = 1 : 2M$ do
7:     Calculate $\{a_i, \theta^{(i)}\}_{i=1}^d = \text{LWS}(\mathcal{X}_m', \eta_m = \sqrt{\tau_{m-2}}/(8\gamma), a_m^\star, \theta_m)$, where $a_i = g^{(m)}(\theta^{(i)})$.[9]
8:     Let $\pi(i) = 1/d \quad \forall i \in [d]$, $a_0 = a_m^\star = g^{(m)}(\theta^{(0)})$, where $\theta^{(0)} = [\theta_m]_q = q\lfloor \theta_m \sqrt{d}/q \rfloor / \sqrt{d}$.
9:     for $i = 1 : d$ do
10:        If $a_i \notin \mathcal{B}_{1/T}$, calculate $\Delta_m(a_i) = \langle a_m^\star - a_i, \theta_m \rangle$ and play $a = \mathcal{O}(\mathcal{A}_t; \theta^{(i)})$, $n_m(i) = \lceil \frac{\pi(i)\tau_m/4}{(1+\sqrt{\tau_{m-1}}\Delta_m(a_i)/(8\gamma))^2} \rceil$ times. go to step 12 if the number of pulls in the current batch reaches $\tau_m$. Terminate Algorithm 1 if the total number of pulls reaches $T$.
11:    play $a = \mathcal{O}(\mathcal{A}_t; \theta^{(0)})$ for $\max\{0, \tau_m - \sum_{i=1}^d n_m(i)\}$ times.
12:    Compute the regularized (with $\lambda = 1$) least squares estimator $\boldsymbol{V}_m = \mathbf{I} + \sum_{i=1}^{\tau_m} a_i a_i^\top$ and $\theta_{m+1} = \boldsymbol{V}_m^{-1} \sum_{i=1}^{\tau_m} r_i a_i$.
13:    Update $a_{m+1}^\star = \mathcal{O}_{1/T}^+(\mathcal{X}_m; \theta_{m+1})$.
14:    Let $g^{(m+1)}(\theta) = \frac{1}{\tau_m} \sum_{t \in H_m} \mathcal{O}(\mathcal{A}_t; \theta)$, $\mathcal{X}_{m+1} = \{g^{(m+1)}(\theta) | \theta \in \Theta'\}$, $\mathcal{X}_{m+1}' = \mathcal{X}_{m+1} \cup \mathcal{B}_{1/T}$, where $H_m$ is the set of indices of the rounds in batch $m$.
```

form $g^{(m)}([\theta]_q)$ for some input to the approximate oracle θ. Hence, the inversion of $g^{(m)}$ for the actions pulled by Algorithm 6 can be performed by storing $[\theta]_q$ whenever the action $g^{(m)}([\theta]_q)$ is stored. This increases both the space and time complexity only by a constant factor.

Gap-dependent regret bounds for contextual linear bandits. The main difficulty in extending the gap-dependent regret bounds in Theorem 8 to the contextual case is that a large minimum action gap in the original action sets $\mathcal{A}_t$ does not imply a large gap in the reduced action set $\mathcal{X}$. As a simple example consider $d = 1$, $\theta_\star = 1$, and two action sets $\mathcal{A}_1 = \{-1, 1\}$, and $\mathcal{A}_2 = \{-1\}$. At each iteration the learner receives the action set $\mathcal{A}_1$ with probability p and $\mathcal{A}_2$ with probability $1 - p$ independently from other iterations. Recall that the action set in the reduced instance $\mathcal{X} = \{g(\theta)|\theta \in [-1, 1]\}$, where $g(\theta) = \mathbb{E}_{\mathcal{A}\sim\mathcal{D}}[\arg\max_{a\in\mathcal{A}}\langle a, \theta\rangle]$. For $\theta \geq 0$ we have that $g(\theta) = p(1) + (1-p)(-1) = 2p - 1$, while for $\theta < 0$ we have $g(\theta) = p(-1) + (1-p)(-1) = -1$. Then $\mathcal{X} = \{-1, 2p - 1\}$. Therefore, the suboptimality gap is $\Delta_{\min} = (2p - 1)(1) - (-1)(1) = 2p$ which can be arbitrarily small depending on p. Note that in the original contextual bandit instance, the minimum gap is at least 2 for both action sets.

While it may be possible to provide gap dependent regret bounds for our algorithm in the contextual case, this will require more sophisticated regret analysis that does not only rely on the reduced linear bandit instance.

Numerical results. In App. B.7, we provide a numerical example to compare the computational complexity of computing the exploration policy of our algorithm versus the complexity of computing the policy in [ZJZ21].

3.6 Conclusion and Open Research Directions

In this chapter, we proposed the first efficient batched algorithm for contextual linear bandits with large action spaces. Our algorithm achieves a high-probability regret upper bound of $\tilde{O}(d\sqrt{dT})$, uses $O(\log\log T)$ batches, and has a computational complexity that is linear in T and polynomial in d.

Even though our algorithm for batched contextual linear bandits, and existing efficient non-batched algorithms, achieve regret bounds with nearly optimal dependency on the time horizon T, they fail to achieve the optimal dependency on the dimension d. Improving the additional $\sqrt{d}$ factor in the regret bound of these efficient algorithms is an interesting and important open problem.

CHAPTER 4

Reward Compression for Bandits

4.1 Summary

In this chapter we move from compressing contexts to compressing the rewards collected by the distributed agents in the multi-armed bandit problem. Unlike compressing contexts, compressing the reward is also needed in the unstructured multi-arm bandit problem (explained in this chapter). Hence, this chapter considers the unstructured multi-arm bandit problem as well as the contextual linear bandit problem.

By providing nearly matching upper and lower bounds, we tightly characterize the *number of bits* needed per reward for the learner to accurately learn without suffering additional regret. In particular, we establish a generic reward quantization algorithm, $QuBan$, that can be applied on top of any (no-regret) MAB algorithm to form a new communication-efficient counterpart. $QuBan$ requires only a few (converging to as low as 3 bits as the number of iterations increases) bits to be sent per reward while preserving the *same* regret bound as uncompressed rewards. Our lower bound is established via constructing hard instances from a subgaussian distribution. Our theory is further corraborated by numerical experiments.

4.2 Introduction

MAB systems in areas such as mobile healthcare, social decision-making and spectrum allocation have already been implemented in a distributed manner, using limited bandwidth wireless links and simple sensors with low computational power [AMT11, BES13, BES14, MGP15, SFS18, DLL19]. Motivated from such communication constrained environments, in this chapter we explore compression schemes tailored to distributed MAB applications.

In the unstructured MAB problem, the set of actions is finite (k actions), and the action means are arbitrary (i.e., may not follow a linear structure as in linear bandits).

Work on the unstructured multi-armed bandit (MAB) algorithms and their applications span several decades, cultivating a rich literature that considers a variety of models and algorithmic approaches [LS20]. MAB algorithms include explore-then-commit [Rob52, Ans63], ϵ-greedy [ACF02], Thomson sampling [Tho33], and the upper confidence bound (UCB) [Lai87, ACF02], to name a few. Under some assumptions on the reward distribution, the explore-then-commit and ϵ-greedy algorithms achieve a regret bound $\propto O(\sqrt{T})$, where T is number of steps the learner plays, for the worst-case but known minimum reward gap[1], while Thomson sampling and UCB achieve a regret bound $\propto O(\sqrt{T\log(T)})$ without knowledge of the minimum means gap[2]. However, all these works assume that the rewards can be communicated to the learner at full precision which can be costly in communication-constrained setups. In this chapter we ask: *is it possible to perform efficient and effective bandit learning with only a few bits communicated per reward?*

In particular, in this chapter we consider reward compression for the setup illustrated in Fig. 4.1, where a central learner can directly communicate with a set of agents. We assume that agents can observe and communicate rewards - but do not keep memory of past rewards. There are several use cases where this setup can apply: the agent may be mobile (e.g., the central learner is a "traffic policeman" that directs passing by small drones to perform manoeuvres and searches for best current policies); they may be low complexity sensors (e.g., swarms of tiny robots such as RoboBees and RoboFlies [WNW13], wearable -inside and outside the body- sensors, backscatterer and RFID networks, IoT and embedded systems); or they may simply be willing to report a few rewards but

[1]The reward gap is defined to be the difference between the reward means of the best and second best arm.

[2]Variants of the UCB [AB09, DP16] can achieve regret $\propto O(\sqrt{T})$, but can be worse than UCB in some regimes [LS20].

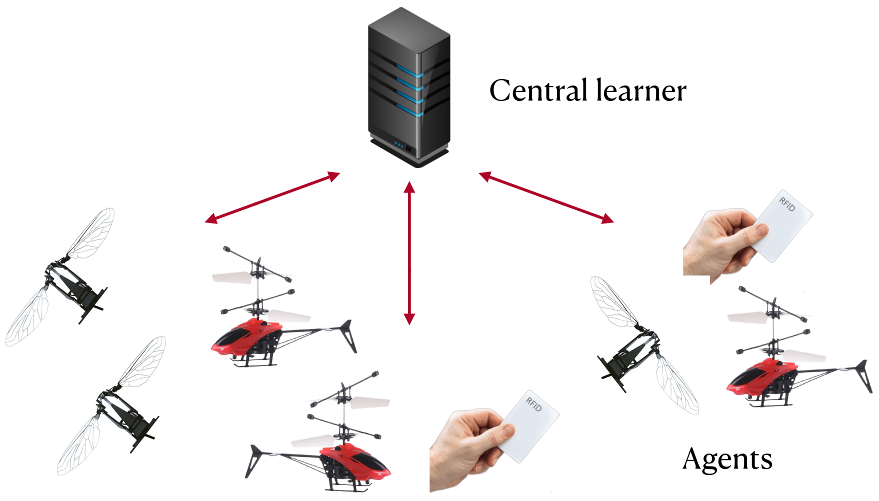


Figure 4.1: A central learner collects rewards from a set of agents. The agents can join and leave at any time and hence can be different and unaware of the historical rewards, (i.e., they are memoryless).

not to perform more involved cooperative operations.

Our main contribution is a set of upper and lower bounds on the required number of bits to achieve the same (as in unquantized communication) regret bound up to a small constant factor. In particular, our lower bound states that it is necessary to send at least 1.9 bits per reward to achieve a regret bound within a factor of 1.5 from the regret bound of unquantized algorithms. Our upper bounds state that, on average, 3.4 bits are sufficient to maintain a regret bound within a factor of 1.5 from the unquantized regret bound.

The upper bounds are proved constructively using a novel quantization scheme, that we term $QuBan$, that is tailored to compressing MAB rewards, and can be applied on top of existing MAB algorithms. $QuBan$ only cares to maintain what matters to the MAB algorithm operation, namely the ability to decide which is the best arm. At a high level, $QuBan$ maps rewards to quantization levels chosen to be dense around an estimate of the arm's mean values and sparse otherwise. $QuBan$ employs a stochastic correction term that enables to convey an unbiased estimate of the rewards with a small variance. It also introduces a simple novel rounding trick to guarantee that the quantization error is conditionally independent of the history given the current pulled arm index. This maintains the Markov property which is crucial in the analysis of bandit algorithms and enables reusing the same analysis methods as for unquantized rewards to bound the regret after quantization. Finally, $QuBan$ encodes the reward values that occur more frequently with shorter representations, in order to reduce the number of bits communicated. Numerical results corroborate that $QuBan$, applied on top of MAB algorithms such as UCB and ϵ-greedy, uses a few bits (as small as 3) to achieve the same regret as unquantized communication.

4.2.1 Related Work

To the best of our knowledge, for the distributed dynamic model we consider, no scheme in the literature can be used to solve the problem of maintaining a regret bound that matches the unquantized regret bound, up to a small constant factor, while using a few bits of communication. In the following, we distinguish our work from a representative sample of existing literature.

MAB algorithms. There is a long line of research in the literature about MAB algorithms. For instance, explore-then-commit [Rob52, Ans63], ϵ-greedy [ACF02], Thomson sampling [Tho33], the upper confidence bound [Lai87, ACF02] and its variant for contextual bandits [DHK08, LCL10]. Under the assumption that the reward distributions are 1-subgaussian, these algorithms provide a worst-case regret bound that is almost $O(\sqrt{T})$. The explore-then-commit regret is upper bounded by $C\sqrt{T}$ for bandits with 2 arms and known minimum means gap, while the regret of ϵ-greedy is upper bounded by $C'\sqrt{kT}$ for k-armed bandits with knowledge of the minimum means gap, where C, C' are constants that do not depend on k, T [LS20]. Thomson sampling and UCB achieve a regret with upper bound $C\sqrt{kT\log(T)}$ for k-armed bandits, where C is a constant that does not depend on k, T [RV14, AG12, AG13a, KR95, Agr95, ACF02]. Recall from Chapter 2 that for contextual linear bandits, the best known (frequentist) regret upper bound is $\tilde{O}(d\sqrt{T})$, where d is the dimension of an unknown system parameter, achieved by LinUCB [LS20, DHK08]. This matches a lower bound (for any algorithm) provided in [LWZ19] up to $\log$ factors. If focusing on Contextual Thompson sampling algorithm, the best known frequentist regret upper bound is $\tilde{O}(d^{3/2}\sqrt{T})$ [AG13b, AL17], and the best known Bayesian regret upper bound is $\tilde{O}(d\sqrt{T})$ [RV14]. These algorithms assume access to a full precision reward at each iteration. Our goal is not to replace the existing MAB algorithms to deal with quantized rewards; instead, we are interested in a general quantization framework that can be applied on top of any existing (or future) MAB algorithm.

Compression for ML and distributed optimization. There is a number of research results targeting reducing the communication cost of learning systems using compression. For instance, compression is applied on gradient updates [SFD14, AGL17, MT20, HEF21]. Recent work has also looked at compression for classification tasks [HES20]. However, compression schemes tailored to active learning, such as MAB problems, have not been explored. Our quantization scheme can be understood as a reward compression scheme that reduces the communication complexity for MABs. The main difference between the quantization for MABs and for distributed learning is that the later targets reducing the dependency of the number of bits and performance on the dimensionality of bounded training data, which can be in the order of tens of millions. In contrast, the rewards of MABs are scalars. The main challenge of our setting is to deal with a reward distribution that is either unbounded or the upper bound on the reward is much larger than the noise variance, which are typical in many MAB applications. This can be done by exploiting the fact that the rewards are more likely to be picked from the arm that appear to be best. Such a property is not applicable in the general distributed optimization setup and comes with new challenges as will discussed later.

Sample complexity. Compression is related to sample complexity [EMM02, MT04]: indeed, sending a small number of samples, reduces the overall communication load. However, the question we ask is different (and complementary): sample complexity asks how many (full precision) samples from each distribution do we need to draw; we are asking, how many bits of each sample do we really need to transmit, when we only care to decide the best arm and not to reconstruct the samples.

Distributed multi-agent MAB. Researchers have explored the distributed multi-agent MAB problem with a single [AVW87] or multiple [SRJ17] decision makers; in these settings, distributed agents pick arms under some constraints (all agents pick the same arm [SRJ17], at most one agent can pick the same arm at a time otherwise no reward is given [AMT11] and other constraints [Lan19]). The agents cooperate to aggregate their observed rewards so as to jointly make a more informed decision on the best arm. Most of the works do not take into account communication constraints, and rather focus on cooperation/coordination schemes. Our setup is different: we have a single learner (central server) and simple agents who do not learn (do not keep memory) but simply observe and transmit rewards, one at a time. Our scheme can be potentially applied to these settings to reduce communication cost.

Within this previous framework, some work considers "batched" rewards, where agents keep their observed rewards in memory and communicate them infrequently, potentially summarizing their findings and thus reducing the communication load [PRC16, EKM19, EMM06]. Such schemes require agents to be present for the whole duration of learning, and can also not be implemented by memory/computation limited agents.

Independently and in parallel to ours, the work in [VSS20] also considered MAB learning with reduced number of bits, restricted in their case to UCB policies. Their main result shows that for rewards supported on $[0, 1]$, one bit of communication is sufficient; our work recovers this result using a much simpler approach as a special case of Section 4.4. Additionally, our work applies on top of any MAB algorithm, and for unbounded rewards.

Chapter Organization. Section 4.3 presents our model and notation; Section 4.4 looks at a special case; Section 4.5 describes $QuBan$; Section 4.6 presents our main theorems and Section 4.7 provides numerical evaluation.

4.3 Model and Notation

MAB Framework. We consider a multi-armed bandit (MAB) problem over a horizon of size T [Rob52]. At each iteration $t = 1, ..., T$, a learner chooses an arm (action) A_t from a set of arms $\mathcal{A}_t$ and receives a random reward r_t distributed according to an unknown reward distribution with mean μ_{A_t}. The reward distributions are assumed to be σ^2-subgaussian [BLM13]. The arm selected at time

t depends on the previously selected arms and observed rewards $\mathcal{A}_1, A_1, r_1, ..., \mathcal{A}_{t-1}, A_{t-1}, r_{t-1}, \mathcal{A}_t$. The learner is interested in minimizing the expected regret $R_T = \mathbb{E}[R'_T]$, where R'_T is the regret defined as

$$R'_T = \Sigma_{t=1}^T (\mu_t^* - r_t), \tag{4.1}$$

where $\mu_t^* = \max_{A \in \mathcal{A}_t} \mu_A$. The expected regret captures the difference between the expected total reward collected by the learner over T iterations and the reward if we would collect if we play the arm with the maximum mean (optimal arm).

Notation. When the set of arms $\mathcal{A}_t$ is finite and does not depend on t: we denote the number of arms by $k = |\mathcal{A}_t|$, the best arm mean by μ^*, and the gap between the best arm and the arm-i mean by $\Delta_i := \mu^* - \mu_i$. If X, Y are random variables, we refer to the expectation of X, variance of X, conditional expectation of X given Y, and conditional variance of X given Y as $\mathbb{E}[X]$, $\sigma^2(X)$, $\mathbb{E}[X|Y]$, and $\sigma^2(X|Y)$ respectively.

System Setup. We are interested in a distributed setting, where a learner asks at each time a potentially different agent to play the arm A_t; the agent observes the reward r_t and conveys it to the learner over a communication constrained channel, as depicted in Fig. 4.1. In our setup, each agent needs to immediately communicate the observed reward (with no memory), using a quantization scheme to reduce the communication cost. As learning progresses, the learner is allowed to refine the quantization scheme by broadcasting parameters to the agents they may need. We do not count these broadcast (downlink) transmissions in the communication cost since the learner has no restrictions in its power. We stress again that the agents cannot store information of the reward history since they may *join and leave* the system at any time. We thus opt to use a setting where the agents have no memory. This setting allows to support applications with simple agents (e.g. RFID applications and embedded systems).

Quantization. A quantizer consists of an encoder $\mathcal{E} : \mathbb{R} \to \mathcal{S}$ that maps $\mathbb{R}$ to a countable set $\mathcal{S}$, and a decoder $D : \mathcal{S} \to \mathbb{R}$. At each time t, the agent that observes the reward r_t transmits a finite length binary sequence representing $\mathcal{E}(r_t)$ to the learner which in turn decodes it using the decoder D to obtain the quantized reward $\hat{r}_t = D(\mathcal{E}(r_t))$. The range of a decoder is referred to as the set of quantization levels; the encoding and decoding operation of a quantizer maps the reward to a quantization level. We next describe a specific quantization module that we will use.

Stochastic Quantization (SQ). A stochastic quantizer that uses quantization levels in a set $\mathcal{L}$, which is a form of dithering [GS93, AGL17], consists of a randomized encoder $\mathcal{E}_{\mathcal{L}}$ and decoder $D_{\mathcal{L}}$ modules that can be described as following. The encoder $\mathcal{E}_{\mathcal{L}}$, that uses the set of quantization levels $\mathcal{L} = \{\ell_i\}_{i=1}^{2^B}$, takes as input a value x in $[\ell_1, \ell_{2^B}]$; it maps x to a level index described by B bits. The decoder, that uses the set of quantization levels $\mathcal{L} = \{\ell_i\}_{i=1}^{2^B}$, takes as input an index in $\{1, ..., 2^B\}$, and outputs the corresponding level value. Precisely,

$$\begin{aligned} &i(x) = \max\{j|\ell_j \leq x \text{ and } j < 2^B\}, \\ &\mathcal{E}_{\mathcal{L}}(x) = \begin{cases} i(x) & \text{with probability } \frac{\ell_{i(x)+1}-x}{\ell_{i(x)+1}-\ell_{i(x)}} \\ i(x)+1 & \text{with probability } \frac{x-\ell_{i(x)}}{\ell_{i(x)+1}-\ell_{i(x)}} \end{cases}, \\ &D_{\mathcal{L}}(j) = \ell_j, j \in \{1, ..., 2^B\}. \end{aligned} \tag{4.2}$$

That is, if x is such that $\ell_i \leq x < \ell_{i+1}$, then the index i is transmitted with probability $\frac{\ell_{i+1}-x}{\ell_{i+1}-\ell_i}$ (and x is decoded to be ℓ_i) while the index $i+1$ is transmitted with probability $\frac{x-\ell_i}{\ell_{i+1}-\ell_i}$ (and x is decoded to be ℓ_{i+1}).

The analysis of bandit algorithms leverages the fact that conditioned on A_t, the communicated reward r_t is an unbiased estimate of the mean μ_{A_t}. It is not difficult to see that SQ preserves this property, namely conditioned on A_t, it conveys to the learner an unbiased estimate of μ_{A_t}.

Performance Metric B_T, $\bar{B}(T)$. Among the schemes that achieve a regret matching the unquantized regret, up to a fixed small constant factor, our performance metrics are the *instantaneous* and *average* number of communication bits per reward B_T, and $\bar{B}(T)$ respectively. Let B_t be the number of bits used to transmit $\hat{r}_t$, and define the average number of bits after T iterations of the algorithm as $\bar{B}(T) = \frac{\sum_{t=1}^{T} B_t}{T}$. Our goal is to design quantization schemes that achieve expected regret matching the expected regret of unquantized communication (up to a small constant factor) while using a small number of bits B_T, and $\bar{B}(T)$.

4.4 A Case Where 1 Bit is Sufficient

In this section we show that there exist some “easy" cases where we can use just one bit per reward and a very simple quantization scheme. Note that one bit is a trivial lower bound on the instantaneous number of bits communicated B_T, since each agent needs to respond to the learner for each observed reward. We also note that by definition of the average number of bits $\bar{B}_T$ as the average of $B_1, ..., B_T$, one bit is also a lower bound on the average number of bits. Consider the case where the rewards are supported on $[0, 1]$ and all reward distributions have the same variance σ (but different means). Since $r_t \in [0, 1]$, its variance is upper bounded by $\frac{1}{4}$; we will here assume this worst case variance $\sigma^2(r_t|A_t) \approx 1/4$[3]. We will use 1-bit Stochastic Quantization (SQ), as in (4.2). The stochastic 1 bit quantizer takes r_t as input and interprets it as probability: outputs 1 with probability r_t and 0 with probability $1 - r_t$. Let $\hat{r}_t$ be the (binary) quantized reward, we then have that

$$\mathbb{E}[\hat{r}_t|A_t] = \mathbb{E}[\mathbb{E}[\hat{r}_t|r_t, A_t]|A_t] = \mathbb{E}[r_t|A_t] = \mu_{A_t}. \tag{4.3}$$

Recall that for bandit algorithms the expected regret scales linearly with the variance. For example, the UCB algorithm (c.f., [LS20]) with unquantized rewards, achieves $R_T \leq C\sigma\sqrt{Tk\log(T)}$ for a constant C that does not depend on k, T.

Proposition 1. *UCB with $r_t \in [0, 1]$ that uses 1-bit SQ achieves a regret $R_T \leq C\sqrt{Tk\log(T)}$.*

Proof. The proof follows directly from the case of reward distributions that are supported on $[0, 1]$ in [ACF02]. It follows a standard analysis based on confidence intervals by bounding the regret conditioning on the good event $\mathcal{G} = \{|\frac{\sum_{i=1}^t r_i \mathbf{1}\{A_i=A_t\}}{\sum_{i=1}^t \mathbf{1}\{A_i=A_t\}} - \mu_{A_t}| \leq \frac{C\log(T)}{\sqrt{\sum_{i=1}^t \mathbf{1}\{A_i=A_t\}}} \forall t = 1, ..., T\}$ which is shown to hold with probability at least $1 - \frac{1}{T}$. Assuming $\mathcal{G}$, it can be shown that the total number of pulls for an arm with gap Δ_i, according to the UCB rule, is $O(\frac{\log(T)}{\Delta_i^2})$ resulting in a regret

[3]A similar analysis extends, showing that UCB with 1-bit SQ achieves a regret within a small constant factor from the unquantized regret, when the variances differ but they are all close to $\frac{1}{4}$.

that is bounded as

$$R_T \leq \mathbb{E}[R'_T|\mathcal{G}] + \mathbb{E}[R'_T|\mathcal{G}^C]\mathbb{P}[\mathcal{G}^C]$$
$$\leq T\Delta + \sum_{i:\Delta_i > \Delta} \frac{C\log(T)}{\Delta_i} + T\frac{1}{T} \leq T\Delta + \frac{Ck\log(T)}{\Delta} + 1. \tag{4.4}$$

The result follow by maximizing over Δ. $\square$

Simulation results, in this section and Section 4.7, verify that, for $r_t \in [0,1]$, 1-bit SQ performs very close to unquantized rewards.

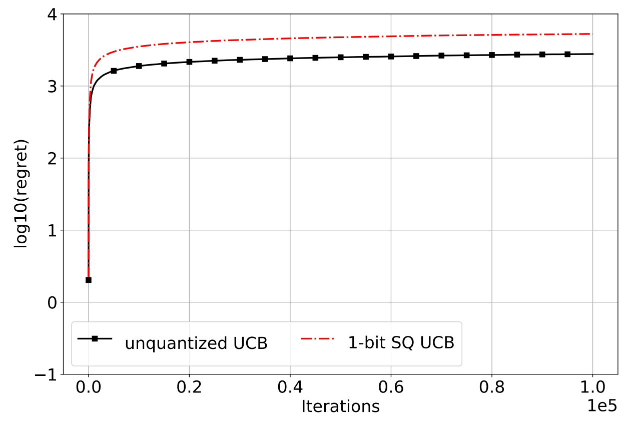
To motivate our general quantization scheme, we consider a case where 1-bit SQ results in a potentially large performance loss. Assume that the variance, σ, is much smaller than the range of r_t: $r_t \in [-\lambda, \lambda]$ and $\sigma = 1$, where $\lambda \gg 1$ is a parameter known to the learner. The 1-bit SQ maps r_t to either λ or $-\lambda$; it is not difficult to see that we still have $\mathbb{E}[\hat{r}_t|A_t] = \mu_{A_t}$, but $R_T \leq C\lambda\sqrt{kT\log(T)}$, where C is a constant that does not depend on T, k [ACF02][4]. This can be seen by observing that the expected regret can be written as $R_T = \sum_{t=1}^{T}\mathbb{E}(\mu_t^* - r_t) = \sum_{t=1}^{T}\mathbb{E}(\mu_t^* - \hat{r}_t) = 2\lambda\sum_{t=1}^{T}\mathbb{E}(\frac{(\mu_t^*+\frac{1}{2})-(\hat{r}_t+\frac{1}{2})}{2\lambda})$, which transforms the problem to one with reward distributions supported on $[0,1]$. Thus the expected regret bound grows linearly with λ, which can be arbitrarily large. In contrast, without quantization UCB achieves $C'\sqrt{kT\log(T)}$, where C' is another constant of the same order of C.

Simulation results verify that the convergence to the unquantized case can be slow. Fig. 4.2 shows the regret of unquantized and 1-bit SQ with the UCB algorithm for the setup described in Section 4.4 with $\sigma = 1$ and clipped reward distributions that have support only on an interval $[-\lambda, \lambda]$, for $\lambda = 1$ and 100 respectively. As discussed, we observe a regret penalty when $\lambda \gg \sigma$.

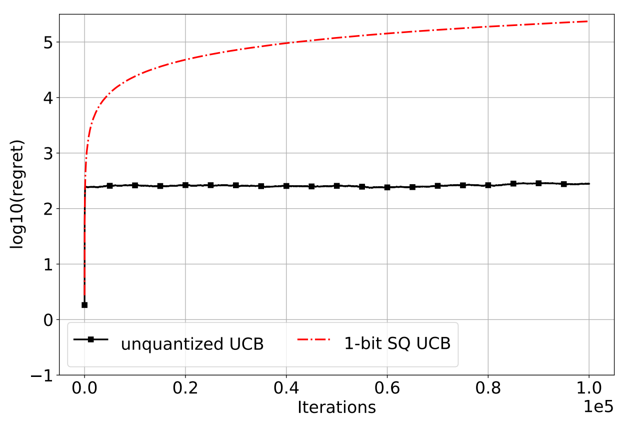
We take away the following observations:

• If the range λ is of the same order as the variance σ, 1-bit SQ is sufficient to preserve the regret bound up to a small constant factor.

[4]We note that this bound cannot be improved using techniques in [ACF02], since it is possible that $\sigma^2(\hat{r}_t|A_t) = \lambda^2$ (e.g., if $r_t = 0$ almost surely).



(a) $\lambda = 1$.



(b) $\lambda = 100$.

Figure 4.2: Reward Compression Example: Regret versus number of iterations. We use $\sigma = 1$.

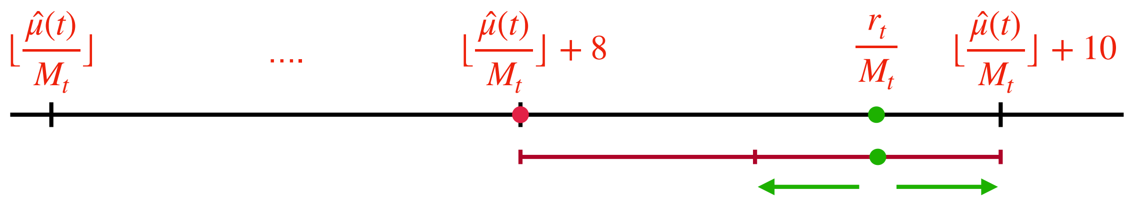
• If the range λ is much larger than σ, 1-bit SQ leads to a regret penalty proportional to $\frac{\lambda}{\sigma}$; thus we may want to only perform stochastic quantization within intervals of size similar to σ.

• In our discussion up to now we assumed that the rewards r_t are bounded almost surely. This is not true in general; we would like an algorithm that uses a small average number of bits even when the reward distributions are unbounded.

In the next section we introduce $QuBan$, that achieves a small average number of bits in all of the above cases.

4.5 $QuBan$: A General Quantizer for Bandit Rewards

In this section, we propose $QuBan$, an adaptive quantization scheme that can be applied on top of any MAB algorithm. Our scheme maintains attractive properties (in particular, the Markov property, unbiasedness, and bounded variance) for the quantized rewards that enable to retain the same regret bound as unquantized communication for the vast majority of MAB algorithms, while using a few bits for communication (simulation results show convergence to ~ 3 bits per iteration for T that is sufficiently large, see Section 4.7).



$\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor$
....
$\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor + 8$
$\frac{r_t}{M_t}$
$\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor + 10$

Figure 4.3: Illustration of $QuBan$ Algorithm. In the shown example, r_t is mapped to a value of the red dot (conveyed with the index $I_t = 4$), and stochastically to one of the two nearest quantization levels depicted on the red line.

$QuBan$ uses ideas that include: (i) centering the quantization scheme around a value that is believed to be close to the picked arm mean in the majority of iterations; (ii) maintaining a quantization error that is conditionally independent on previously observed rewards given the arm selection, which is achieved by choosing the quantization center to be an integer value (illustrated in more detail in the proof of Theorem 2); (iii) assigning shorter codes to the values near the quantization center and otherwise longer codes to maintain a finite expected number of bits even if the reward distribution has infinite support; and (iv) using stochastic quantization to convey an unbiased estimate of the reward. We next describe $QuBan$ in more detail.

$QuBan$ Centers the Quantization Around a Value $\hat{\mu}(t)$

Recall that at time t the learner selects an action A_t and needs to convey the observed reward r_t. As we expect r_t to be close to the mean μ_{A_t}, we would like to use quantization levels that are dense around μ_{A_t} and sparse in other areas. Since μ_{A_t} is unknown, we estimate it using some function of the observed rewards that we term $\hat{\mu}(t)$; we can think of $\hat{\mu}(t)$ as specifying a “point" on the real line around which we want to provide denser quantization.

Choices for $\hat{\mu}(t)$. In this work, we analyze the following three choices for $\hat{\mu}(t)$, the first two applying to MAB with a finite fixed set of arms, while the third to linear bandits.

- **Average arm point (Avg-arm-pt):** $\hat{\mu}(t) = \hat{\mu}_{A_t}(t-1)$. We use $\hat{\mu}_{A_t}(t-1)$, the average of the samples picked from arm A_t up to time $t-1$, as an estimate of μ_{A_t}.

• **Average point (Avg-pt)**: $\hat{\mu}(t) = \frac{1}{t-1}\sum_{j=1}^{t-1}\hat{r}_j$ (the average over all observed rewards). Here we can think of $\frac{1}{t-1}\sum_{j=1}^{t-1}\hat{r}_j$ as an estimate of the mean of the best arm. Indeed, the average reward of a well behaved algorithm will converge to the best mean reward.

These two choices of $\hat{\mu}(t)$ give us flexibility to fit different regimes of MAB systems. In particular, we expect the avg-arm-pt to be a better choice for a small number of arms k and MAB algorithms that achieve good estimates of μ_{A_t} (explore all arms sufficiently so that $\hat{\mu}_{A_t}(t-1)$ approaches μ_{A_t}). However, since the first time an arm is pulled we do not have an estimate of its mean, this results in possibly larger number of bits for the first pull; this penalty increases with the number of arms k. As our analysis also shows (see Section 4.6), if k is large, acquiring good estimates for all arms may be costly and not what good algorithms necessarily pursue; instead, the avg-pt has a simpler implementation, as it only requires to keep track of a single number, and still enables to distinguish well in the neighborhood of the best arm (connecting the number of bits to the regret), which is essentially what we mostly want.

• **Contextual bandit choice**: $\hat{\mu}(t) = \langle \theta_t, A_t \rangle$. Consider the widely used stochastic linear bandits model in Section 4.3. We observe that linear bandit algorithms, such as contextual Thomson sampling and LinUCB, choose a parameter θ_t believed to be close to the unknown parameter θ_*, and pick an action based on θ_t. For example, LinUCB [DHK08] chooses a confidence set $\mathcal{C}_t$ with center θ_t believed to contain θ_* and picks an action $A_t = \arg\max_{a\in\mathcal{A}_t}\max_{\theta\in\mathcal{C}_t}\langle \theta, a\rangle$. Accordingly, we propose to use $\hat{\mu}(t) = \langle \theta_t, A_t \rangle$. We note that our intuition for the avg-pt choice does not work for contextual bandits as it relies on that $\max_{a\in\mathcal{A}_t}\langle \theta_*, a\rangle$ is the same for all t, which might not hold in general. Likewise, the avg-arm-pt choice will not work as the set of actions $\mathcal{A}_t$ can be infinite or change with time.

We underline that the estimator $\hat{\mu}(t)$ is only maintained at the learner's side and is broadcasted to the agents. As discussed before, this downlink communication is not counted as communication cost.

$QuBan$ **Components**

As discussed, at iteration t, $QuBan$ centers its quantization around the value $\hat{\mu}(t)$. It then quantizes the normalized reward $\bar{r}_t = r_t/M_t - \lfloor \hat{\mu}(t)/M_t \rfloor$ to one of the two values $\lfloor \bar{r}_t \rfloor, \lceil \bar{r}_t \rceil$, where $M_t = \epsilon \sigma X_t$[5], ϵ is a parameter to control the regret vs number of bits trade-off as will be illustrated later in this section, and $\{X_t\}_{i=1}^T$ are independent samples from a $\frac{1}{4}$-subgaussian distribution satisfying $|X_t| \geq 1$ almost surely, e.g., we can use $X_t = 1$ almost surely[6]. If X_t is allowed to take larger values with some probability, it will result in coarser quantization with some probability, and a smaller number of bits. This introduces an error in estimating $\bar{r}_t$ that is bounded by 1, which results in error of at most M_t in estimating $r_t = M_t(\bar{r}_t + \lfloor \hat{\mu}(t)/M_t \rfloor)$. This quantization is done in a randomized way to convey an unbiased estimate of r_t. The encoding of $\hat{r}_t$ is a composition of: sign of $\bar{r}_t$, a unary encoding of the least power of 2 below $|\bar{r}_t|$ (denoted by 2^{I_t}), and SQ for $|\bar{r}_t| - 2^{I_t}$[7]. The unary encoding of I_t consists of I_t zeros followed by 1 one. Both the unary encoding and the SQ use $O(\log(\bar{r}_t))$ bits. We recall that $\frac{\hat{\mu}(t)}{M_t}$ is believed to be close to $\frac{r_t}{\sigma}$ in the majority of iterations resulting in small values for $\log(\bar{r}_t)$.

The precise learner and agent operations used for $QuBan$ are presented in pseudo-code in Algorithms 7 and 8 (see Fig. 4.3 for an example), respectively. The learner at each iteration broadcasts $\hat{\mu}(t)$ and asks one of the agents available at time t to play an action A_t. Initially, since we have no knowledge about μ_i, the learner assumes that $\hat{\mu}(0) = 0$. The agent that plays the action uses the observed r_t together with $\hat{\mu}(t)$ it has received to transmit three values we term (b_t, I_t, e_t), to the learner, as described in Algorithm 2 using $O(\log(|\bar{r}_t|))$ bits.

Rounding of $\hat{\mu}(t)/M_t$: the reason for choosing the quantization to be centered around $\lfloor \hat{\mu}(t)/M_t \rfloor$ instead of $\hat{\mu}(t)/M_t$ is to guarantee that the distance between r_t and the two closest quantization

[5]The case where σ is unknown is discussed in Section 4.6.

[6]For our proofs we set $X_t = 1$ for simplicity; more sophisticated choices can further improve the upper bounds such as X_t picked from a Gaussian distribution.

[7]Note that $0 \leq |\bar{r}_t| - 2^{I_t} \leq 2^{I_t}$.

Algorithm 7 Learner operation with input MAB algorithm Λ

```
1: Initialize: $\hat{\mu}(1) = 0$
2: for $t = 1, ..., T$ do
3:     Choose an action $A_t$ based on the bandit
4:       algorithm $\Lambda$ and ask the next agent to play it
5:     Send ${M_t}^8$, $\hat{\mu}(t)$ to an agent
6:     Receive the encoded reward $(b_t, I_t, \mathcal{E}_{\mathcal{L}_t}(e_t))$ (see
7:       Algorithm 8)
8:     Decode $\hat{r}_t$:
9:     if length($b_t$)$\leq 3$ then
10:        $\hat{r}_t$ can be decoded using a lookup table
11:    else
12:        Decode the sign, $s_t$, of $\hat{\hat{r}}_t$ from $b_t$
13:        Set $\ell_t$ to be the $I_t$-th element in the set
14:        $\{0, 2^0, ...\}$
15:        Set $\mathcal{L}_t = \{\ell_t, \ell_t + 1, ..., \max\{2\ell_t, \ell_t + 1\}\}$
16:        Define $e_t^{(q)} = D_{\mathcal{L}_t}(\mathcal{E}_{\mathcal{L}_t}(e_t))$
17:        $\hat{r}_t = (s_t(e_t^{(q)} + \ell_t + 3.5s_t + 0.5) + \lfloor \hat{\mu}(t)/M_t \rfloor)M_t$
18:    Calculate $\hat{\mu}(t + 1)$ (using one of the discussed
19:      choices)
20:    Update the parameters required by $\Lambda$
```

Algorithm 8 Distributed Agent Operation

```
1: Inputs: $r_t$, $\hat{\mu}(t)$ and $M_t$
2: Set $L = \{\lfloor \bar{r}_t \rfloor, \lceil \bar{r}_t \rceil\}$, $\hat{\bar{r}}_t = D_L(\mathcal{E}_L(\bar{r}_t))$, where $\bar{r}_t = r_t/M_t - \lfloor \hat{\mu}(t)/M_t \rfloor$.
3: Set $b_t$ with three bits to distinguish between the $8$ cases: $\hat{\bar{r}}_t < -2, \hat{\bar{r}}_t > 3, \hat{\bar{r}}_t = i, i \in$
   $\{-2, -1, 0, 1, 2, 3\}$. This implicitly encodes the sign of $\hat{\bar{r}}_t$, which we denote $s_t$.
4: if $|\hat{\bar{r}}_t| > |a|$ and $\hat{\bar{r}}_t a > 0$, $a \in \{-2, 3\}$ then
5:     Augment $b_t$ with an extra one bit to indicate if $|\hat{\bar{r}}_t| = |a| + 1$ or $|\hat{\bar{r}}_t| > |a| + 1$.
6:     if $|\hat{\bar{r}}_t| > |a| + 1$ then
7:         Let $L' = \{0, 2^0, ...\}$
8:         Set $\ell_t = \max\{j \in L' | j \leq |\bar{r}_t| - (|a| + 1)\}$
9:         Encode $\ell_t$ by $I_t - 1$ zeros followed by a one (unary coding), where $I_t$ is the index of
10:        $\ell_t$ in the set $L'$.
11:        Let $e_t = |\bar{r}_t| - (|a| + 1) - \ell_t$
12:        Set $\mathcal{L}_t = \{\ell_t, \ell_t + 1, ..., \max\{2\ell_t, \ell_t + 1\}\}$
13:        Encode $e_t$ using SQ to get $\mathcal{E}_{\mathcal{L}_t}(e_t)$
14: Transmit $(b_t, I_t, \mathcal{E}_{\mathcal{L}_t}(e_t))$
```

levels is independent of $\hat{\mu}(t)$[9] (which is dependent on $\hat{r}_1, ..., \hat{r}_{t-1}$). As we discuss in the following section, this preserves the Markov property (given A_t, the quantized reward $\hat{r}_t$ is conditionally independent on the history $A_1, \hat{r}_1, ..., A_{t-1}, \hat{r}_{t-1}$), a property that is exploited in the analysis of bandit algorithms to guarantee that $|\sum_{t=1}^{T} \hat{r}_t - \mu_{A_t}/T|$ approaches zero in some probabilistic sense as T increases.

Sending the least power of 2 below $\bar{r}_t$: For simplicity we consider the case where $\bar{r}_t \geq 0$. We note that since it is possible for the decoded reward to take any value in the set $\{\lfloor \frac{\hat{\mu}}{M_t} \rfloor, \lfloor \frac{\hat{\mu}}{M_t} \rfloor + 1, \lfloor \frac{\hat{\mu}}{M_t} \rfloor + 2...\}$ (to guarantee the uniform upper bound on $|\hat{\bar{r}}_t - \bar{r}_t|$), every value in that set needs to be encoded. A good encoding strategy assigns shorter codes to the levels that are close to $\lfloor \frac{\hat{\mu}}{M_t} \rfloor$ as they are expected to occur more often. Hence, the best we can hope for is to encode r_t using $O(\log(\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}}{M_t} \rfloor)$ bits as it is quantized to either $\lfloor \frac{r_t}{M_t} \rfloor$ or $\lceil \frac{r_t}{M_t} \rceil$ and the quantization level at $\lfloor \frac{r_t}{M_t} \rfloor$ is encoded using the largest number of bits among the levels in the set $\{\lfloor \frac{\hat{\mu}}{M_t} \rfloor, \lfloor \frac{\hat{\mu}}{M_t} \rfloor + 1, \lfloor \frac{\hat{\mu}}{M_t} \rfloor + 2..., \lfloor \frac{r_t}{M_t} \rfloor\}$. As can be seen in Appendix C.2, sending the greatest power of 2 below $\bar{r}_t$ then quantizing the difference using SQ gives that r_t is encoded using $O(\log(\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}}{M_t} \rfloor)$ bits. This is achieved since I_t is $O(\log(\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}}{M_t} \rfloor)$ and the SQ uses $2^{I_t} + 1$ quantization levels. An alternative way to encode I_t is using integer compression, or recursively applying our scheme by using unary coding to transmit the largest $I_t^{(2)}$ with $2^{I_t^{(2)}} \leq I_t$ and then encode the difference $I_t - 2^{I_t^{(2)}}$ using $\log(1 + 2^{I_t^{(2)}})$ bits noting that $I_t - 2^{I_t^{(2)}} \leq 2^{I_t^{(2)}}$. This results in using $O(\log(\log(\frac{r_t}{\sigma} - \lfloor \frac{\hat{\mu}}{\sigma} \rfloor))$ bits to encode I_t. We keep the unary coding for I_t for simplicity and since it does not dominate the average number of bits.

Preserving regret bounds: The main reasons $QuBan$ preserves existing regret bounds is that it does not destroy the Markov property (as we prove in Appendix C.2) and it provides that $|\hat{\bar{r}}_t - \bar{r}_t|$ is uniformly upper bounded. The later property implies that if given A_t, r_t is conditionally integrable, sub-exponential, sub-gaussian, or almost surely bounded, then M_t can be chosen such that given A_t, $\hat{r}_t$ is conditionally integrable, sub-exponential, sub-gaussian, or almost surely bounded respectively. A widely used assumption is that given A_t, r_t is conditionally sub-gaussian.

[9]As will be shown in App. C.1, centering the quantization around any integer value implies that the two closest quantization levels to $\frac{r_t}{M_t}$ are $\lfloor \frac{r_t}{M_t} \rfloor, \lceil \frac{r_t}{M_t} \rceil$.

We observe that all the operations in $QuBan$ can be performed in a constant time except steps 9, 13 in Algorithm 8 which require $O(B_t)$ running time. As shown in Sections 4.6, 4.7, B_t is only a few bits on average resulting in a linear amortized running time.

4.6 Upper and Lower Bounds

In this section we present an upper bound on the number of bits used by $QuBan$ and show that it provides properties for the quantized reward that result in a regret within a small constant factor from the unquantized regret. We also present a lower bound, within 1.5 bits from the upper bound, on the number of bits needed to satisfy the required properties. Before stating the results, we state our assumptions.

Assumption 1. We assume that we are given:
(i) a MAB instance with σ^2-subgaussian[10] rewards where the Markov property holds: conditioned on the action at time t, the current reward is conditionally independent on the history (past actions and rewards).
(ii) a MAB algorithm Λ such that for any instance with σ^2-subgaussian rewards, and time horizon T, the algorithm's expected regret (with unquantized rewards) is upper-bounded by R_T^U.

We note that assumption (i) is standard for the analysis of MAB algorithms, while assumption (ii) essentially only introduces notation.

4.6.1 Upper Bounds

The following proposition gives an upper bound on the regret after quantization, and shows that for $\epsilon = 1$, the regret is within a factor of 1.5 from the regret of the unquantized case. The proof is provided in App. C.1.

[10]This is a standard assumption used for simplicity but is not required for our main results.

Proposition 1. Suppose Assumption 1 holds. Then, when we apply $QuBan$, the following hold:
1. Conditioned on A_t, the quantized reward $\hat{r}_t$ is $((1+\frac{\epsilon}{2})\sigma)^2$-subgaussian, conditionally independent on the history $A_1, \hat{r}_1, ..., A_{t-1}, \hat{r}_{t-1}$ (Markov property), and satisfies $\mathbb{E}[\hat{r}_t|A_t] = \mu_{A_t},\ |\hat{r}_t - r_t| \leq M_t$ almost surely ($t = 1, \ldots, T$).
2. The expected regret R_T is bounded as $R_T \leq (1+\frac{\epsilon}{2})R_T^U$, where ϵ is a parameter to control the regret vs number of bits trade-off.

In the following we provide an upper bound on the expected average number of bits. We also provide a high-probability upper bound on the instantaneous number of bits. For simplicity we only consider the case where $\epsilon = 1$ and discuss the other case in App. C.2. The proof is given in App. C.2. At a high level, to upper bound the regret after quantization we show that $QuBan$ maintains a number of desirable properties for the quantized rewards, namely, unbiasedness, and the fact that the quantized rewards are $(1.5\sigma)^2$-subgaussian and satisfy the Markov property. To upper bound the expected number of bits we use the fact that $QuBan$ assigns short representations for the rewards around an estimate of the mean, which we expect to see more frequently.

Theorem 11. *Suppose Assumption 1 holds. Let $\epsilon = 1$. There is a universal constant C such that:*
1. *For $QuBan$ with $\hat{\mu}(t) = \hat{\mu}_{A_t}(t-1)$ (avg-arm-pt), the average number of bits communicated satisfies that $\mathbb{E}[\bar{B}(T)] \leq 3.4 + (C/T)\sum_{i=1}^{k}\log(1+|\mu_i|/\sigma) + C/\sqrt{T}$.*
2. *For $QuBan$ with $\hat{\mu}(t) = \frac{1}{t-1}\sum_{j=1}^{t-1}\hat{r}_j$ (avg-pt), the average number of bits communicated satisfies $\mathbb{E}[\bar{B}(T)] \leq 3.4 + \frac{C}{T}\left(1 + \log(1+\frac{|\mu^*|}{\sigma}) + \frac{R_T}{\sigma} + \sum_{t=1}^{T-1}\frac{R_t}{(\sigma t)}\right) + C/\sqrt{T}$.*
3. *For $QuBan$ with $\hat{\mu}(t) = \langle\theta_t, A_t\rangle$ (stochastic linear bandit), the average number of bits communicated satisfies that $\mathbb{E}[\bar{B}](T) \leq 3.4 + C\mathbb{E}[\sum_{t=1}^{T}|\langle\theta_t - \theta_*, A_t\rangle|]/(\sigma T)$.*

In App. C.2 we also provide almost surely bounds on the asymptotic average number of bits, namely, $\lim_{T\to\infty}(1/T)\sum_{t=1}^{T}B_t \leq 3.4$ almost surely.

In the following we provide a high probability bound on the number of bits that $QuBan$ uses in each iteration. We analyze the performance for avg-arm-pt only; the other choices for $\hat{\mu}(t)$ can be handled similarly.

Theorem 12. *For a MAB instance with σ^2-subgaussian rewards, $QuBan$ with $\epsilon = 1, \hat{\mu}(t) = \hat{\mu}_{A_t}(t-1)$ (avg-arm-pt), satisfies that for t with $T_t(A_t) > 0$, where $T_t(i)$ is the number of pulls for arm i prior to iteration t, with probability at least $1 - \frac{1}{T}$ it holds that $\forall t \leq T$:*

$$B_t \leq 4 + \lceil \log(4\log(T)) \rceil + \lceil \log\log(4\log(T)) \rceil. \tag{4.5}$$

The proof is provided in App. C.3.

Remark 1. Using the previous lemma we can modify $QuBan$ to have that (4.5) is satisfied almost surely, by sending a random 1 bit when (4.5) is not satisfied. This will only add at most $T\sum_{i=1}^{k}\Delta_i$ regret with probability at most $\frac{1}{T}$. Hence, the expected regret is increased by at most a factor of 2.

Remark 2. Throughout the chapter, we assume a known upper bound on the noise variance. However, it is not difficult to see that a variance estimate within a constant factor would suffice. Running $QuBan$ with an estimate σ' that is possibly different from the true σ results in a degradation in the regret by a factor of $\max\{1, \frac{\sigma'}{\sigma}\}$ and increase in the communication by $2\log(\frac{\sigma}{\sigma'})$ bits. An optimistic estimate of the noise $\sigma' < \sigma$ results in finer quantization, hence, no degradation in the regret at the cost of increasing the communication by $2\log(\frac{\sigma}{\sigma'})$ bits.

4.6.2 Lower Bound

In this subsection we provide a lower bound showing that an average number of 1.9 bits per iteration are required to maintain a sublinear regret and a $(\frac{\sigma}{2})^2$-subgaussian quantization error, $\hat{r}_t - r_t$. We also show that the instantaneous number of bits cannot be almost surely bounded by a constant. In our lower bound, we focus on prefix free codes [Cov99]; a similar analysis can be performed for non-singular codes leading to different constants. We also note that our achievable scheme (Algorithms 7, 8) provides a prefix-free code. We first state the following lemma which shows that for the quantizer to preserve the sublinear regret of the bandit algorithm, Q needs to to satisfy that $\mathbb{E}[Q(r_t)|r_t] = c_1 r_t + c_2$, where c_1, c_2 are constants. Hence, by a proper shifting and scaling, the quantizer Q can be made unbiased, i.e., satisfying $\mathbb{E}[Q(r_t)|r_t] = r_t$.

Lemma 4. Let ALG be any algorithm for multi-arm bandits with sublinear regret and Q be (a possibly randomized) quantizer. Let R_T be the worst-case expected regret of ALG when using rewards $Q(r_1), ..., Q(r_t)$. If R_T is sublinear in T, then Q satisfies

$$\mathbb{E}[Q(r_t)|r_t] = c_1 r_t + c_2,$$

where c_1, c_2 are constants.

By the previous lemma, it suffices to only consider unbiased quantizers. We next state our lower bound theorem.

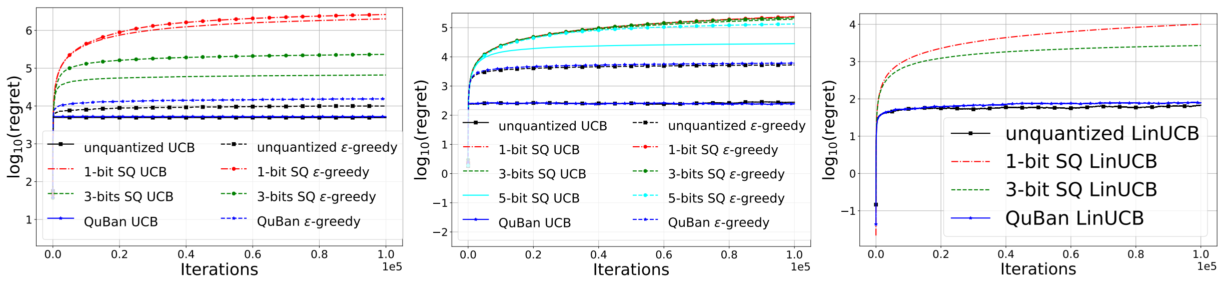
Theorem 13. *Any (possibly randomized) quantizer Q that uses prefix-free encoding and satisfies:*
$i.$ ***(Unbiased Property)*** $\mathbb{E}[Q(r_t)|r_t] = r_t$,
$ii.$ ***(SubGaussian Property)*** *Conditioned on* r_t, $Q(r_t) - r_t$ *is* $(\frac{\sigma}{2})^2$*-subgaussian (*$t = 1, \ldots, T$*),*
we have that there exist $(4\sigma)^2$*-subgaussian reward distributions for which:*
1. $(\forall b \in \mathbb{N})$ $(\exists t, \delta > 0)$ *such that* $\mathbb{P}[B_t > b] > \delta$.
2. $(\forall t > 0)$ $(\exists T > t)$ *such that* $\mathbb{E}[\bar{B}(T)] \geq 1.9$ *bits.*

The proofs are given in App. C.4.

4.6.3 Application to UCB, ϵ-greedy, and LinUCB

We here leverage Theorem 11 to derive bounds for three widely used MAB algorithms. We highlight that although the regret bounds hide constant factors, these constants are within 1.5 of the unquantized constants according to Theorem 11. The proofs are in App. C.5 for Corollaries 1 and 2 and in App. C.6 for Lemma 3.

Corollary 1. Assume we use $QuBan$ with avg-pt on top of **UCB** [ACF02] with σ^2-subgaussian reward distributions. Then there is a constant C that does not depend on T and k such that $R_T \leq C\sigma\sqrt{Tk\log(T)},\ \mathbb{E}[\bar{B}(T)] \leq 3.4 + C\sqrt{k\log(T)/T}$.



(a) Setup 1 (larger Δ_i values). (b) Setup 2 (smaller Δ_i values). (c) Setup 3 (linear bandits).

Figure 4.4: Regret versus number of iterations.

Corollary 2. Assume we use $QuBan$ with avg-pt on top of ϵ-greedy [ACF02] with σ^2-subgaussian reward distributions and constant gaps $\Delta_i\ \forall i$. Let $\epsilon_t = \min\{1, Ck/(t\Delta^2_{\text{min}})\}$, where $\Delta_{\text{min}} = \min_i\{\Delta_i|\Delta_i > 0\}$ and $C > 0$ is a sufficiently large universal constant. Then there exists a constant C' that does not depend on T and k such that $R_T \leq C'\sigma k \log(1+T/k)$, $\mathbb{E}[\bar{B}(T)] \leq 3.4 + C'(k\log^2(T)/T + 1/\sqrt{T})$.

To simplify the expressions, we include the dependency on μ^* and Δ_i in the constant C for Corollary 1 and respectively C' for Corollary 2.

Corollary 3. Assume we use $QuBan$ on top of **LinUCB** [DHK08], then there is a constant C that does not depend on T and d such that $R_T \leq Cd\sqrt{T}\log(T)$, $\mathbb{E}[\bar{B}(T)] \leq 3.4 + C\frac{d\log(T)}{\sqrt{T}}$.

4.7 Numerical Evaluation

We here present our numerical results.

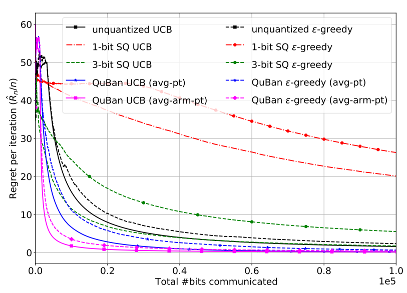
Quantization Schemes. We compare $QuBan$ against the baseline schemes described next.

Unquantized. Rewards are conveyed using the standard 32 bits representation.

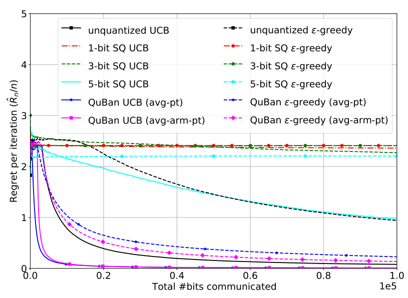
r-bit SQ. We implement r-bit stochastic quantization, by using the quantizer described in Section 4.3, with 2^r levels uniformly dividing a range $[-\lambda, \lambda]$.

$QuBan$. We implement $QuBan$ with $\epsilon = X_t = 1$.

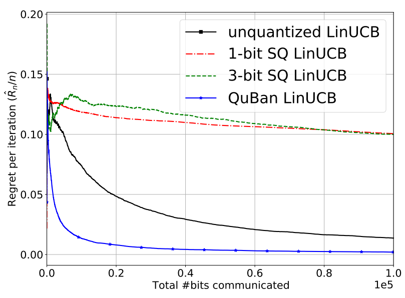
MAB Algorithms. We use quantization on top of:



(a) Setup 1 (larger Δ_i values).



(b) Setup 2 (smaller Δ_i values).



(c) Setup 3 (linear bandits).

Figure 4.5: Total number of bits versus regret per iteration.

(i) the UCB implementation in [LS20, chapter 8]. The UCB exploration constant is chosen to be σ_q, an estimate of the standard deviation of the quantized reward distribution.
(ii) the ϵ-greedy algorithm in [LS20, chapter 6], where ϵ_t is set to be $\epsilon_t = \min\{1, \frac{C\sigma_q k}{t\Delta^2_{\text{min}}}\}$.
(iii) the LinUCB algorithm for stochastic linear bandits in [LS20, chapter 19].

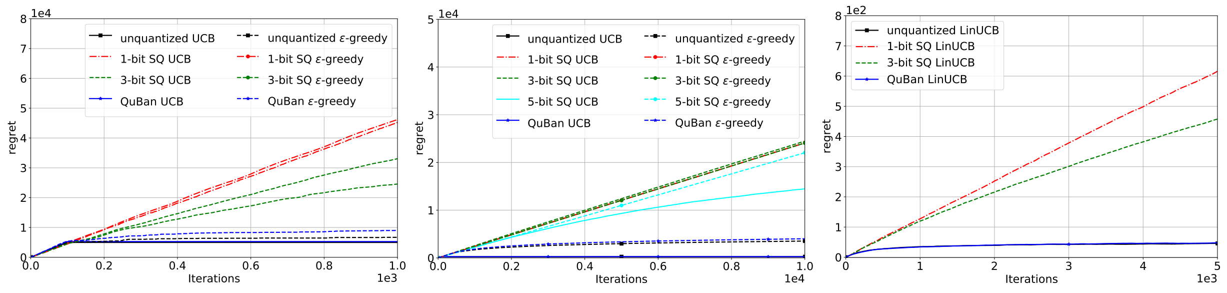
Synthetic Dataset:

MAB Setup. We simulate three cases. In each case we average over 10 runs of each experiment. The parameters σ_q, C are determined by the underlying MAB algorithm we use. In our simulations, we set σ_q to the variance of the quantized rewards (or best known upper bound), while C is chosen to be the value resulting in best regret among $\{0, \lambda/100, ..., \lambda\}$.

• **Setup 1: (Figs** $4-6$**(a)).** We use $k = 100, \lambda = 100, C = 10$, the arms' means are picked from a Gaussian distribution with mean 0 and standard deviation 10 and the reward distributions are conditionally Gaussian given the actions A_t with variance 0.1. The parameter σ_q is set to be 0.1 for $QuBan$ and $200/2^r - 1$ for the r-bit SQ.

• **Setup 2: (Figs** $4-6$**(b))** This differs from the previous only in that the means are picked from a Gaussian distribution with mean 95 and standard deviation 1 (leading to smaller Δ_i).

• **Setup 3: (Figs** $4-6$**(c)).** This is our contextual bandit setup. We use $d = 20$ dimensions, θ_* picked uniformly at random from the surface of a radius 1 ball centered at the origin, and the noise η_t is picked from a Gaussian distribution with zero mean and 0.1 variance. At each time t we construct



(a) Setup 1 (larger Δ_i values). (b) Setup 2 (smaller Δ_i values). (c) Setup 3 (linear bandits).

Figure 4.6: Regret versus number of iterations.

the actions set $\mathcal{A}_t$ by sampling 5 actions uniformly at random from the surface of a radius 0.5 ball centered at the origin independently of the previously sampled actions. We evaluate the regret and the average number of bits used by $QuBan$ as well as the 3 and 1 bit stochastic quantizers in the interval $[-10, 10]$ (the interval in which we observe the majority of rewards). These quantization schemes are used on top of the LinUCB algorithm. The LinUCB exploration constant is chosen to be σ_q, where σ_q is set to be 0.1 for $QuBan$ and $\frac{20}{2^r-1}$ for the r-bit SQ.

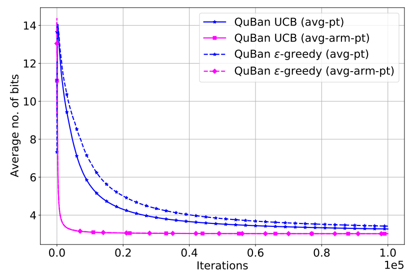
Results. Fig. 4.4 plots the regret R'_T in (4.1) vs. the number of iterations, Fig. 4.5 plots $\frac{\hat{R}_T}{T}$, the regret per iteration, vs. the total number of bits communicated, Fig. 4.6 plots the regret versus number of iterations, and Fig. 4.7 plots the average number of bits versus iterations. We find that:

• $QuBan$ in all three setups offers minimal or no regret increase compared to the unquantized rewards regret and achieves savings of tens of thousands of bits as compared to unquantized communication.

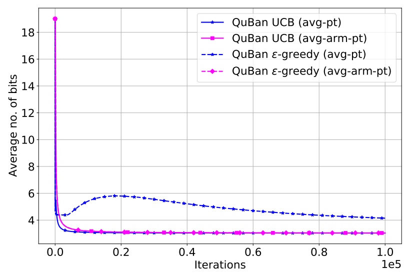
• 1-bit SQ significantly diverges in most cases; 3-bit and 5-bit SQ show better performance yet still not matching $QuBan$ with a performance gap that increases when the arms means are closer (Δ_i smaller), and hence, more difficult to distinguish.

• $QuBan$ allows for more than 10x saving in the number of bits over the unquantized case to achieve the same regret. In all three setups $QuBan$ achieves $\mathbb{E}[\bar{B}(T)] \approx 3$ (see Fig. 4.7).

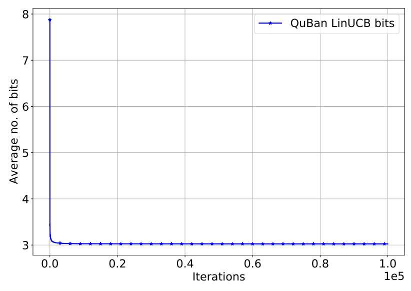
• Both $QuBan$ avg-pt and avg-arm-pt achieve the same regret (they are not distinguishable in Fig. 4.4 and thus we use a common legend), yet avg-arm-pt uses a smaller number of bits when the



(a) Setup 1 (larger Δ_i values).



(b) Setup 2 (smaller Δ_i values).



(c) Setup 3 (linear bandits).

Figure 4.7: Average number of bits versus iterations.

means of the arms tend to be well separated (Fig. 4.5(a)) while avg-pt uses a smaller number of bits when they tend to be closer together (Fig. 4.5(b)). We also observe that the avg-pt tends to perform better for a well-behaved bandit scheme, while the avg-arm-pt performs better when the algorithm picks sub-optimal arms for many iterations (e.g., ϵ-greedy in Fig. 4.7(b)).

Cryptocurrency Returns Dataset:

MAB Setup. In this part we compare the performance of our scheme against 3-bit SQ using multiple cryptocurrencies prices from binance.com in October 2021, where the reward is the investment return. The action represents which cryptocurrency, among {Bitcoin, Ethereum, Dogecoin, and Litecoin}, to buy then sell on the next day. The return for each currency is samples uniformly at random from the daily returns in the month of October 2021.

Results. In Fig. 4.8 we plot the regret (daily return - optimal average return) versus the number of iterations for the UCB algorithm using our quantization scheme and 3-bit SQ. We observe that the performance of $QuBan$ almost matches the unquantized performance (using ≈ 3 bits) while the regret of 3-bit SQ is linear for the used number of iterations.

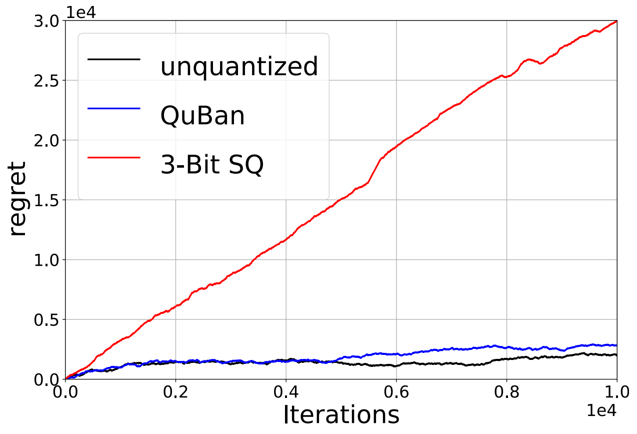


Figure 4.8: Regret versus number of iterations for the cryptocurrency prices dataset.

4.8 Conclusion and Open Research Directions

In this chapter, we provided a generic framework, $QuBan$, to quantize rewards for MAB problems. This framework can be used on top of nearly all the existing and future MAB algorithms, making them attractive for distributed learning applications where communication can become a bottleneck. We have demonstrated that, both in theory and by numerical experiments, $QuBan$ can provide very significant savings in terms of communication and barely affects the learning performance.

We identify several open research directions:

- How to close the gap between the upper and lower bounds on the number of bits, even in the asymptotic regime?
- How to exploit memory? In the setup we consider, the remote agents are changing over time, and thus they are essentially memoryless, i.e., a new agent does not know the history information of previous agents.
- How to deal with heavy tailed noise?

Resolving such questions can offer additional benefits for communication-sensitive bandit learning setups.

CHAPTER 5

The Common Information Dimension

5.1 Summary

Quantifying the common information between random variables is a fundamental problem with a long history in information theory. Traditionally, common information is measured in number of bits and thus such measures are mostly informative when the common information is finite. However, the common information between continuous variables can be infinite; in such cases, a real-valued random vector W may be needed to represent the common information, and to be used for instance for distributed simulation. In this chapter, we propose the concept of Common Information Dimension (CID) with respect to a given class of functions $\mathcal{F}$, defined as the minimum dimension of a random vector W required to distributively simulate a set of random vectors $X_1, \cdots, X_n$, such that W can be expressed as a function of $X_1, \cdots, X_n$ using a member of $\mathcal{F}$. We compute the common information dimension for jointly Gaussian random vectors in a closed form, with $\mathcal{F}$ being the linear functions class. We also analytically prove, under three different formulations, that the growth rate of common information in the nearly infinite regime is determined by the common information dimension, for the case of two Gaussian vectors.

5.2 Introduction

Quantifying the common information between random variables is a fundamental problem with a long history in information theory [Wat60, GK73, Wyn75, Bel03, YLC17], and has found application in diverse areas including source coding [GW74, XLC15, SG22], cryptography [AC93, AC98, Mau93, CN00] and multimodal learning [LYZ18, SDV21, RDN22, KAS23]. Multiple information theoretical notions have been developed to measure the common randomness, for instance, Gács-Körner's common information [GK73], Wyner's common information [Wyn75], common entropy [KLE14], and coordination capacity [CPC10] (see also the monograph [YT22]); but as far as we know, all of them measure common information in terms of bits. In this chapter, we introduce the notion of common information dimension, that uses dimensionality instead of bits to quantify the common randomness for continuous random variables, and establish a connection with the common information.

We will illustrate the necessity of a new notion of common randomness through an example. Let $X = [X_1, V]^\top, Y = [Y_1, V]^\top$ be two Gaussian random vectors with X_1, Y_1, V being independent scalar variables. The common information between X and Y is captured by V, which is a continuous scalar variable with infinite entropy - and thus the common information between X and Y, as calculated for instance in [SC15], is also infinite. For general Gaussian random vectors, we distinguish two cases: if rank$(\mathbf{\Sigma}_X)$ + rank$(\mathbf{\Sigma}_Y)$ $=$ rank$(\mathbf{\Sigma})$, where $\mathbf{\Sigma}_X = \mathbb{E}(XX^\top)$, $\mathbf{\Sigma}_Y = \mathbb{E}(YY^\top)$ and $\mathbf{\Sigma}$ is the joint covariance matrix of the vector $[X\ Y]$, then the common information can be described using a finite number of bits; while if rank$(\mathbf{\Sigma}_X)$ + rank$(\mathbf{\Sigma}_Y)$ $>$ rank$(\mathbf{\Sigma})$, the common information measured in bits becomes infinite (in this second case, we say that X and Y are *jointly singular*). To the best of our knowledge, there is no proposed metric that distinguishes between different amounts of common information in the infinite bits regime.

Our observation is that, for the latter case, a real-valued random vector W may be needed to represent the common randomness, and its dimension could provide guidance for practical applications. This is akin to compressed sensing [Don06, CRT06], where we seek a low dimensional representation in a high dimensional space.

Note that if we do not impose structural assumptions (on how W depends on variables $X_1, \ldots, X_n$), the minimum dimension cannot exceed one. This is due to the existence of measurable bijections between $\mathbb{R}$ and $\mathbb{R}^d$ for any $d \geq 1$. However, these functions are not implementable and unstable, as noted in [WV10], and hence, are not useful for applications. Therefore, regularity constraints on the common variable need to be considered. In particular, in our definition, we allow for a restriction of the form $W = g(X_1, \cdots, X_n)$ for some $g \in \mathcal{F}$, where $\mathcal{F}$ is a given class of functions. For example, $\mathcal{F}$ may be chosen to be the set of linear, or smooth functions. If $\mathcal{F}$ is chosen to contain all possible functions, then the minimum dimension of W will be upper bounded by one, as previously explained.

In addition to exploring what is the common information dimension of random variables, we consider the regime where the common information between random variables can be (or can approach) infinity and ask three questions: (1) How fast does the common information grow, from a finite to an infinite number of bits, as the dependency between variables increases? (2) How well can we "approximately" simulate a pair of random variables (X, Y) using a finite number of shared bits, even though their common randomness is infinite? and (3) If the target continuous random variables are quantized into discrete values, how large is the resulting common information between the quantized variables for a certain quantization resolution?

Contributions. Our main contributions in this chapter are as follows:

- We propose the concept of Common Information Dimension (CID), defined as the minimum dimension of a random variable, with respect to a class of functions, required to distributively simulate a set of random vectors $X_1, ..., X_n$. We define the Rényi common information dimension (RCID) as the minimum Rényi dimension of a random variable, with respect to a class of functions, required to distributively simulate $X_1, ..., X_n$. We define the Gács-Körner's common information dimension (GKCID) as the maximum Rényi dimension of a common function that can be extracted from each random variable individually.

- We prove that for jointly Gaussian random vectors and $\mathcal{F}$ being the class of linear functions, CID and RCID coincide, and GKCID is upper bounded by CID. Moreover, we give closed-form solutions for the CID, RCID, and GKCID in such case and an efficient method to construct W with the minimum dimension that enables the distributed simulation of $X_1, \cdots, X_n$. These can be computed by examining ranks of covariance matrices.

- We characterize the common information in the nearly infinite regime by considering a sequence of nearly singular Gaussian pairs with decreasing distances to a jointly singular target random variables X, Y. We prove in closed form that the common information of such sequences grows proportionally to $d(X, Y)$, which establishes a link between common information and the common information dimension.

- We define the approximate common information as the minimum amount of common information between random variables that approximate a target distribution within a given error. We prove that, in this case as well, the growth rate of the approximate common information is proportional to the common information dimension. Our result quantifies the number of shared bits needed to distributedly simulate jointly singular distributions within a desired accuracy. We illustrate this through numerical evaluations in Section 5.7.

- Moreover, we show that the common information between a uniformly quantized pair of Gaussian random variables also grows as a function of $d(X, Y)$. This result provides guidance on the number of shared bits needed for the distributed simulation under different quantization precisions.

Chapter organization. We review the related work and results on the common information of Gaussian vectors in Section 5.3. We introduce the definitions of common information dimension in Section 5.4. We present our results in calculating the common information dimension in Section 5.5. We state our problem formulation and results in the asymptotic behavior of common information in Section 5.6. We present our numerical evaluation in Section 5.7 and conclude the chapter in Section 5.8. We include proof outlines in the chapter and detailed proofs in the Appendix.

5.3 Related Work and Preliminaries

5.3.1 Common Information

Gács-Körner's common information [GK73] and Wyner's common information [Wyn75] are perhaps the most classical notions of common information. Gács-Körner's common information is defined as the maximum number of bits per symbol of the information that can be individually extracted from two dependent discrete variables X, Y, namely

$$C_{\text{GK}}(X, Y) := \max_{f,g: f(X)=g(Y)} H(f(X)), \tag{5.1}$$

where f and g are deterministic functions. However, it is known from [GK73,Wit75] that $C_{\text{GK}}(X, Y)$ equals zero except for one special case where $X = (X', V)$, $Y = (Y', V)$ and X', Y', V are independent variables.

Wyner's common information was originally defined for a pair of discrete sources $(X, Y) \sim \pi_{XY}$ as

$$C_{\text{Wyner}}(X, Y) := \min_{P_W P_{X|W} P_{Y|W}: P_{XY}=\pi_{XY}} I(X\,Y; W). \tag{5.2}$$

Wyner [Wyn75] provided two operational interpretations. One is for the *source coding* problem: the minimum common rate for the lossless source coding problem over the Gray-Wyner network, subject to a sum rate constraint. The other is for the *distributed simulation* problem: the minimum amount of shared randomness to simulate a given joint distribution π_{XY}. Recently, the works in [LXC10] and [CPC10] generalized Wyner's common information to n discrete random variables settings; and the works [VAR14, XLC15] and [LE17, YT20] generalized its interpretations to continuous sources in lossy source coding and distributed simulation, respectively.

Wyner's distributed simulation assumes codes of large block length (i.e., multi-shot) and approximate generation: the relative entropy between the generated distribution and the target distribution goes to zero as the block length goes to infinity. Kumar, Li and El Gamal [KLE14] extended Wyner's work to define the exact common information (also called common entropy) which requires a single-shot (i.e., block length 1) and exact generation of π_{XY}. The common entropy is defined as

$$G(X,Y) := \min_{P_W P_{X|W} P_{Y|W} : P_{XY} = \pi_{XY}} H(W). \tag{5.3}$$

To generalize this to the multi-shot (asymptotic) setting, they also defined the exact common information rate as

$$C_{\text{Exact}(X,Y)} := \lim_{n \to \infty} \frac{1}{n} G(X^n, Y^n). \tag{5.4}$$

The exact common information was extended to n continuous variables in [LE17], and was shown to provide an upper bound on Wyner's common information in [KLE14].

The calculation of Wyner's common information and its variants are challenging in general since it involves optimizing a concave function over a non-convex set. Therefore, closed-form solutions are available only for special cases [Wyn75, Wit76]. In particular, for continuous sources, closed-form solution is known only for Gaussian sources. [XLC15] calculated it for bivariate Gaussian and the multivariate one with certain correlation structure, while [SC15] and [SG22] extended it to a pair of Gaussian vectors. The general formula of Wyner's common information between Gaussian vectors $X \in \mathbb{R}^n$ and $Y \in \mathbb{R}^n$ is

$$C_{\text{Wyner}}(X,Y) = \frac{1}{2} \sum_{i=1}^{n} \log \frac{1+\rho_i}{1-\rho_i}, \tag{5.5}$$

where ρ_i's are the singular values of the normalized cross-covariance matrix $\mathbf{\Sigma}_X^{-1/2} \mathbf{\Sigma}_{XY} \mathbf{\Sigma}_Y^{-1/2}$, and $\mathbf{\Sigma}_X^{-1/2}, \mathbf{\Sigma}_Y^{-1/2}$ are defined using pseudo-inverse when needed. Observe that when X and Y are jointly singular (i.e., (5.7) holds and thus $\rho_i = 1$ for some i in (5.5)), the Wyner's common information $C_{\text{Wyner}}(X,Y)$ is infinite.

Wyner also describes two natural relaxations in [Wyn75]: (i) one replaces the conditional independence with a bounded conditional mutual information; (ii) the other allows a small distance between the generated and the target distributions, measured by Kullback–Leibler (KL) divergence. However, these were only analyzed in discrete settings. Recently, [SG22] studies the first relaxation in the case of Gaussian random variables. However, this version of the relaxed common information is still infinite when singular distributions are involved. In a separate study, [HLF22] explores a related, but different, problem of exchanging a small number of bits to break/reduce the dependency between distributed source. On the other hand, this chapter considers relaxation (ii) (with a different distance[1]) which allows an approximate generation when the sources can be continuous and the distributions may be singular.

5.3.2 Dimension

In our work, we consider two notions of dimension: the number of elements of a vector and the information dimension (also called Rényi dimension). The Rényi dimension is an information measure for random vectors in Euclidean space that was proposed by Rényi in 1959 [Ren59]. It characterizes the growth rate of the entropy of successively finer discretizations of random variables. The Rényi dimension of a random vector $W \in \mathbb{R}^{d_W}$ is defined as (when the limit exists)

$$d^R(W) = \lim_{m\to\infty} \frac{H(\langle W\rangle_m)}{\log m}, \tag{5.6}$$

where $\langle W\rangle_m$ is the element-wise discretization of W defined as $\langle W^{(i)}\rangle_m = \frac{\lfloor mW^{(i)}\rfloor}{m}$ and $H(V)$ is the entropy of V. Wu and Verdú [WV10] interpreted the Rényi dimension as the fundamental limit of almost lossless data compression for analog sources under regularity constraints that include linearity of the compressor and Lipschitz continuity of the decompressor.

5.4 Notation and Model

Notation. We use boldface capital letters to represent matrices and use capital letters to represent (vectors of) random variables, while P is specifically used to denote a probability distribution. For a random (vector) variable X, we use d_X to denote the number of its dimensions. For a matrix $\boldsymbol{M}$, we use $r(\boldsymbol{M})$ to denote the number of its rows. We use $X = [X_1, \ldots, X_n]$ when X_i's are column vectors to refer to $X = [X_1^\top \ \ldots \ X_n^\top]^\top$. We use $[X_1 \perp\!\!\!\perp \ldots \perp\!\!\!\perp X_n|W]$ to abbreviate that $X_1, \cdots, X_n$ are conditionally independent given W. The entropy of a random variable V is denoted

[1]Note that the KL divergence between any singular and non-singular distributions is always infinite, it is not suitable for the task of approximating a singular distribution with a non-singular one.

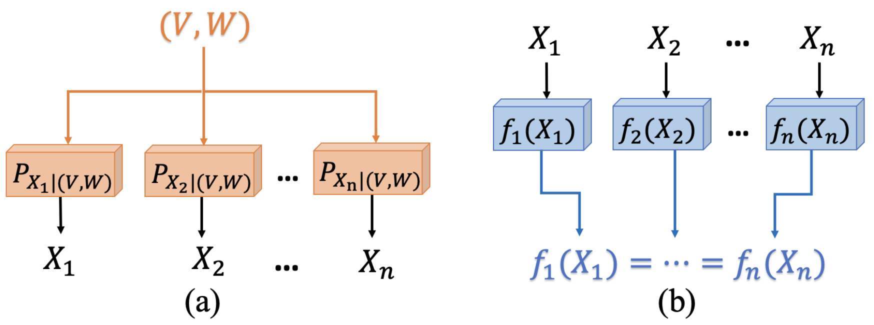


Figure 5.1: The one-shot exact version of (a) Wyner's distributed simulation problem and (b) Gács-Körner's distributed randomness extraction problem.

as $H(V)$. For a discrete random variable $H(V)$ is defined as $\sum_{v \in \text{Supp}(V)} -\mathbb{P}(V=v)\log_2 \mathbb{P}(V=v)$, where $\text{Supp}(V)$ is the support of V. Since the quantities we are interested in are independent of the choice of mean values, we assume without loss of generality that all variables have zero mean. For a pair of zero-mean random variables (X,Y), we use $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_X & \boldsymbol{\Sigma}_{XY}^\top \\ \boldsymbol{\Sigma}_{XY} & \boldsymbol{\Sigma}_Y \end{bmatrix}$ to denote their covariance matrix, where $\boldsymbol{\Sigma}_X = \mathbb{E}(XX^\top)$ and $\boldsymbol{\Sigma}_Y = \mathbb{E}(YY^\top)$ are the marginal covariance matrices, and $\boldsymbol{\Sigma}_{XY} = \mathbb{E}(XY^\top)$ is the cross-covariance matrix. We say that X, Y are *jointly singular* if

$$\text{rank}(\boldsymbol{\Sigma}) < \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y). \tag{5.7}$$

Our proofs show that we can assume without loss of generality that the marginal covariances are non-singular.

Common Information Dimension (CID). We consider the one-shot exact version of the distributed simulation problem in Fig. 5.1 (a), where n distributed nodes leverage the common randomness (V,W), in addition to their own local randomness, to simulate random vectors $X_1, \cdots, X_n$ that follow a given joint distribution $\pi_{X_1,\cdots,X_n}$. We note that the distributed simulation is possible only if $X_1,\cdots,X_n$ are conditionally independent given (V,W). Specifically, each node i generates X_i according to a distribution $P_{X_i|(V,W)}$, and the joint distribution is required to satisfy $\pi_{X_1,\cdots,X_n} = \mathbb{E}_{V,W}\left[\prod_{i=1}^n P_{X_i|(V,W)}\right]$.

We assume that V is a (one-dimensional) random variable with finite entropy $H(V) < \infty$, and $W \in \mathbb{R}^{d_W}$ is a possibly continuous random vector of dimension d_W that can be expressed as

$W = g(X_1, \ldots, X_n)$ for some function g in a given class of functions $\mathcal{F}$. Our goal is to determine the minimum dimension of W that is necessary to enable the distributed simulation. Note that we allow for the finite entropy random variable V to not follow the $\mathcal{F}$ restriction, and thus the common randomness needs to be expressed using a function in $\mathcal{F}$ only up to finite randomness. This allows the common information dimension to be zero when a finite amount of randomness is sufficient for the simulation, and avoids extra dimensions that may arise when this sufficient finite randomness cannot be expressed using $\mathcal{F}$.

Definition 5. The **Common Information Dimension (CID)** of random variables $X_1, \cdots, X_n$ with respect to a class of functions $\mathcal{F}$, is defined as

$$d_{\mathcal{F}}(X_1, \cdots, X_n) = \min\{d_W | W \in \mathcal{W}_{\mathcal{F}}\}, \tag{5.8}$$

where

$$\mathcal{W}_{\mathcal{F}} = \{W | \ \exists V, g : \mathbb{R}^{\sum_i^n d_{X_i}} \to \mathbb{R}^{d_W} \in \mathcal{F}, \text{ such that}$$
$$X_1 \perp\!\!\!\perp \ldots \perp\!\!\!\perp X_n | (V, W),\ H(V) < \infty,\ W = g(X_1, \cdots, X_n)\}.$$

We next define the concept of the **Rényi Common Information Dimension (RCID)** by replacing the dimension of W with the Renyi dimension described in (5.6).

Definition 6. The **Rényi Common Information Dimension (RCID)** of random variables $X_1, \cdots, X_n$ with respect to a class of functions $\mathcal{F}$ is defined as

$$d^R_{\mathcal{F}}(X_1, \cdots, X_n) = \inf\{d^R(W) | W \in \mathcal{W}_{\mathcal{F}}\}. \tag{5.9}$$

Finally, we define the **Gács-Körner's Common Information Dimension (GKCID)** (illustrated in Fig. 5.1 (b)) for continuous random variables by replacing the entropy with the Rényi dimension in the Gács-Körner's common information definition. This measures the maximum dimension of a vector W that can be extracted from each random variable individually, using a potentially different function $f_i \in \mathcal{F}$.

Definition 7. The **Gács-Körner's Common Information Dimension (GKCID)** of random variables $X_1, \cdots, X_n$ with respect to a class of functions $\mathcal{F}$ is defined as

$$d_{\mathcal{F}}^{GK}(X_1, \cdots, X_n) = \sup_{(\forall i \in [n]) W = f_i(X_i), f_i \in \mathcal{F}} d^R(W), \tag{5.10}$$

where the optimization is over $W, \{f_i\}_{i=1}^n$.

5.5 CID, RCID and GKCID for Jointly Gaussian Random Variables

We note that although CID, RCID and GKCID are well-defined, it is not clear whether and how they can be computed. In this section, we characterize the CID, RCID, and GKCID for an arbitrary number of jointly Gaussian random variables when $\mathcal{F}$ is the class of linear functions. Our results show that CID and RCID are equal and that CID, and RCID can be computed simply from ranks of covariance matrices, while GKCID is not larger than CID.

5.5.1 Main Results

We consider a jointly Gaussian random vector $X = [X_1, \cdots, X_n]$, where $X_i \in \mathbb{R}^{d_{X_i}}$, with covariance matrix Σ_X. We use $\Sigma_{I|J}$ for $I, J \subseteq \{1, \cdots, n\}$ to denote the conditional covariance matrix of X_I conditioned on X_J, where X_I denotes the elements of X with indices in the set I. We also use $-I$ to denote the complement of set I in $\{1, \cdots, n\}$. To simplify notation, we drop the parentheses when listing the elements of the sets I, J. Also, as we only consider $\mathcal{F}$ being the class of linear functions, we omit it in the subscripts.

5.5.1.1 CID of Jointly Gaussian Random Variables

Theorem 14 derives the CID for two jointly Gaussian random vectors X, Y in a closed form, with $\mathcal{F}$ the class of linear functions.

Theorem 14. *Let $[X, Y]$ be a jointly Gaussian random vector. Then, the common information* *dimension between X, Y with respect to the class of linear functions equals*

$$d(X,Y) = \mathit{rank}(\mathbf{\Sigma}_X) + \mathit{rank}(\mathbf{\Sigma}_Y) - \mathit{rank}(\mathbf{\Sigma}).$$

Moreover, when $\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y$ are non-singular[2]*, an W that satisfies the minimum in* (5.8) *is given by $W = \boldsymbol{N}_X X$, where $\boldsymbol{N} = \begin{bmatrix}\boldsymbol{N}_X & -\boldsymbol{N}_Y\end{bmatrix}$ is any basis of the row space of the matrix $\mathbf{\Sigma}$.*

Our result enables the simple calculation of CID with only the knowledge of covariance matrices. The proof also provides methods to construct the pair (V, W). This can be achieved by examining the null space of (linear transformations of) the covariance matrix $\mathbf{\Sigma}$. Theorem 15 extends the result to arbitrary number of Gaussian random vectors.

Theorem 15. *Let $X = [X_1, \cdots, X_n]$ be a jointly Gaussian random vector. The common information dimension between $X_1, \cdots, X_n$ with respect to the class of linear functions is*

$$d(X_1, \cdots, X_n) = \sum_{i=1}^{n} \mathit{rank}(\mathbf{\Sigma}_{-i}) - (n-1)\mathit{rank}(\mathbf{\Sigma}),$$

where $\mathbf{\Sigma}$ is the covariance matrix of X, and $\mathbf{\Sigma}_{-i}$ is the covariance matrix of the random vector $[X_1, \cdots, X_{i-1}, X_{i+1}, \cdots, X_n]$. Moreover, a W that satisfies the minimum in (5.8) *is given by Algorithm 9.*

5.5.1.2 RCID of Jointly Gaussian Random Variables

Our next result shows that for jointly Gaussian random variables RCID and CID are the same. The proof is provided at the end of Appendix D.2.

Lemma 5. Let $[X_1, \cdots, X_n]$ be a jointly Gaussian random vector. Then, the Rényi common information dimension between $X_1, \cdots, X_n$ with respect to the class of $\mathcal{F}$ of linear functions is given by

$$d^R(X_1, \cdots, X_n) = d(X_1, \cdots, X_n).$$

[2]*Note that this can always be achieved by a linear transformation on X, and a linear transformation of Y.*

Algorithm 9 Algorithm to find (V, W)

```
1: for $i = 1, \cdots, n$ do
2:     Find $\boldsymbol{A}_i$, a basis of the row space of $\boldsymbol{\Sigma}_{i|1:i-1}$.
3:     Define $U_i = \boldsymbol{A}_i X_i$ (remove parts from $X_i$ that can be obtained from previous $X_1, \cdots, X_{i-1}$).
4:     Find $\boldsymbol{B}_i$, a basis of the row space of $\boldsymbol{\Sigma}_{i+1:n|1:i-1}$.
5:     Define $Y_i = \boldsymbol{B}_i[X_{i+1}, \cdots, X_n]$ (remove parts from $[X_{i+1}, \cdots, X_n]$ that can be obtained from previous $X_1, \cdots, X_{i-1}$).
6:     Find $\tilde{\boldsymbol{N}}_i = [\boldsymbol{N}_i\ \bar{\boldsymbol{N}}_i]$, the null space of $\boldsymbol{\Sigma}_{U_i Y_i}$.
7:     Let $Z_i = \boldsymbol{N}_i U_i$ (the parts of $X_{i+1}, \cdots, X_n$ that can be obtained from $X_i$ but cannot be obtained from $X_1, \cdots, X_{i-1}$).
8: Let $W = [Z_1, \cdots, Z_n]$.
9: Find $\boldsymbol{C}_i$: basis for the covariance matrix of $X_i$ conditioned on $W$ (the parts of $X_i$ that cannot be obtained from $W$). Let $T_i = \boldsymbol{C}_i X_i$.
10: Use [LE17] to get $V$ that breaks the dependency of $T_1, \cdots, T_n$ conditioned on $W$.
```

5.5.1.3 GKCID of Jointly Gaussian Random Variables

Theorem 16 states the closed-form solution for GKCID of jointly Gaussian random variables. The proof of this theorem also gives an efficient method to construct W, given in (D.43) in Appendix D.3, with the maximum information dimension.

Theorem 16. *Let $X = [X_1, \cdots, X_n]$ be a jointly Gaussian random vector. The GKCID between $X_1, \cdots, X_n$ with respect to the class of linear functions is given by*

$$d^{GK}(X_1, \cdots, X_n) = r(\tilde{\boldsymbol{\Sigma}}) - \textit{rank}(\tilde{\boldsymbol{\Sigma}}), \tag{5.11}$$

where $r(\tilde{\boldsymbol{\Sigma}})$ is the number of rows of $\tilde{\boldsymbol{\Sigma}}$, with

$$\tilde{\boldsymbol{\Sigma}} = \begin{bmatrix} \boldsymbol{\Sigma}_{X_1'X_2'} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}_{X_2'X_3'} & \cdots & 0 \\ & \cdots & & \\ 0 & 0 & \cdots & \boldsymbol{\Sigma}_{X_{n-1}'X_n'} \\ \mathbf{0}_1\,\mathbf{1}_2 & \mathbf{1}_2\,\mathbf{0}_3 & \cdots & \mathbf{0}_{n-1}\,\mathbf{0}_n \\ \mathbf{0}_1\,\mathbf{0}_2 & \mathbf{0}_2\,\mathbf{1}_3 & \cdots & \mathbf{0}_{n-1}\,\mathbf{0}_n \\ \mathbf{0}_1\,\mathbf{0}_2 & \mathbf{0}_2\,\mathbf{0}_3 & \cdots & \mathbf{1}_{n-1}\,\mathbf{0}_n \end{bmatrix}, \tag{5.12}$$

$X_i' = \boldsymbol{F}_i X_i, \forall i \in [n]$, $\boldsymbol{F}_i$ is a basis of the row space of $\boldsymbol{\Sigma}_{X_i}$, $\mathbf{0}_i \in \mathbb{R}^{1\times d_{X_i'}}$ and $\mathbf{1}_i \in \mathbb{R}^{1\times d_{X_i'}}$ are all zeros (and ones respectively) row vectors with the same dimension as X_i'.

The following corollary follows from Theorems 15 and 16.

Corollary 4. *For two jointly Gaussian random variables X_1, X_2 we have that $d(X_1, X_2) = d^{GK}(X_1, X_2)$.*

This result does not extend to more than two variables: as the following example shows, GKCID can be strictly less than CID. We consider three random vectors X_1, X_2, X_3 with non-zero variance, $X_1 = X_2$ a.s., and $X_3, [X_1, X_2]$ independent. A W of dimension equal to d_{X_1} is required to break the dependency, hence, $d(X_1, X_2, X_3) = d_{X_1}$. However, as X_3 is independent of X_1, all functions with $f_1(X_1) = f_3(X_3)$ have zero entropy [GK73, Wit75], and zero information dimension. Corollary 5 follows from the proof of Theorem 16 and Lemma 10.

Corollary 5. *Let $X = [X_1, \cdots, X_n]$ be jointly Gaussian vectors, then $d^{GK}(X_1, \cdots, X_n) \leq d(X_1, \cdots, X_n)$.*

5.5.2 Proofs of Theorem 14 and 15

• Theorem 14:

Table 5.1: Table of notation for Theorem 14

Notation	**Definition**
X, Y	jointly Gaussian random vectors
$\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_X, \boldsymbol{\Sigma}_Y$	covariance matrices of $[X, Y], X, Y$ respectively
$\boldsymbol{N}$	basis of the null space of $\boldsymbol{\Sigma}$
$\boldsymbol{N}_X, \boldsymbol{N}_Y$	$\boldsymbol{N} = \begin{bmatrix} \boldsymbol{N}_X & -\boldsymbol{N}_Y \end{bmatrix}$ (5.13)
$\boldsymbol{N}'_X, \boldsymbol{N}'_Y$	basis of the complementary space of $\boldsymbol{N}_X, \boldsymbol{N}_Y$(5.14)
$\boldsymbol{M}_X, \boldsymbol{M}_Y$	$\boldsymbol{M}_X = \begin{bmatrix} \boldsymbol{N}_X \\ \boldsymbol{N}'_X \end{bmatrix}, \boldsymbol{M}_Y = \begin{bmatrix} \boldsymbol{N}_Y \\ \boldsymbol{N}'_Y \end{bmatrix}$ (5.15)

Proof. We start by stating the following lemma that enables to discover deterministic relations between X, Y by just examining the joint covariance matrix $\boldsymbol{\Sigma}$. The proofs of all lemmas are in Appendix D.1.

Lemma 6. Let $X = [X_1, X_2, ..., X_n]$ be a d_X-dimensional random vector, $d_X = \sum_{i=1}^{n} d_{X_i}$, with zero mean and covariance matrix $\boldsymbol{\Sigma}$. For any vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{d_X}$, we have that

$$\boldsymbol{a}^\top X = \boldsymbol{b}^\top X \text{ almost surely, if and only if } \boldsymbol{a}^\top \boldsymbol{\Sigma} = \boldsymbol{b}^\top \boldsymbol{\Sigma}.$$

Corollary 6. *There is a subset $I \subseteq \{1, ..., d_X\}$ such that $|I| = rank(\boldsymbol{\Sigma}_X)$, and $X_I \perp\!\!\!\perp Y|(V, W)$ if and only if $X \perp\!\!\!\perp Y|(V, W)$.*

Therefore, without loss of generality, we assume that X and Y are non-singular, which implies that $d_X = \text{rank}(\boldsymbol{\Sigma}_X)$, and $d_Y = \text{rank}(\boldsymbol{\Sigma}_Y)$.

Let $\boldsymbol{N} \in \mathbb{R}^{r(\boldsymbol{N}) \times (d_X + d_Y)}$ be a basis of the left null space of $\boldsymbol{\Sigma}$. We next show some properties for the matrix $\boldsymbol{N}$ that will help us to prove the theorem. By definition of $\boldsymbol{N}$, we have the following facts

Fact 1. $\boldsymbol{N}\boldsymbol{\Sigma} = \boldsymbol{0}$. (5.16)

Fact 2. $\text{rank}(\boldsymbol{N}) = d_X + d_Y - \text{rank}(\boldsymbol{\Sigma}) = \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y) - \text{rank}(\boldsymbol{\Sigma})$. (5.17)

Using Lemma 6 and (5.16), we have that $\boldsymbol{N}\begin{bmatrix} X^\top & Y^\top \end{bmatrix}^\top = \boldsymbol{0}$ almost surely. We partition $\boldsymbol{N}$ as $\begin{bmatrix} \boldsymbol{N}_X & -\boldsymbol{N}_Y \end{bmatrix}$, where $\boldsymbol{N}_X \in \mathbb{R}^{r(\boldsymbol{N})\times d_X}$ and $\boldsymbol{N}_Y \in \mathbb{R}^{r(\boldsymbol{N})\times d_Y}$. Then we have that

$$\boldsymbol{N}_X X = \boldsymbol{N}_Y Y. \tag{5.18}$$

$\boldsymbol{N}$ is full-row-rank by definition; in the following, we show that $\boldsymbol{N}_X$ (and similarly $\boldsymbol{N}_Y$) are also full-row-rank.

Lemma 7. Let $[X, Y]$ be a random vector with covariance matrix $\boldsymbol{\Sigma}$ and $\boldsymbol{N} = \begin{bmatrix} \boldsymbol{N}_X & -\boldsymbol{N}_Y \end{bmatrix}$ be a basis for the null space of $\boldsymbol{\Sigma}$, where $\boldsymbol{N} \in \mathbb{R}^{r(\boldsymbol{N})\times(d_X+d_Y)}$, $\boldsymbol{N}_X \in \mathbb{R}^{r(\boldsymbol{N})\times d_X}$, and $\boldsymbol{N}_Y \in \mathbb{R}^{r(\boldsymbol{N})\times d_Y}$. If X and Y are non-singular (i.e., $\boldsymbol{\Sigma}_X, \boldsymbol{\Sigma}_Y$ are full-rank), we have that

$$\text{rank}(\boldsymbol{N}_X) = \text{rank}(\boldsymbol{N}_Y) = \text{rank}(\boldsymbol{N}). \tag{5.19}$$

Next, we define two square non-singular matrices $\boldsymbol{M}_X$ and $\boldsymbol{M}_Y$ as $\boldsymbol{M}_X = \begin{bmatrix} \boldsymbol{N}_X^\top & \boldsymbol{N}_X'^\top \end{bmatrix}^\top \in \mathbb{R}^{d_X\times d_X}$ and $\boldsymbol{M}_Y = \begin{bmatrix} \boldsymbol{N}_Y^\top & \boldsymbol{N}_Y'^\top \end{bmatrix}^\top \in \mathbb{R}^{d_Y\times d_Y}$, where $\boldsymbol{N}_X', \boldsymbol{N}_Y'$ are a basis for the complementary space of $\boldsymbol{N}_X$ and $\boldsymbol{N}_Y$, respectively. Lemma 8, and Lemma 9 show two properties of the quantities $\boldsymbol{M}_X, \boldsymbol{M}_Y, \boldsymbol{N}_X', \boldsymbol{N}_Y'$ which we prove in the Appendix D.1.

Lemma 8. Let $\boldsymbol{M}_X \in \mathbb{R}^{d_X\times d_X}, \boldsymbol{M}_Y \in \mathbb{R}^{d_Y\times d_Y}$ be full-rank matrices, and X, Y, V, W be random vectors of dimension d_X, d_Y, d_V, and d_W respectively. We have that

$$\boldsymbol{M}_X X \perp\!\!\!\perp \boldsymbol{M}_Y Y | (V, W) \text{ if and only if } X \perp\!\!\!\perp Y | (V, W).$$

Lemma 9. Let $\boldsymbol{N}_X, \boldsymbol{N}_Y, \boldsymbol{N}_X', \boldsymbol{N}_Y'$ be as defined in (5.13) and (5.14). Conditioned on $\boldsymbol{N}_X X$, we have that $\begin{bmatrix} (\boldsymbol{N}_X' X)^\top & (\boldsymbol{N}_Y' Y)^\top \end{bmatrix}^\top$ has full-rank covariance matrix.

We are now ready to prove Theorem 14. We first show that the common information dimension is upper bounded as $d(X,Y) \leq \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y) - \text{rank}(\boldsymbol{\Sigma})$. Consider $\boldsymbol{N}_X X$ as a possible W. Conditioned on $\boldsymbol{N}_X X$, both $\boldsymbol{N}_X X$ and $\boldsymbol{N}_Y Y$ are deterministic, since $\boldsymbol{N}_X X = \boldsymbol{N}_Y Y$ from (5.18). Hence, conditioned on $\boldsymbol{N}_X X$, breaking the dependency between $\boldsymbol{M}_X X$ and $\boldsymbol{M}_Y Y$ reduces to breaking the dependency between $\boldsymbol{N}'_X X$ and $\boldsymbol{N}'_Y Y$ conditioned on $\boldsymbol{N}_X X$.

From Lemma 9, we have that conditioned on $\boldsymbol{N}_X X$, $\begin{bmatrix}(\boldsymbol{N}'_X X)^\top & (\boldsymbol{N}'_Y Y)^\top\end{bmatrix}^\top$ is jointly Gaussian with full-rank covariance matrix. Hence, by the result in [LE17], there is a V_W with $H(V_W) < \infty$ such that $\boldsymbol{N}'_X X \perp\!\!\!\perp \boldsymbol{N}'_Y Y | (W, V_W)$, where $W = \boldsymbol{N}_X X$. Since the covariance matrix of $\begin{bmatrix}(\boldsymbol{N}'_X X)^\top & (\boldsymbol{N}'_Y Y)^\top\end{bmatrix}^\top$ conditioned on $\boldsymbol{N}_X X$ does not depend on the value of $\boldsymbol{N}_X X$ and is only a function of the covariance matrix of $\boldsymbol{N}_X X$, V_W can be the same for all W, and we can refer to V_W as V. This shows that $\boldsymbol{M}_X X \perp\!\!\!\perp \boldsymbol{M}_Y Y | (\boldsymbol{N}_X X, V)$. By Lemma 8, $X \perp\!\!\!\perp Y | (\boldsymbol{N}_X X, V)$. Thus,

$$d(X,Y) \leq d_{\boldsymbol{N}_X X} \overset{(i)}{=} \text{rank}(\boldsymbol{N}_X) \overset{(5.19)}{=} \text{rank}(\boldsymbol{N}) \overset{(5.16)}{=} \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y) - \text{rank}(\boldsymbol{\Sigma}), \tag{5.20}$$

where (i) follows since $\boldsymbol{N}_X$ is full rank by Lemma 7.

Next, we prove in Lemma 10 the other direction, that $d(X,Y) \geq \text{rank}(\boldsymbol{N})$. At the heart of the lemma, we prove that if there is a common function that can be extracted from both X, Y, namely, $f_X(X) = f_Y(Y)$ a.s. for some f_X, f_Y, then for (V, W) to break the X, Y dependency, $f_X(X)$ (and hence $f_Y(Y)$) is a deterministic function of (V, W). We also show that if f_X, f_Y are linear and $W = \boldsymbol{A}\begin{bmatrix}X^\top & Y^\top\end{bmatrix}^\top$ for some $\boldsymbol{A}$, then $d_W \geq d_{f_X(X)}$.

Lemma 10. Let $[X, Y]$ be a jointly Gaussian random vector and V, W be random variables such that $W = \boldsymbol{A}\begin{bmatrix}X^\top & Y^\top\end{bmatrix}^\top$ for some matrix $\boldsymbol{A}$, $H(V) \leq \infty$ and $X \perp\!\!\!\perp Y | (V, W)$. Let matrix $\boldsymbol{N}_X$ be such that $\boldsymbol{N}_X X$ is non-singular. If there exists matrix $\boldsymbol{N}_Y$ such that $\boldsymbol{N}_X X = \boldsymbol{N}_Y Y$ a.s., then $\boldsymbol{N}_X X = \boldsymbol{A}' W$ a.s. for some matrix $\boldsymbol{A}'$ and $d_W \geq d_{\boldsymbol{N}_X X}$.

Proof. First, we show that $\boldsymbol{N}_X X$ is a deterministic function of (V, W). Suppose towards a contradiction that there is a set $\mathcal{S} \subseteq \mathbb{R}^{r(\boldsymbol{N})}$ such that $0 < \mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)] < 1$. Since $\boldsymbol{N}_X X = \boldsymbol{N}_Y Y$

a.s., we have that

$$\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S}, \boldsymbol{N}_Y Y \in \mathcal{S}^C | (V, W)] = 0, \tag{5.21}$$

where $\mathcal{S}^C$ is the complement of $\mathcal{S}$. However, from $0 < \mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)] < 1$ we get that

$$\begin{aligned} &\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)]\mathbb{P}[\boldsymbol{N}_Y Y \in \mathcal{S}^C | (V, W)] \\ &= \mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)]\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S}^C | (V, W)] \neq 0. \end{aligned} \tag{5.22}$$

This implies that

$$\begin{aligned} &\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S}, \boldsymbol{N}_Y Y \in \mathcal{S}^C | (V, W)] \\ &\neq \mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)]\mathbb{P}[\boldsymbol{N}_Y Y \in \mathcal{S}^C | (V, W)]. \end{aligned} \tag{5.23}$$

However, as functions of independent random variables are independent, we have that $\boldsymbol{N}_X X$ and $\boldsymbol{N}_Y Y$ are conditionally independent given (V, W); being projections of $\boldsymbol{M}_X X, \boldsymbol{M}_Y Y$. This contradicts (5.23).

This implies that for any $\mathcal{S} \subseteq \mathbb{R}^{r(\boldsymbol{N})}$ we either have $\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)] = 1$ or $\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)] = 0$. We show next that this implies that $\boldsymbol{N}_X X$ is a deterministic function of (V, W).

As the interval from $(-\infty, \infty)$ can be partitioned into countably many sets of the form $(0 + m, 1 + m]$, by countable additivity of measures we get that there is a cube of the form $\mathcal{S} = \prod_{i=1}^{r(\boldsymbol{N})}(0 + m_i, 1 + m_i]$ that has $\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{S} | (V, W)] = 1$. If we repeatedly halve one of the largest dimensions of the cube we get a sequence of hyper-rectangles $... \subseteq \mathcal{R}_2 \subseteq \mathcal{R}_1$ such that $\mathbb{P}[\boldsymbol{N}_X X \in \mathcal{R}_i | (V, W)] = 1, \forall i = 1, 2, ...$ and $\cap_{i \in \mathbb{N}} \mathcal{R}_i$ contains exactly one member. The last fact is proved in the following. We notice that $\cap_{i \in \mathbb{N}} \mathcal{R}_i$ contains at most one member because for any two points $x_1, x_2 \in \mathbb{R}^{r(\boldsymbol{N})}$, there is some i such that the largest dimension of $\mathcal{R}_i$ is less than $\|x_1 - x_2\|_2$ which implies that at most one point of x_1, x_2 can be in R_i. It is also not possible that $\cap_{i \in \mathbb{N}} \mathcal{R}_i$ is empty as by the continuity from above of finite measures, we have that

$$\mathbb{P}[\boldsymbol{N}_X X \in \cap_{i \in \mathbb{N}} \mathcal{R}_i | (V, W)] = 1. \tag{5.24}$$

Therefore, $\cap_{i \in \mathbb{N}} \mathcal{R}_i$ must contain a single member. Let us denote the unique point in $\cap_{i \in \mathbb{N}} \mathcal{R}_i$ by $g(V, W)$, where g is a deterministic function. Then, we have that $\mathbb{P}[\boldsymbol{N}_X X = g(V, W) | (V, W)] = 1$.

Hence, we have that

$$\begin{aligned} H(\boldsymbol{N}_X X|W) &\leq H(\boldsymbol{N}_X X, V|W) \\ &= H(\boldsymbol{N}_X X|(V,W)) + H(V|W) \\ &= H(V|W) \leq H(V) < \infty. \end{aligned} \tag{5.25}$$

Since $W = \boldsymbol{A}\begin{bmatrix} X^\top & Y^\top \end{bmatrix}^\top$, we have that $(\boldsymbol{N}_X X, W)$ follows a jointly Gaussian distribution. As a result, conditioned on W, we have that $\boldsymbol{N}_X X$ is also jointly Gaussian, whose entropy is either 0 (for zero variance) or ∞. Based on (5.25), it must be that $H(\boldsymbol{N}_X X|W) = 0$. Hence, we have that $\boldsymbol{N}_X X = \boldsymbol{B}W$, for some $\boldsymbol{B} \in \mathbb{R}^{\text{rank}(\boldsymbol{N}) \times d_W}$. And as a result, $\boldsymbol{N}_X \boldsymbol{\Sigma}_X = \boldsymbol{B}\mathbb{E}(WX^\top)$. Then we have that

$$\text{rank}(\boldsymbol{N}_X \boldsymbol{\Sigma}_X) \overset{(i)}{=} \text{rank}(\boldsymbol{N}_X) \overset{(5.19)}{=} \text{rank}(\boldsymbol{N}) \leq \text{rank}(\boldsymbol{B}) \leq d_W, \tag{5.26}$$

where (i) follows from the fact that $\boldsymbol{N}_X X$ has full rank covariance matrix. □

Combining (5.20) and (5.26), we conclude that

$$d(X,Y) = \text{rank}(\boldsymbol{N}) = \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y) - \text{rank}(\boldsymbol{\Sigma}).$$

□

• **Proof Outline of Theorem 15:**

Proof. We here give a proof outline, and provide the complete proof in App. D.2. The main part of the proof, illustrated in Algorithm 9, constructs variables $Z = [Z_1, \cdots, Z_n]$ that satisfy:
(i) Conditioned on Z, the dependency between $X_1, \cdots, X_n$ can be broken using finite randomness (see Lemma 20 in App. D.2). This is proved by showing that after eliminating from X the parts that can be almost surely determined by Z, the remaining part is jointly non-singular Gaussian. This shows that CID is upper bounded by the total number of dimensions of Z.
(ii) For any V, W that break the dependency between $X_1, \cdots, X_n$, we have that Z is a linear function of W (see Lemma 21 in App. D.2). By showing that Z is a jointly non-singular Gaussian

vector, we prove that the dimension of W is lower bounded by the dimension of Z, hence, CID is lower bounded by the number of dimensions of Z.

We build Z sequentially as follows. Z_1 represents the information that X_1 contains about $[X_2, \cdots, X_n]$; namely, the linearly independent dimensions of $X_2, \cdots, X_n$ that can be determined from X_1. Then, Z_2 contains the amount of information that X_2 contains about $X_3, \cdots, X_n$ that X_1 does not contain. Generally, Z_i contains the information that X_i contains about $X_{i+1}, \cdots, X_n$ which is not contained in any of the previous $X_1, \cdots, X_{i-1}$. $\square$

5.6 The relation between CID and Wyner's Common Information

In this section, we restrict our attention to the case of two Gaussian random vectors, and formulate three approximation problems, to investigate the asymptotic behavior of Wyner's common information in the (nearly) infinity regime. Our results indicate that the growth rates of the approximate common information in all these scenarios are determined by the common information dimension.

Remark 2. We make two observations related to our formulations:
1) We restrict our attention to two Gaussian random vectors because, as mentioned before, the calculation of Wyner's common information is challenging, and the closed-form solution for continuous random vectors (of arbitrary dimension) is only known for two Gaussian sources, while a closed-form expression for multiple Gaussian sources remains open.
2) We do not consider the GK-version of the approximation problem since the GK common information has an inherent discontinuity. In particular, it is easy to see that if the Gaussian sources are singular, then the GK common information is infinite (as is the case for the Wyner as well), however, if they are approximated by any non-singular Gaussian distribution, then the GK common information of the approximate distribution is zero. Hence, the GK version of the common information is not suitable for such approximations. Thus in this section, we exclusively focus on the Wyner version. For simplicity, we drop the subscripts in $C_{\text{Wyner}}(X, Y)$ and use $C(X, Y)$ instead to denote the Wyner's common information.

5.6.1 Problem Statements

5.6.1.1 Common information of nearly singular sources

This formulation aims to study the growth rate of the common information for a sequence of pairs of random variables that approach joint singularity. In particular, let X, Y be Gaussian random variables with rank($\mathbf{\Sigma}$) $<$ rank($\mathbf{\Sigma}_X$) + rank($\mathbf{\Sigma}_Y$), and hence, $d(X, Y) \geq 1$ and $C(X, Y) = \infty$. Let $\{(X_\epsilon, Y_\epsilon)\}_{\epsilon > 0}$ be a sequence of Gaussian random variables satisfying

$$\mathbf{\Sigma}_{X_\epsilon} = \mathbf{\Sigma}_X, \mathbf{\Sigma}_{Y_\epsilon} = \mathbf{\Sigma}_Y, \text{ and } \forall\, i, \;\; |\rho_i(\epsilon) - \sigma_i| = \epsilon, \tag{5.27}$$

where $\{\sigma_i\}$ and $\{\rho_i(\epsilon)\}$ are the singular values of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$ and $\mathbf{\Sigma}_{X_\epsilon}^{-1/2}\mathbf{\Sigma}_{X_\epsilon Y_\epsilon}\mathbf{\Sigma}_{Y_\epsilon}^{-1/2}$ respectively, in a decreasing order. These requirements ensure that (X_ϵ, Y_ϵ) remain non-singular (and thus have finite common information), while the joint distribution of (X_ϵ, Y_ϵ) converges to that of X, Y as $\epsilon \downarrow 0$.

Remark 3. The conditions in (5.27) force each singular value of $\mathbf{\Sigma}_{X_\epsilon}^{-1/2}\mathbf{\Sigma}_{X_\epsilon Y_\epsilon}\mathbf{\Sigma}_{Y_\epsilon}^{-1/2}$ to go to the corresponding singular value of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$ at an identical rate ϵ. This enables us to study how the common information increases as a function of ϵ. There exist however other cases where the same results apply, and we next give two examples.

One case is when we have different convergence rates for each singular value. The second condition in (5.27) can be replaced with $\forall i, \;\; |\rho_i(\epsilon) - \sigma_i| = a_i\epsilon$, where a_i's are constants with different values. provided that these rates are of the same order, meaning that they differ only by a multiplicative constant.

A second case is when we only approximate the singular values that equal to 1. Recall that from (5.5), the common information is infinite when $\sigma_i = 1$ for some i. Consider a sequence of covariance matrices that have singular values satisfying (5.27) when $\sigma_i = 1$ and share the same singular values with the target distribution for all other indices, i.e., $\rho_i = \sigma_i$ when $\sigma_i \neq 1$. It is easy to see that the same results we establish assuming the condition in (5.27) holds, also extend for the described sequence as well.

5.6.1.2 Approximate simulation

This formulation looks at approximating a pair of Gaussian random variables X, Y that are jointly singular ($C(X,Y) = \infty$) with Gaussian random variables $\hat{X}, \hat{Y}$ that (i) are non-singular ($C(\hat{X}, \hat{Y})$ is finite) and (ii) have a distribution close to the distribution of X, Y. In other words, we ask, if we are restricted to using a finite number of bits as common information, how well can we (approximately) simulate X, Y.

We use the Frobenius-norm between covariance matrices to measure how close two Gaussian distributions are. For some $\epsilon > 0$, we define the *ϵ-approximation common information* as

$$C_\epsilon(X,Y) := \min_{\|\mathbf{\Sigma}-\hat{\mathbf{\Sigma}}\|_F \leq \epsilon} C(\hat{X}, \hat{Y}), \tag{5.28}$$

where the optimization is over all pairs $(\hat{X}, \hat{Y})$ with covariance matrix $\hat{\mathbf{\Sigma}}$ and $\|\cdot\|_F$ is the Frobenius norm of a matrix.

Remark 4. The results on $C_\epsilon(X,Y)$ extend if we replace the Frobenius norm with any distribution distance dist$(XY, \hat{X}\hat{Y})$ that satisfies $a\|\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}\|_F \leq$ dist$(XY, \hat{X}\hat{Y}) \leq b\|\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}\|_F$ for all Gaussian variables $XY, \hat{X}\hat{Y}$ and some constants a and b.

Remark 5. Note that formulation 1 in (5.27) studies a more restricted set of sequences than the sequences included in the feasible set of the optimization problem in (5.28). However, the result we show for formulation 1 is stronger as it holds for *all sequences* that satisfy the condition in (5.27). In contrast, the results in formulation 2 only hold for the sequence with the minimum common information (that achieves the optimal value of the minimization problem). It can be easily shown that there exist sequences in the feasible set of formulation 2 that have different asymptotics. For example, if some singular values of the approximation matrix take the value 1 or approach 1 at a rate different from $\Theta(\epsilon)$(e.g., ϵ^2 or 2^ϵ).

5.6.1.3 Common information between quantized variables

In this formulation, we study Wyner's common information between quantized continuous random vectors. Let X and Y be a pair of jointly singular Gaussian random vectors with $C(X, Y) = \infty$. For $m > 0$, we use $\langle X \rangle_m = [\langle X_1 \rangle_m, \cdots, \langle X_{d_X} \rangle_m]^\top$ and $\langle Y \rangle_m = [\langle Y_1 \rangle_m, \cdots, \langle Y_{d_Y} \rangle_m]^\top$ to denote the quantized X and Y, where the quantization operator on each element is defined as $\langle X_i \rangle_m = \frac{\lfloor m X_i \rfloor}{m}$ and $\langle Y_i \rangle_m = \frac{\lfloor m Y_i \rfloor}{m}$. Note that $C(\langle X \rangle_m, \langle Y \rangle_m) < \infty$ for all $0 < m < \infty$, and as m approaches ∞, $\langle X \rangle_m$ and $\langle Y \rangle_m$ converges to X and Y respectively. We are interested in the growth rate of their common information $C(\langle X \rangle_m, \langle Y \rangle_m)$.

5.6.2 Asymptotic Behavior of Wyner's Common Information

In this section, we present our main results and proof outlines for the three formulations described in Section 5.6.1. The detailed proofs are provided in Appendices D.5 and D.6.

Before stating our main results, we present two properties of covariance matrices and the common information dimension, which are important to the proofs of Theorems 17 and 18. As stated in (5.5) the common information is determined by the singular values of the normalized cross-covariance matrix $\mathbf{\Sigma}_X^{-1/2} \mathbf{\Sigma}_{XY} \mathbf{\Sigma}_Y^{-1/2}$. Lemma 11 proves a bound on these singular values.

Lemma 11. Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be jointly Gaussian variables with covariance matrix $\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_X & \mathbf{\Sigma}_{XY}^\top \\ \mathbf{\Sigma}_{XY} & \mathbf{\Sigma}_Y \end{bmatrix}$, and $d = \min\{d_X, d_Y\}$. Then the singular values of $\mathbf{\Sigma}_X^{-1/2} \mathbf{\Sigma}_{XY} \mathbf{\Sigma}_Y^{-1/2}$, denoted as $\{\sigma_i\}_{i=1}^d$, satisfy

$$0 \le \sigma_i \le 1, \forall i \in [d]. \tag{5.29}$$

The following lemma shows the relationship between the common information dimension and the singular values of $\mathbf{\Sigma}_X^{-1/2} \mathbf{\Sigma}_{XY} \mathbf{\Sigma}_Y^{-1/2}$, which will enable us to connect the quantities $C(X_\epsilon, Y_\epsilon)$ and $C_\epsilon(X, Y)$ with the common information dimension $d(X, Y)$.

Lemma 12. Assume $X \in \mathbb{R}^{d_X}$, $Y \in \mathbb{R}^{d_Y}$ are jointly Gaussian variables with covariance matrix

$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_X & \boldsymbol{\Sigma}_{XY}^\top \\ \boldsymbol{\Sigma}_{XY} & \boldsymbol{\Sigma}_Y \end{bmatrix}$, and $\{\sigma_i\}$ are the singular values of $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$. Then the common information dimension between X and Y, with respect to linear functions, satisfies

$$d(X, Y) = \sum_{i=1}^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}. \tag{5.30}$$

5.6.2.1 Common information of nearly singular sources

We consider a sequence of pairs of Gaussian random variables $\{(X_\epsilon, Y_\epsilon)\}_{\epsilon>0}$ satisfying (5.27). The following result shows that the growth rate of the common information $C(X_\epsilon, Y_\epsilon)$ is determined by the common information dimension $d(X, Y)$ with respect to linear functions.

Theorem 17. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly singular Gaussian variables, and $\{(X_\epsilon, Y_\epsilon)\}_{\epsilon>0}$ be a sequence as defined in* (5.27). *Then the common information $C(X_\epsilon, Y_\epsilon)$ satisfies*

$$\lim_{\epsilon \downarrow 0} \frac{C(X_\epsilon, Y_\epsilon)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = d(X, Y). \tag{5.31}$$

Proof Outline of Theorem 17:

Proof. The main technical challenge in proving Theorem 17 is the fact that there exist multiple sequences of random variables X_ϵ, Y_ϵ, with different values of $C(X_\epsilon, Y_\epsilon)$, that satisfy the constraints in (5.27). To address this issue, we prove the result by deriving an upper and a lower bound on $C(X_\epsilon, Y_\epsilon)$ that have the same asymptotic behavior.

The proof focuses on showing that $\lim_{\epsilon\downarrow 0} \frac{C(X_\epsilon, Y_\epsilon)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = \sum_i^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}$, where $\{\sigma_i\}$ are the singular values of the matrix $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$. We prove this by providing an upper and lower bound on $\frac{C(X_\epsilon, Y_\epsilon)}{\frac{1}{2}\log(\frac{1}{\epsilon})}$ that have the same limit when $\epsilon \downarrow 0$. Then we relate $\sum_i^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}$ to the common information dimension $d(X, Y)$ using Lemma 12. $\square$

5.6.2.2 Approximate simulation

The following result shows that the ϵ-approximation common information $C_\epsilon(X,Y)$, defined in (5.28), for Gaussian variables grows at a rate determined by the common information dimension $d(X,Y)$ with respect to linear functions.

Theorem 18. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly Gaussian random variables, then*

$$\lim_{\epsilon \downarrow 0} \frac{C_\epsilon(X,Y)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = d(X,Y). \tag{5.32}$$

Proof Outline of Theorem 18:

Proof. The main technical challenge in proving Theorem 18 is the difficulty in finding a closed form solution of the optimization problem defining $C_\epsilon(X,Y)$. To address this issue, we follow a similar approach as in Theorem 17 by deriving an upper and a lower bound on C_ϵ that have the same asymptotics. However, it turns out that finding upper and lower bounds that have the same asymptotics is more involved than in the case of Theorem 17.

The proof uses a pair of upper and lower bounds, derived as described next, to show that $\lim_{\epsilon\downarrow 0} \frac{C(X_\epsilon,Y_\epsilon)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = \sum_i^{\min\{d_X,d_Y\}} \mathbb{1}\{\sigma_i = 1\}$, where $\{\sigma_i\}$ are the singular values of the matrix $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$. From Lemma 12, this concludes the proof of Theorem 18.

Upper Bound. As $C_\epsilon(X,Y)$ is the optimal value of a minimization problem, any feasible solution provides an upper bound. To find a feasible solution we use $\mathbf{\Sigma}_{\hat{X}} = \mathbf{\Sigma}_X, \mathbf{\Sigma}_{\hat{Y}} = \mathbf{\Sigma}_Y$. Then, we design the singular values of $\mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}$, denoted as $\{\rho_i\}$, as follows. We set $\rho_i = \sigma_i$ when $\sigma_i \neq 1$. Recall that choosing a singular value to be 1 results in an infinite value for the common information. Hence, when $\sigma_i = 1$ we choose $\rho_i = 1-\delta$ where δ is the largest value that does not violate the constraint $||\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}||_F \leq \epsilon$.

Lower Bound. To find a lower bound, we relax the constraints set $||\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}||_F \leq \epsilon$, resulting in a smaller optimal value, to make it possible to find a closed form solution of the problem. The proof of the lower bound hinges on showing that $||\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}||_F \leq \epsilon$ implies

$$||\Lambda - \hat{\Lambda}||_F \leq c\epsilon, \tag{5.33}$$

where $\Lambda = \text{diag}(\sigma_i), \hat{\Lambda} = \text{diag}(\rho_i)$ are matrices containing the singular values of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$, $\mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}$ respectively, and c is a constant that may depend on $\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y$. To further simplify the problem, we remove from the objective function the terms corresponding to $\sigma_i < 1$, and also remove the value $\log(1+\rho_i)$ from each term (recall the common information in (5.5)). We note that each term in the objective function is non-negative, and hence, removing terms will not increase the optimal solution value. Furthermore, we expect the asymptotics of the common information to be influenced by the singular values corresponding to $\sigma_i = 1$. This results in the following optimization problem

$$\begin{aligned} \min_{\rho} \quad & \frac{1}{2}\sum_{i:\sigma_i=1} \log\frac{1}{1-\rho_i} \\ \text{s.t.} \quad & \sum_{i:\sigma_i=1} (\sigma_i - \rho_i)^2 \leq \epsilon^2, 0 \leq \rho_i \leq 1, \end{aligned}$$

which can be solved in a closed form using symmetry and concavity of the $\log$ function. $\square$

Remark 6. We note that we can efficiently construct random variables for each ϵ with common information that has the asymptotic behavior in Theorem 2 (and thus can be used to approximate the target singular distribution with (nearly) the smallest common information). A possible choice is $\mathbf{\Sigma}_{\hat{X}} = \mathbf{\Sigma}_X, \mathbf{\Sigma}_{\hat{Y}} = \mathbf{\Sigma}_Y, \mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2} = U\hat{\Lambda}V$, where U, V are orthonormal matrices of the singular value decomposition of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$, and $\hat{\Lambda}$ is given in (D.58) (Appendix D.6).

Remark 7. Why do these two theorems have the same bound? It may seem at first surprising that even though $C(X_\epsilon, Y_\epsilon)$ and $C_\epsilon(X,Y)$ have different definitions, they both grow (nearly) as $\frac{1}{2}d(X,Y)\log(1/\epsilon)$. Indeed, as we observed in Remark 5 the feasible set defining $C_\epsilon(X,Y)$ in (5.28) contains different sequences of random variables than those satisfying the conditions in (5.27). However, the proof of Theorem 18 shows that the random variables which minimize the common information satisfy a constraint similar to (5.27); namely, the singular values ρ_i corresponding to $\sigma_i = 1$ have the same distance to 1, where σ_i and ρ_i are the singular value of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$ and $\mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}$ respectively. Intuitively, to minimize the common information in (5.5), we need ρ_i to be as far as possible from the value 1, however, the distance constraint in (5.28) restricts us from choosing ρ_i too far from 1 whenever $\sigma_i = 1$. If one ρ_i is very close to 1, it will dominate the summation in (5.5) resulting in large common information. Hence, a good solution to (5.28) distributes the distance budget ϵ evenly across the ρ_i's corresponding to $\sigma_i = 1$.

Remark 8. Consider a machine that stores each real value in B bits using standard floating-point representation. Such a type of machine can store numbers up to a precision[3] of 2^{-B}. Suppose that we are interested in the distributed simulation of two random variables X, Y up to the maximum machine precision. Our result indicates that the minimum amount of shared randomness between the quantized X, Y is proportional to $\frac{1}{2}\log(\frac{1}{2^{-B}})d(X,Y) = \frac{1}{2}Bd(X,Y)$ bits. As each variable stores at most B bits, $\frac{1}{2}d(X,Y)$ variables (dimensions) are required to store the shared randomness on such machines in order to perform the distributed simulation up to the required 2^{-B} accuracy.

5.6.2.3 Common information between quantized variables

The following result shows that the Wyner's common information between uniformly quantized Gaussian random vectors also grows in proportion to the common information dimension, as the quantization precision increases.

Theorem 19. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly singular Gaussian random vectors. Then the common information between the quantized $\langle X\rangle_m$ and $\langle Y\rangle_m$ satisfies*

$$\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} = d(X,Y), \tag{5.34}$$

where $d(X,Y)$ is the common information dimension of X and Y with respect to the class of linear functions.

Proof Outline of Theorem 19:

Proof. Since it is hard in general to directly solve Wyner's common information even for discrete variables [YT22], we prove the result through matching upper and lower bounds. First, we show

[3]Note that for any quantization scheme, there exists at least one input value such that the quantization error is greater or equal to 2^{-B}.

that $\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \geq d(X,Y)$ using the inequality that mutual information is not larger than the Wyner's common information [Wyn75].

Next, we prove $\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \leq d(X,Y)$ as follows. Recall (from the proofs of Lemma 11 and 12) that there exist invertible linear transformations $\boldsymbol{T}_X$ and $\boldsymbol{T}_Y$ such that $X = \boldsymbol{T}_X X'$ and $Y = \boldsymbol{T}_Y Y'$, where $X' \in \mathbb{R}^{d_X}$ and $Y' \in \mathbb{R}^{d_Y}$ are another pair of Gaussian random vectors with independent elements (i.e. their covariance matrices $\boldsymbol{\Sigma}_{X'}$, $\boldsymbol{\Sigma}_{Y'}$, and $\boldsymbol{\Sigma}_{X'Y'}$ are all diagonal). We first show that $\lim_{m'\to\infty} \frac{C(\langle X'\rangle_{m'}, \langle Y'\rangle_{m'})}{\log m'} \leq d(X',Y')$. Note that $d(X',Y') = d(X,Y)$ since $\boldsymbol{T}_X, \boldsymbol{T}_Y$ are invertible transformations.

However, $C(\langle X\rangle_m, \langle Y\rangle_m), C(\langle X'\rangle_m, \langle Y'\rangle_m)$ may not be equal in general (note that $\langle \boldsymbol{T}_X \langle X'\rangle_m\rangle_m \neq \langle \boldsymbol{T}_X X'\rangle_m = \langle X\rangle_m$) (similarly for Y' and Y). Our proof proceeds by showing that $\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \leq \lim_{m\to\infty} \frac{C(\langle X'\rangle_m, \langle Y'\rangle_m)}{\log m'}$. This is proved using the following ideas: (i) $\lim_{m\to\infty} \frac{C(\langle X'\rangle_m, \langle Y'\rangle_m)}{\log m} = \lim_{m\to\infty} \frac{C(\langle X'\rangle_{\alpha m}, \langle Y'\rangle_{\alpha m})}{\log m}$ for any fixed $\alpha \in [0,\infty]$ (ii) $\langle \boldsymbol{T}_X \langle X'\rangle_{\alpha m}\rangle_m = \langle X\rangle_m$ and $\langle \boldsymbol{T}_Y \langle Y'\rangle_{\alpha m}\rangle_m = \langle Y\rangle_m$ can be achieved with high probability by choosing a large α. $\square$

5.7 Numerical Evaluation

In this section, we numerically verify the asymptotic behaviors of $C_\epsilon(X,Y)$, $C(X_\epsilon, Y_\epsilon)$ and $C(\langle X\rangle_m, \langle Y\rangle_m)$ defined in Section 5.6 through two setups. In the first setup, we examine the growth rate of the approximate common information as a function of the approximation error ϵ (or the quantization size m). We use the second setup to illustrate the linear relationship between the approximate common information and common information dimension $d(X,Y)$.

In all simulation results below, we obtain the value of approximate common information as follows:

- To measure $C(X_\epsilon, Y_\epsilon)$ (note that there exist multiple sequences[4] $\{(X_\epsilon, Y_\epsilon)\}_{\epsilon>0}$ that satisfy the requirements in (5.27)), we choose two representative sequences and plot the results for

[4]Note that there are at most $2^{\min\{d_X, d_Y\}}$ (X_ϵ, Y_ϵ) that satisfy (5.27), for each ϵ.

both: $\{(\underline{X}_\epsilon, \underline{Y}_\epsilon)\}$ which has the minimum common information among such sequences for all $\epsilon > 0$, and $\{(\overline{X}_\epsilon, \overline{Y}_\epsilon)\}$ which has the maximum common information. We calculate the $C((\underline{X}_\epsilon, \underline{Y}_\epsilon))$ and $C(\overline{X}_\epsilon, \overline{Y}_\epsilon)$ using the closed-form solution in (5.5) [SC15].

- To calculate the ϵ-approximation common information $C_\epsilon(X, Y)$, we solve the optimization problem in (5.28) numerically using SciPy [VGO20].

- For the $C(\langle X\rangle_m, \langle Y\rangle_m)$, we use a pair of upper and lower bounds from the proof of Theorem 19:

$$I(\langle X\rangle_m; \langle Y\rangle_m) \le C(\langle X\rangle_m, \langle Y\rangle_m) \le \sum_{X=Y} H(\langle X_i\rangle_m) + \sum_{X\neq Y} C(X_i, Y_i).$$

In Figure 5.2 and 5.3, the exact value of $C(\langle X\rangle_m, \langle Y\rangle_m)$ lies in the colored area in between.

5.7.1 Setup 1

We let $X \in \mathbb{R}^4$ and $Y \in \mathbb{R}^4$ be jointly Gaussian vectors with zero means and covariance matrices

$$\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{\Sigma}_{XY} = \begin{bmatrix} 1 & 0.5 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0.3 \end{bmatrix}.$$

It is evident that $X_1 = Y_1, X_2 = Y_2, X_3 = Y_3$ almost surely, and rank$(\mathbf{\Sigma}) = 5 <$ rank$(\mathbf{\Sigma}_X) +$ rank$(\mathbf{\Sigma}_Y)$, thus, X and Y are jointly singular.

Figure 5.2 illustrates the normalized common information $\frac{C_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)}$, $\frac{C(X_\epsilon,Y_\epsilon)}{\frac{1}{2}\log(1/\epsilon)}$ and $\frac{C(\langle X\rangle_m,\langle Y\rangle_m)}{\log(m)}$ for the three formulations introduced in Section 5.6.1. They are plotted against the approximation error ϵ (or the quantization size m). We observe that both $\frac{C_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)}$ and $\frac{C(X_\epsilon,Y_\epsilon)}{\frac{1}{2}\log(1/\epsilon)}$ converge to $d(X, Y) =$ rank$(\mathbf{\Sigma}_X) +$ rank$(\mathbf{\Sigma}_Y) -$ rank$(\mathbf{\Sigma}) = 3$ as ϵ approaches 0. Similarly, $\frac{C(\langle X\rangle_m,\langle Y\rangle_m)}{\log(m)}$ converges to $d(X, Y)$ as m approaches ∞. These verify our results in Section 5.6.2. Moreover, they reach a value that is close to $d(X, Y)$ (e.g., a value < 4) quickly, even when ϵ is relatively large

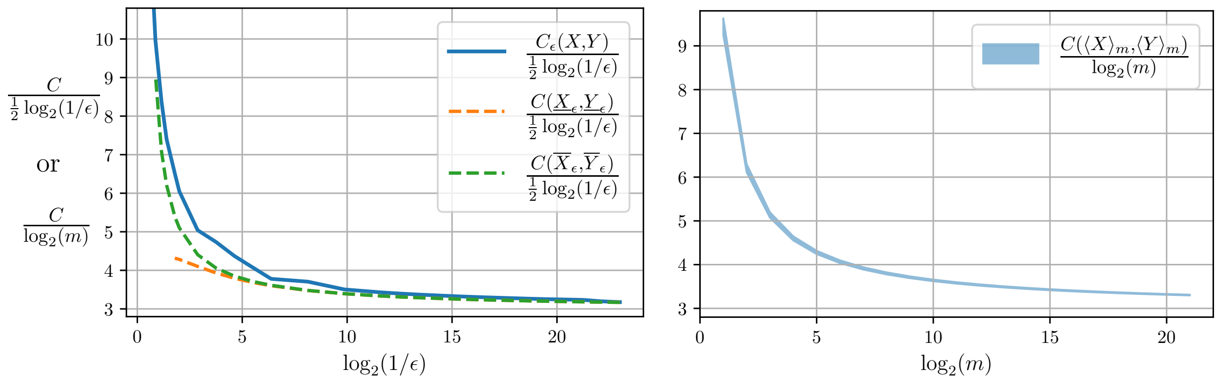


Figure 5.2: (a) the growth rate of $C_\epsilon(X, Y)$ and $C(X_\epsilon, Y_\epsilon)$; (b) the growth rate of $C(\langle X\rangle_m, \langle Y\rangle_m)$.

(or m is relatively small). The trade-off between ϵ-approximate common information $C_\epsilon(X, Y)$ and error ϵ indicates that the common information dimension $d(X, Y)$ provides a theoretical limit of distributed simulation, on the maximum achievable accuracy given a finite number of bits to represent the common randomness; or equivalently, on the minimum number of bits required for the shared randomness to achieve a target simulation accuracy. Similarly, the trade-off between $C(\langle X\rangle_m, \langle Y\rangle_m)$ and the approximation precision m reflects the special case when uniform quantization is applied to the target random variables.

5.7.2 Setup 2

In this example, we use $X \in \mathbb{R}^7$ and $Y \in \mathbb{R}^7$ with $\mathbf{\Sigma}_X = \mathbf{\Sigma}_Y = \boldsymbol{I}_7$, while we choose cross-covariance matrices $\mathbf{\Sigma}_{XY}$ to be a diagonal matrix with $d(X, Y)$ number of diagonal elements set to be 1 and the rest to be 0.5.

In Figure 5.3(a), we plot ϵ-approximate common information (i.e. the minimum number of bits required to approximate X, Y) versus different choices common information dimensions $d(X, Y)$. We use two different levels of accuracy: $\epsilon = 2^{-5}$ and $\epsilon = 2^{-20}$. We observe that the approximate common information grows linearly with the common information dimension $d(X, Y)$, where the slope is given by $\frac{1}{2}\log(1/\epsilon)$, as we also proved in Theorem 18. In addition, this plot provides guidance on the minimum number of bits that need to be shared to perform the distribution simulation within a given error. For instance, to simulate a target distribution with $d(X, Y) = 5$, we

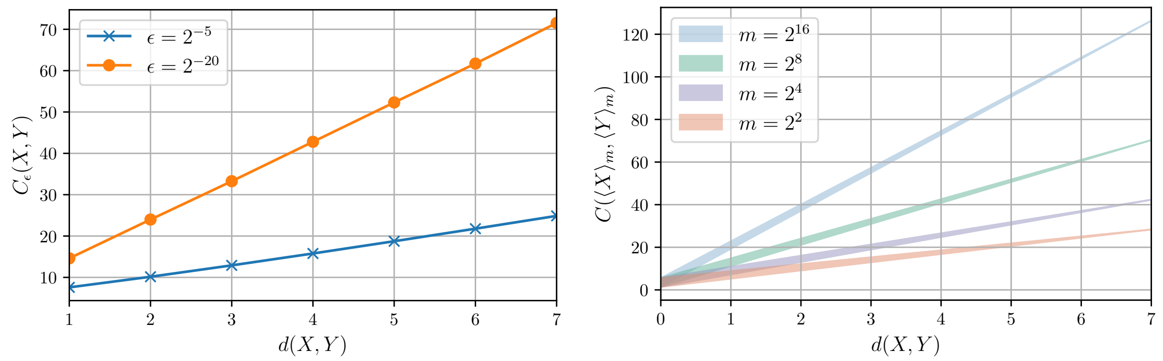


Figure 5.3: Common information dimension $d(X,Y)$ vs (a) the ϵ-approximate common information $C_\epsilon(X,Y)$; (b) the common information of quantized variables $C(\langle X\rangle_m, \langle Y\rangle_m)$.

need to share 19 bits to achieve a relatively low 2^{-5} accuracy, or 53 bits to achieve a relatively high 2^{-20} accuracy.

In Figure 5.3(b), we plot the common information between quantized random variables $\langle X\rangle_m$, $\langle Y\rangle_m$ versus different values of common information dimension $d(X,Y)$. We consider 4 different levels of quantization: $m = 2^k$, where $k \in \{1,2,4,8\}$. We observe that the approximate common information grows linearly with the common information dimension $d(X,Y)$, and the slope is roughly $\log(m)$ for large enough m, as we also proved in Theorem 19. Moreover, this plot specifies the minimum number of bits needed to distributedly simulate target random variables that are uniformed quantized into discrete values, with a known quantization precision.

5.8 Conclusion and Open Research Directions

In this chapter, we introduced the notion of CID which uses dimensions to measure the amount of common information between random variables, that is infinite in bits. We provide a method to compute CID for gaussian distributions and show that it characterizes the growth rate of the common information in the nearly infinite regime.

We leave the following as open direction:

- How to compute CID for general distributions? In this chapter, we only compute CID for

Gaussian distributions.

- A data-driven approach to compute CID. Our method to compute CID require the knowledge of distribution; approximating CID using samples generated from the distribution is an interesting open problem.

Acknowledgement

The work in this chapter has been done in collaboration with Xinlin Li, who is currently a PhD student at UCLA.

CHAPTER 6

Distributed Feature Compression for Classification

6.1 Summary

We consider the problem of distributed feature quantization, where the goal is to enable a pretrained classifier at a central node to carry out its classification on features that are gathered from distributed nodes through communication constrained channels. We propose the design of distributed quantization schemes specifically tailored to the classification task: unlike quantization schemes that help the central node reconstruct the original signal as accurately as possible, our focus is not reconstruction accuracy, but instead correct classification. Our work does not make any a priori distributional assumptions on the data, but instead uses training data for the quantizer design. Our main contributions include: we prove NP-hardness of finding optimal quantizers in the general case; we design an optimal scheme for a special case; we propose quantization algorithms, that leverage discrete neural representations and training data, and can be designed in polynomial-time for any number of features, any number of classes, and arbitrary division of features across the distributed nodes. We find that tailoring the quantizers to the classification task can offer significant savings: as compared to alternatives, we can achieve more than a factor of two reduction in terms of the number of bits communicated, for the same classification accuracy.

6.2 Introduction

Quantization forms the core of almost all lossy data-compression algorithms, and is widely used to reduce the number of bits required for storage and communication. These schemes optimize

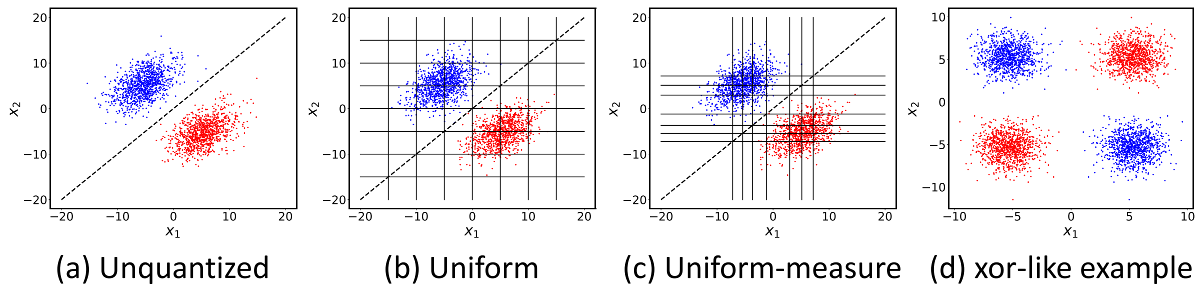


Figure 6.1: Quantization example and xor-like example.

a rate-distortion trade-off, where the goal is to represent data using a limited number of bits as precisely as possible. Instead, in this work, we propose distributed quantization schemes tailored to data that are going to be used for classification. That is, we explore the design of distributed quantizers for a rate-classification error trade-off: our quantization schemes are not optimized for reconstruction accuracy, but instead correct classification.

Fig. 6.1(a)-(c) illustrates the difference between the two aforementioned approaches. Given a number of bits, we create a corresponding number of quantization regions in the space (3 bits/feature = $2^3.2^3$ = 64 regions in our example). Intuitively, for data reconstruction, we want to more finely represent the regions of high signal concentration (Fig. 6.1(b)); for classification, we want to more finely represent areas closer to the classification boundary where errors may happen (Fig. 6.1(c)).

In our work, we aim to design quantizers for the following generic scenario. A central entity has access to a pretrained subdifferentiable classifier (*e.g.,* a multilayer perceptron - MLP [CB15]) and wishes to apply that classifier on data features collected at K distributed sensor nodes. The communication between the sensors and the central entity comes at a cost (is rate limited), and thus it is expensive to send the measured features with full precision. Instead, each node employs a *distributed single-shot* quantizer[1], independently from other nodes, in order to encode its measurements into bit representations that can be sent to the central entity efficiently as soon as sensed. We emphasize that we do not make any a priori distributional assumptions on the data, as is common

[1]Single-shot means that we do not collect samples over time and jointly quantize them. Thus, a set of *local* features observed at a node are quantized together whenever observed; motivated by delay requirement for classification.

in many learning scenarios. Moreover, the data may be heterogeneous, from unknown composite distributions (*e.g.,* multimodal observations of sensors that capture video, sound, and radar signals). We simply use training samples from the data to design the quantizers.

This scenario is motivated by many machine learning applications, that include wireless cyber-physical systems, immersive environments and supported health. For example, in brain-to-computer interface applications, multiple electrodes are placed around the brain to capture brain signals which are used collectively as features to classify in what direction a person is trying to move his hand [LN06]. Such features need to be quantized at the sensor peripheral nodes, and communicated through rate constrained channels to a central node for processing, so that classification (and decisions based on it) can be done within a reasonable time of sensing. Other applications where distributively generated features need to be quantized for communication include distributed environment monitoring in sensor networks [OR11] and real-time control in Internet of Things applications [FF20].

Our assumption of a pretrained classifier is motivated by the following practical considerations: (i) we may not know the communication channel constraints when designing the classifier and we may want to use the same classifier over systems with different communication channels; (ii) we may not have access to the data used to train the classifier (*e.g.,* we use a pretrained classifier from cloud services such as Google Cloud [goo08] or Clarifai [cla]), but are able to personalize the quantizers leveraging locally available data.

Contributions. We begin our formal study of the problem by proving that in general, it is NP-hard[2] to design an optimal distributed quantization system tailored for classifying a given set of data points. We also show that the problem is hard to approximate, therefore motivating alternate approaches to the design, and empirical evaluations of proposed techniques.

Given the difficulty of designing the optimal quantization system, we propose a data-based greedy quantization boundary insertion strategy, which we term `GBI`, which can be used for any

[2]This hardness is in terms of the problem parametrization, *e.g.,* number of training points and the number of features.

type of classifier, any number of features, any number of classes, and arbitrary division of features across the distributed nodes. Operationally, `GBI` creates rectangular quantization regions by greedily deciding how to divide the training data along each feature. We demonstrate that `GBI` has quadratic complexity in the number of training samples and linear in the number of features.

To further reduce complexity and capture richer quantization boundaries (beyond rectangular), we propose a (deep) learning based approach to design our quantizers that makes use of the subdifferentiable nature of the classifier employed by the central node. This is inspired by the recent success of learning discrete latent variables [OVK17,ROV19], joint source channel coding [CTW18] and discrete representations for image compression [TSC17, BLS16, VKK16]. Our design can be understood as a *distributed* discrete neural representation optimized for classification. We leverage the `GBI` algorithm by making it a module within the discrete neural representation.

Through numerical evaluation, we show that for the same representation budget (number of bits available at sensor nodes for each measurement), we can achieve four folds gains in classification accuracy compared to approaches that try to learn discrete representations aimed directly at reconstruction.

Our main contributions can thus be summarized as follows:

• We prove the NP-hardness and hardness to approximation for designing optimal distributed quantizers for classification.

• We design optimal quantizers for linearly separable data and two features under some structural restrictions.

• We propose a polynomial-complexity greedy quantization algorithm, `GBI`, optimized for classification, that can be used for any number of features and any classifier.

• We propose a novel distributed discrete neural representation for classification, which can also be combined with `GBI`.

• When compared with approaches for data reconstruction, we demonstrate benefits of 50% gain in terms of classification accuracy for our proposed quantization approaches, on an sEMG dataset and 300% improvement on the CIFAR10 dataset.

Chapter Organization. Section 6.3, reviews related work; Section 6.4 develops the notation and problem framework; Section 6.5 proves the NP-hardness results; Section 6.6, introduces the `GBI` algorithm; Section 6.7 proposes neural representation schemes; and Section 6.8 presents our numerical evaluation.

6.3 Related Work

We will give representative examples of related literature to put our work in context, with an organization around specific approaches/problems.

Distributed detection and hypothesis testing. The problem studied in this chapter is related to distributed estimation and detection in communication-constrained networks, extensively studied in the literature (see [CV07, Luo05] and references therein). Differently from our work, a common assumption is that sensor measurements are independently distributed given the detection hypothesis, and that these conditional distributions are known. In [LLG90], scalar quantization for distributed hypothesis testing was studied, using *known* conditional distribution of features. In contrast to all these works, we neither assume knowledge of the sensor measurements distribution, nor do we make independence assumptions.

The information-theoretic study through error exponents where features are observed at different nodes, is surveyed in [HA98]. Here, differently from our single-shot setup, an asymptotically long sequence of i.i.d. time samples, from a fixed underlying (unknown) distribution, are jointly compressed to distinguish between two hypotheses (*e.g.,* testing for independence). There have also been several recent works in information theory and machine learning on distributed probability estimation, property testing and simulation [HOW18, DGL17, ACT18]. These works assume that each node observes all features, and has access to independent samples from an unknown underlying distribution. Distinct from this in our setup, each node observes subsets of (non-overlapping) features, *i.e.,* the observations at different nodes are not identically distributed.

Scalar and Vector quantization. In [Poo88, SMG13, MGV11] and references therein, a high-rate quantization theory is developed for computing *known* functions from distributed observations, where the source distributions are known. For binary classification, the work in [BNO18] studies a fundamental limit of designing a centralized deterministic scalar quantizer in terms of the maximum mutual information that can be retained between the quantized features and classification labels in the worst case. Designing *centralized* quantizers for classification can also be done using the learning vector quantization (LVQ) [KHK96, SY96] and generalized vector quantization [RMR96] frameworks, where a number of prototype classified vectors are defined and updated to reduce misclassification error. In contrast, our problem requires decentralized quantization; the feature space partitioning in the aforementioned approaches may not be decomposable into decision boundaries applicable by distributed quantization.

Multi-terminal function computation. Rate-distortion literature has considered several related problems, where asymptotically large number of samples are jointly represented; moreover these problems assume that distribution of the sources are known. In the classical CEO problem [BZV96, VB97, Ooh98, PTR04], a central node reconstructs a value from independently corrupted versions measured at distributed sensors. Distributed compression for functional computation with distortion has been studied in [Wag11, KP09, DSM10]. Our work focuses on single-shot quantization for a priori unknown source distributions, without explicit knowledge of the classifier function.

Model Compression. Quantization is also used in inference tasks for model compression [ZYG17, JKC18, WLC18, GLW19], with the goal to simplify implementation and reduce storage. However, differently from our work, the goal is to quantize the model operands rather than focus on distributively quantizing the inputs to the model.

Decision stumps. A closely related algorithm that could be adapted to use for feature quantization is AdaBoost [FS97, HRZ09] with decision stumps. In this case, the majority rule on the decision stumps naturally partitions (quantizes) the space based on the number of stumps corresponding to each feature. However, AdaBoost with decision stumps will not necessarily be able to return viable quantizers in all cases. For example, if we consider labeled data points with an XOR pattern in $\mathbb{R}^2$ (centered at [-1,-1], [-1,1],[1,-1] and [1,1] as shown in Fig. 6.1(d)) then AdaBoost with stumps is not able to represent the XOR pattern in its decision regions [TDW14]. In contrast, it is not difficult to see that two quantization boundaries at $x_1 = 0$ and $x_2 = 0$, respectively, are enough to allow a good classifier to correctly classify the quantized points.

Latent variable models. Perhaps the closest approach to ours, are those of learning latent representations for data reconstruction. In variational autoencoders (VAEs) [KW13,HMP17,ZSE19,MRS19], a continuous latent representation space is learned from the inputs, that can then be used to reconstruct inputs or generate new data that follow the same distribution as the data in the training set. In [OVK17], the authors present a new way of training VAEs to learn discrete latent space representations, which naturally leads to a compression algorithm, since continuous (or full-precision) inputs can be mapped to discrete latent representations typically using fewer bits. In [CTW18], the authors also study the inference of discrete latent variables for joint source and channel coding. In particular, discrete latent variables are learned such that they can be used for compression as in VQ-VAEs; they are also robust to transmission over noisy discrete channels for reconstruction.

A main difference between these implementations and our setup is the *distributed* (decomposable) structure of our quantization system. In addition, it is intuitive to expect that reconstruction may not yield the best classification results; what is perceived by the reconstruction loss as a good approximation of the image might be inappropriate for a classifier as compared to the performance of a classification tailored approach. We explore the latter point empirically in Section 6.8. Therefore, our work can be thought of as an approach to distributed discrete (neural) representation for classification. Recently, [AFD17] presented a variational approximation to the information bottleneck method [TPB00] to design classifiers. However, differently from our work, it assumed a centralized encoder and continuous latent variables.

6.4 Notation and Problem Formulation

Let $\mathcal{X}^n$ be the n-dimensional space of possible input features (e.g., sensor measurements, images, text, etc.) and $\mathcal{Y}$ be the set of possible classification classes over the space $\mathcal{X}^n$. We use $\mathbf{x} \in \mathcal{X}^n$ to

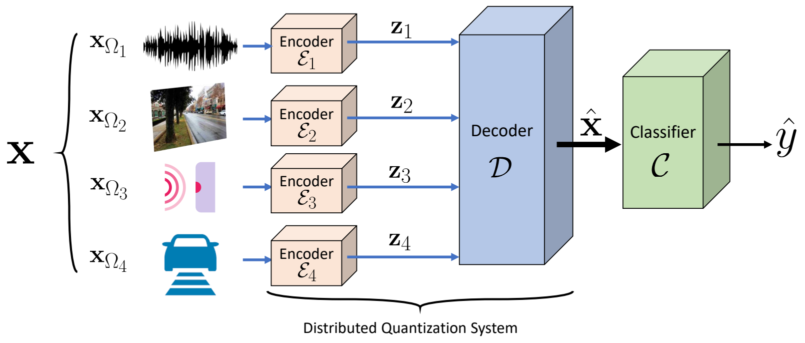


Figure 6.2: An example for distributed quantization of features for classification with $K = 4$ nodes.

denote an n-dimensional input and $y(\mathbf{x}) \in \mathcal{Y}$ to denote the class associated with $\mathbf{x}$.

We consider a scenario where the features of $\mathbf{x}$ are not collected (sensed) all at the same entity but instead at K distributed nodes. An example is shown in Fig. 6.2 for $K = 4$. In particular, node $k \in [1 : K]$ collects the vector of features $\mathbf{x}_{\Omega_k}$ indexed by a set $\Omega_k \subseteq [1 : n]$. We assume that the index sets $\{\Omega_k\}_{k=1}^{K}$ are disjoint and their union is the set $[1 : n]$. We also assume that the feature vector $\mathbf{x}$ is ordered such that $\mathbf{x} = [\mathbf{x}_{\Omega_1}, \mathbf{x}_{\Omega_2}, \cdots, \mathbf{x}_{\Omega_K}]$. We denote by $\mathcal{X}_{\Omega_k}$ the space defined by the features indexed by Ω_k, with $\mathcal{X}^n = \prod_{k=1}^{K} \mathcal{X}_{\Omega_k}$.

A central node wishes to classify the collected input features $\mathbf{x}$ using a pretrained subdifferentiable[3] classifier $\mathcal{C}(.)$ where

$$\mathcal{C}(.) : \mathcal{X}^n \to \mathbb{R}^{|\mathcal{Y}|}, \tag{6.1}$$

and the output class label $\widehat{y}(\mathbf{x})$ is given by

$$\widehat{y}(\mathbf{x}) = \underset{i \in [1:|\mathcal{Y}|]}{\arg\max} [\mathcal{C}(\mathbf{x})]_i. \tag{6.2}$$

Note that $\widehat{y}(\mathbf{x})$ may be different than the true label $y(\mathbf{x})$. With no communication constraints, node k can perfectly convey $\mathbf{x}_{\Omega_k}$ to the central node. Instead, we assume that node k is constrained to use

[3]By "subdifferentiable" classifier, we refer to a classifier that has non-trivial subgradient sets (i.e., non-empty subgradient sets everywhere).

R_k bits (much less than full precision). That is, node k uses a quantizer/encoder $\mathcal{E}_k$, that takes as input $\mathbf{x}_{\Omega_k}$ and produces a discrete representation $\mathbf{z}_k$ from an alphabet $\mathcal{M}_k$ of size at most 2^{R_k}, with

$$\mathbf{z}_k = \mathcal{E}_k(\mathbf{x}_{\Omega_k}) : \mathcal{X}_{\Omega_k} \to \mathcal{M}_k, \quad \forall k \in [1:K]. \tag{6.3}$$

Based on (6.3), we will denote the preimage of $\mathcal{E}_k$ as

$$\mathcal{E}_k^{-1}(\mathbf{z}_k) = \{\mathbf{x}_{\Omega_k} \in \mathcal{X}_{\Omega_k} | \mathcal{E}_k(\mathbf{x}_{\Omega_k}) = \mathbf{z}_k\}. \tag{6.4}$$

Note that the computed $\mathbf{z}_k$ depends only on $\mathbf{x}_{\Omega_k}$, the features available at node k. At the central node, in order to apply the pretrained classifier $\mathcal{C}$, a decoder $\mathcal{D}$ generates $\widehat{\mathbf{x}} \in \mathcal{X}^n$ from $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K]$ and uses it as the input to $\mathcal{C}$. The end-to-end operation, depicted in Fig. 6.2, is given by (6.3) and

$$\begin{aligned}
&\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K] \\
&\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{z}) : \prod_{k=1}^{K} \mathcal{M}_k \to \mathcal{X}^n, \\
&\widehat{y}(\widehat{\mathbf{x}}) = \arg\max_{i\in[1:|\mathcal{Y}|]} [\mathcal{C}(\widehat{\mathbf{x}})]_i.
\end{aligned} \tag{6.5}$$

We refer to a set of encoders $\mathcal{E} = \{\mathcal{E}_k\}_{k=1}^{K}$ and a decoder $\mathcal{D}$ as a *distributed quantization system* $(\mathcal{E}, \mathcal{D})$. Ideally, we would like to use an $(\mathcal{E}, \mathcal{D})$ system that minimizes the probability of misclassification. That is, the encoders and decoder are the solution of the optimization problem

$$\begin{aligned}
&\min_{\mathcal{E},\mathcal{D}:|\mathcal{M}_k|\le 2^{R_k}} \mathbb{E}_{\mathbf{x},y(\mathbf{x})\sim p(\mathbf{x},y(\mathbf{x}))}[\mathbb{I}(\widehat{y}(\widehat{\mathbf{x}}) \neq y(\mathbf{x}))] \\
&= \min_{\mathcal{E},\mathcal{D}:|\mathcal{M}_k|\le 2^{R_k}} \mathbb{E}_{\mathbf{x},y(\mathbf{x})\sim p(\mathbf{x},y(\mathbf{x}))}[\mathbb{I}(\widehat{y}(\mathcal{D}(\mathcal{E}(\mathbf{x}))) \neq y(\mathbf{x}))],
\end{aligned} \tag{6.6}$$

where: (i) $p(\mathbf{x}, y(\mathbf{x}))$ is the input data distribution; (ii) $\widehat{y}(\widehat{\mathbf{x}})$ and $\widehat{\mathbf{x}}$ are obtained from $\mathbf{x}$ using (6.3) and (6.5); and in (iii) we used $\mathbf{z} = \mathcal{E}(\mathbf{x})$ for brevity.

However, in this work we assume that the distribution $p(\mathbf{x}, y(\mathbf{x}))$ is not known: instead, we are given a dataset $\mathcal{T} = \{(\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)}))\}_{i=1}^{N}$ which contains N independent samples drawn from

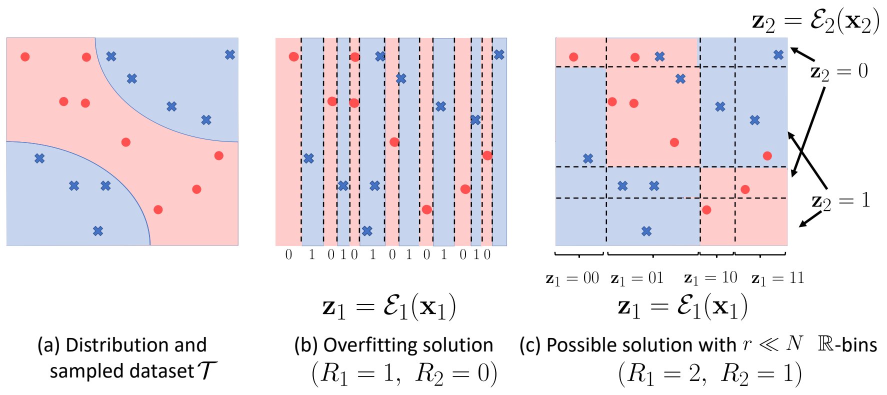

$\mathbf{z}_2 = \mathcal{E}_2(\mathbf{x}_2)$
$\mathbf{z}_2 = 0$
$\mathbf{z}_2 = 1$
0 1 0 1 0 1 0 1 0 1 0 1 0
$\mathbf{z}_1 = 00$ $\mathbf{z}_1 = 01$ $\mathbf{z}_1 = 10$ $\mathbf{z}_1 = 11$
$\mathbf{z}_1 = \mathcal{E}_1(\mathbf{x}_1)$
$\mathbf{z}_1 = \mathcal{E}_1(\mathbf{x}_1)$
(a) Distribution and sampled dataset $\mathcal{T}$
(b) Overfitting solution $(R_1 = 1,\ R_2 = 0)$
(c) Possible solution with $r \ll N$ $\mathbb{R}$-bins $(R_1 = 2,\ R_2 = 1)$

Figure 6.3: Quantization example.

$p(\mathbf{x}, y(\mathbf{x}))$. Thus, we can only empirically approximate the expectation in (6.6) using the dataset $\mathcal{T}$, and hence, our objective is to minimize the *misclassification loss* $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T})$, calculated as

$$\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) = \frac{1}{N}\sum_{i=1}^{N}[\mathbb{I}(\widehat{y}(\widehat{\mathbf{x}}^{(i)}) \neq y(\mathbf{x}^{(i)}))]. \tag{6.7}$$

In the rest of the chapter, we will say that a distributed quantization system $(\mathcal{E}, \mathcal{D})$ is *optimal*, if the encoders $\mathcal{E} = \{\mathcal{E}_k\}_{k=1}^{K}$ and the decoder $\mathcal{D}$ are an optimal solution of the problem

$$\min_{\mathcal{E},\mathcal{D}:|\mathcal{M}_k|\leq 2^{R_k}} \mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}). \tag{6.8}$$

Remark 3. Note that in (6.7), given a labeled dataset $\mathcal{T}$, our objective is to minimize the empirical probability of misclassifying the data points $\mathbf{x}^{(i)}$ after quantization. Instead, if we are given a local dataset of unlabeled data points, $\mathcal{T}_u = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$, we could create a labeled dataset, $\widehat{\mathcal{T}} = \{(\mathbf{x}^{(i)}, \widehat{y}(\mathbf{x})\}_{i=1}^{N}$, by applying the pretrained classifier $\mathcal{C}$ on the local unlabeled data $\mathcal{T}_u$. We can then apply (6.7) - and the logic in the remainder of the chapter - on the data $\widehat{\mathcal{T}} = \{(\mathbf{x}^{(i)}, \widehat{y}(\mathbf{x})\}_{i=1}^{N}$. In this case, our objective is equivalent to keeping the classifier output consistent before and after applying the distributed quantization system.

Remark 4. With no structural restrictions on the encoders $\mathcal{E}$, it is possible to achieve $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) = 0$ almost-surely through over-fitting. For instance, if the distribution $p(\mathbf{x})$ is a continuous distribution, the probability that two data points have the same value for $\mathbf{x}_{\Omega_k}$ is zero for any k. Thus, we can consider for example the first node ($k = 1$), and partition the space of $\mathbf{x}_{\Omega_1}$ into N disjoint regions such that each region contains only one training data point. Then, for each region, the encoder function $\mathcal{E}_1(.)$ at node 1 can directly output the class $y(\mathbf{x}^{(i)})$ of the data point $\mathbf{x}^{(i)}$ contained in that region, then the decoder outputs a data point that is classified by the classifier to be $y(\mathbf{x}^{(i)})$. This requires only $\log_2(|\mathcal{Y}|)$ bits. Hence, the rates $[R_1, ..., R_K] = [\log_2(|\mathcal{Y}|), 0, ..., 0]$ are sufficient to achieve $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T})$=0. Although such a quantization scheme will have zero loss when evaluated on the dataset $\mathcal{T}$, it would obviously handle out of sample points very poorly. To avoid this, we restrict the preimage $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ to be the union of at most r $\mathcal{X}_{\Omega_k}$-bins, which are defined below.

Definition 1 ($\mathcal{S}$-bin)**.** We say that the set $\mathcal{A} \subseteq \mathcal{S}$ is an $\mathcal{S}$-bin if $\mathcal{A}$ is path-connected [Mun14] in $\mathcal{S}$. A set $\mathcal{A}$ is path-connected if and only if for every pair of points $a, b \in \mathcal{A}$, there exists a path that connects a, b which completely lies inside $\mathcal{A}$. More formally, a set $\mathcal{A}$ is said to be path-connected if and only if for every pair of points $a, b \in \mathcal{A}$, there exists a continuous function $f : [0, 1] \to \mathcal{A}$ such that $f(0) = a$ and $f(1) = b$ [Mun14].

By restricting the preimage of $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ to $r \ll N$ $\mathcal{X}_{\Omega_k}$-bins, we force $\mathcal{E}_k$ to assign the same $\mathbf{z}_k$ to a limited number of path-connected regions (earlier, these could be as many as the number of data points from the same class). By doing so, the solution discussed above (where a single encoder can fully carry the burden of classifying the data points) is eliminated.

We illustrate how the introduction of this restriction can reduce overfitting in the learned quantization system through the example shown in Fig. 6.3. Fig. 6.3(a) depicts the underlying true class function $y(\mathbf{x})$ through colored regions of $\mathbb{R}^2$ and the sampled dataset $\mathcal{T}$ as points scattered in the plot. Fig. 6.3(b) shows how an overfitting quantization system (as described in Remark 4) using only $\mathbf{x}_1$ would approximate the underlying class function y. Note that, although the resulting system provides poor approximation, it classifies the dataset points perfectly (given by the background color in each region). Finally, Fig. 6.3(c) shows an example where each encoder $\mathcal{E}_k$ assigns at most 2 $\mathbb{R}$-bins to the same $\mathbf{z}_k$. It is not difficult to see that although the illustrated quantization system does misclassify some points in the dataset (decision given by the background in each region), it gives a better approximation of $y(\mathbf{x})$ outside the given dataset compared to the design in 6.3(b).

A summary of the notation used throughout the chapter is given in Table 6.1.

Symbol	Description	Symbol	Description
K	Number of distributed sensing nodes	$\mathcal{M}_k$	Set of possible values for $\mathbf{z}_k$ (**def.** (6.3))
n	Number of features of data point $\mathbf{x}$	$\mathbf{z}$	Collection of encoder outputs $[\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K]$ (**def.** (6.5))
$\mathcal{Y}$	Set of possible classes	$\mathcal{M}$	Set of possible values for $\mathbf{z}$ (**def.** (6.5))
y	True class label of data point $\mathbf{x}$	$\widehat{\mathbf{x}}$	Reconstructed input from $\mathbf{z}$ using $\mathcal{D}$ (**def.** (6.5))
Ω_k	Set of features at node k	$\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T})$	Empirical misclassification loss (**def.** (6.7))
R_k	Number of bits/data point at node k	$\mathcal{S}$-bin	Path-connected subset of $\mathcal{S}$ (**def.** Definition 1)
$\mathcal{C}$	Pretrained classifier (**def.** (6.1))	p_e	Misclassification loss threshold (**def.** Lemma 5)
$\widehat{y}$	Output class by classifier $\mathcal{C}$ (**def.** (6.2))	d	Set of boundaries used in optimal on-the-line quantizer
$\mathcal{E}_k$	Encoder at node k (**def.** (6.3))	$f_k(\cdot; \theta_k)$	Neural network of neural driven encoder $\mathcal{E}_k$
$\mathcal{E}_k^{-1}$	Preimage of $\mathcal{E}_k$(**def.** (6.4))	$\mathbb{R}^{m_k}$	Output space of neural encoder f_k (**def.** (6.15))
$\mathcal{E}$	Collection of encoders $\{\mathcal{E}_k\}_{k=1}^K$	$\mathcal{Q}_k$	Quantizer of neural driven encoder $\mathcal{E}_k$ (**def** (6.15))
$\mathcal{D}$	Decoder at central node (**def.** (6.5))	$\mathcal{G}$	Initial mapping of neural driven decoder $\mathcal{D}$ (**def.** (6.16))

$\mathcal{T}$	Training dataset $\{(\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)})\}_{i=1}^{N}$	$g(\cdot;\phi)$	Neural network of neural driven decoder $\mathcal{D}$ (**def.** (6.16))
N	Number of data points in $\mathcal{T}$	$\mathcal{L}^{\mathrm{c}}$	Misclassification loss for neural based approach (**def.** (6.18))
$\mathbf{z}_k$	Output of encoder $\mathcal{E}_k$ (**def.** (6.3))	$\mathcal{L}^{\mathrm{q}}$	Quantization loss for neural based approach (**def.** (6.20))

Table 6.1: Notation used throughout Chapter 6.

6.5 On the Complexity of Finding an Optimal Distributed Quantization System

In this section we study the complexity of finding an optimal quantization system $(\mathcal{E}, \mathcal{D})$ that minimizes the loss in (6.8) over the dataset $\mathcal{T}$. We first start by describing how to find an optimal decoder $\mathcal{D}$ assuming that the optimal encoders $\mathcal{E}^\star = \{\mathcal{E}_k^\star\}_{k=1}^{K}$ are given. We then discuss the complexity of finding optimal encoders and show that the problem is NP-hard in all cases but one. For the case where the problem is not NP-hard, we propose a polynomial-time algorithm to find the optimal quantization system (encoders/decoder) under some structural restrictions on the encoders.

6.5.1 Optimal Decoder

Assuming that the optimal encoders $\mathcal{E}^\star$ are given, we are interested in a decoder $\mathcal{D}^\star$ that minimizes the misclassification loss in (6.8). For brevity, let us denote the set of all possible encoded values

$\mathbf{z} = \mathcal{E}^\star(\mathbf{x})$ as $\mathcal{M}$, i.e.,

$$\mathbf{z} \in \mathcal{M}, \quad \text{where} \quad \mathcal{M} = \prod_{k=1}^{K} \mathcal{M}_k. \tag{6.9}$$

The operation of the optimal decoder is described in the following lemma.

Lemma 4. For given fixed encoders $\mathcal{E}^\star$, the optimal decoder $\mathcal{D}^\star$ is defined by

$$\mathcal{D}^\star(\mathbf{z}) = \widehat{\mathbf{x}} \quad \text{s.t.} \quad \widehat{y}(\widehat{\mathbf{x}}) = \arg\max_{c \in \mathcal{Y}} \sum_{i:\mathcal{E}^\star(\mathbf{x}^{(i)})=\mathbf{z}} \mathbb{I}\left[y(\mathbf{x}^{(i)}) = c\right], \tag{6.10}$$

and $\widehat{y}$, defined in (6.2), is the label output of classifier $\mathcal{C}$ for $\widehat{\mathbf{x}}$.

Proof. The misclassification loss in (6.7) can be rewritten as

$$\mathcal{L}(\mathcal{E}^\star, \mathcal{D}, \mathcal{T}) = \frac{1}{N} \sum_{\mathbf{z} \in \mathcal{M}} \sum_{i:\mathcal{E}^\star(\mathbf{x}^{(i)})=\mathbf{z}} \mathbb{I}\left[y(\mathbf{x}^{(i)}) \neq \widehat{y}(\mathcal{D}(\mathbf{z}))\right], \tag{6.11}$$

where $\widehat{y}(\mathcal{D}(\mathbf{z}))$ is obtained by (6.5). Since for a fixed $\mathbf{z}$, $\mathcal{D}(\mathbf{z})$ only affects one term in the outer summation in (6.11), each of the outer summation terms can be independently minimized by choosing $\mathcal{D}^\star(\mathbf{z})$ to be a point $\widehat{\mathbf{x}} \in \mathcal{X}^n$ satisfying

$$\begin{aligned}
\widehat{y}(\widehat{\mathbf{x}}) &= \arg\min_{c \in \mathcal{Y}} \sum_{i:\mathcal{E}^\star(\mathbf{x}^{(i)})=\mathbf{z}} \mathbb{I}\left[y(\mathbf{x}^{(i)}) \neq c\right] \\
&= \arg\min_{c \in \mathcal{Y}} \left(N - \sum_{i:\mathcal{E}^\star(\mathbf{x}^{(i)})=\mathbf{z}} \mathbb{I}\left[y(\mathbf{x}^{(i)}) = c\right] \right) \\
&= \arg\max_{c \in \mathcal{Y}} \sum_{i:\mathcal{E}^\star(\mathbf{x}^{(i)})=\mathbf{z}} \mathbb{I}\left[y(\mathbf{x}^{(i)}) = c\right].
\end{aligned} \tag{6.12}$$

That is, $\widehat{\mathbf{x}} = \mathcal{D}^\star(\mathbf{z})$ can be any point in $\mathcal{X}^n$ such that $\widehat{y}(\widehat{\mathbf{x}})$ (the decision of classifier $\mathcal{C}$ for $\widehat{\mathbf{x}}$) is the majority true label $y(\mathbf{x})$ among the points of the dataset $\mathcal{T}$ that fall in $\mathcal{E}^{-1}(\mathbf{z})$. For instance, in the example shown in Fig. 6.3, if $\mathcal{E}^{\star-1}(\mathbf{z})$ has one "x" (blue) and two "o" (red) training points, any point that the classifier $\mathcal{C}$ would classify to be "o" (red) can be selected as $\widehat{\mathbf{x}} = \mathcal{D}^\star(\mathbf{z})$. Thus, $\mathcal{D}^\star$ manipulates the classifier $\mathcal{C}$ to output a classification that best serves the loss function in (6.11). $\square$

With the optimal decoder in mind, we are now ready to discuss the hardness of the problem of finding the optimal encoders $\mathcal{E}$ in the following subsection.

6.5.2 Hardness of Finding an Optimal Quantizer

Given a training dataset $\mathcal{T} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, our goal is to design an *optimal* distributed quantization system $(\mathcal{E}, \mathcal{D})$ which minimizes the misclassification loss in (6.7) for a given communication budget of R_k bits per data point at each node k. We study four different cases of the problem:

(P1) For number of features $n > 1$, number of classes $|\mathcal{Y}| > 1$, dataset $\mathcal{T} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ and given $\{R_k\}_{k=1}^{K}$: find the optimal ($\mathcal{E}$, $\mathcal{D}$) that minimizes the misclassification loss $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T})$, assuming that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is the union of $r < N$ $\mathcal{X}_{\Omega_k}$-bins.

(P2) Restricting (**P1**) to the case of linearly separable data.

(P3) Restricting (**P1**) so that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is a single $\mathcal{X}_{\Omega_k}$-bin.

(P4) Restricting (**P3**) to the case of linearly separable data.

Next, we prove that the first three problems for $n > 1$ and the last problem for $n > 2$ are NP-hard and also prove their hardness of approximation. In the general case, as a result of the hardness and hardness of approximation, we focus on finding heuristic approaches to find a good distributed quantization solution that may not necessarily be optimal.

Remark 5. For problem (**P4**), we prove the hardness results for number of features $n \geq 3$. In the next subsection, we introduce optimal polynomial-time algorithm for the case $n = 2$ under some structural restrictions on the encoders.

For all the problems, in order to prove the goal results, it is sufficient to consider prototype settings with predefined number of features n, number of classes $|\mathcal{Y}|$, number of nodes K, and communication budget $\{R_k\}_{k=1}^{K}$ and allow the size N of the dataset $\mathcal{T}$ to grow. It follows that the general problems, which are expansions of these prototype problems, are also NP-hard. In particular, in all cases, we assume that each distributed node quantizes only one feature (i.e., $n = K$) and the number of classes $|\mathcal{Y}| = 2$. The remaining parameters are defined below for each problem

$$\begin{aligned} &\bullet\ (\mathbf{P1})\ \&\ (\mathbf{P2}) : n = K = 2,\ \text{finite } R_1 \text{ and } R_2 \to \infty; \\ &\bullet\ (\mathbf{P3}) : n = K = 2,\ \text{finite } R_1 = R_2; \\ &\bullet\ (\mathbf{P4}) : n = K = 3,\ R_3 = 0 \text{ and finite } R_1 = R_2. \end{aligned} \tag{6.13}$$

We start by showing that, under polynomial-time reductions, the problem of finding the optimal quantization system in (**P1**)- (**P4**) with the aforementioned parameters is equivalent to finding the minimum number of bits $\{R_k\}_{k=1}^{K}$ required for a particular fixed misclassification error p_e. In particular, the equivalence is summarized in the following lemma.

Lemma 5. For a fixed $p_e \in [0, 1]$ and number of classes $|\mathcal{Y}| = 2$: (**P1**) - (**P4**) with parameters in (6.13) are equivalent to (**P1'**) - (**P4'**) below under polynomial-time reductions:

(**P1'**) For number of features $n = 2$, $R_2 \to \infty$, finding the minimum R_1 for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < p_e$, assuming that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is the union of $r < N$ $\mathcal{X}_{\Omega_k}$-bins.

(**P2'**) Restriction of (**P1'**) to the case of linearly separable data.

(**P3'**) For number of features $n = 2$, finding the minimum $R_1 = R_2$ for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < p_e$, assuming that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is a single $\mathcal{X}_{\Omega_k}$-bin.

(**P4'**) For number of features n=3, R_3=0, finding the minimum $R_1 = R_2$ for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < p_e$ for the case of linearly separable data, assuming that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is a single $\mathcal{X}_{\Omega_k}$-bin.

Proof. The proof is based on the observation that the loss $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T})$ can only take one of the $N+1$ values: 0, $\frac{1}{N}$, $\frac{2}{N}$, ..., $\frac{N}{N}$, and 2^{R_1} can only take values in $[1 : N]$. Hence, if we have a polynomial-time algorithm which solves problem (**P1**) in $O(f(N))$, then we can answer the mentioned question in $O(Nf(N))$ by finding the minimum loss $\forall R_1 \in [\log_2(1), \log(2), \cdots, \log_2(N)]$ and pick the smallest R_1 for which the minimum achieved loss is less than p_e. Similarly, if we have a polynomial-time algorithm that answers this question, then we can solve problem (**P1**) in polynomial time.

Hence, (**P1'**) and problem (**P1**) are equivalent under polynomial-time reduction. Following the same logic, problem (**P2**) - (**P4**) are equivalent to (**P2'**) - (**P4'**). □

Based on Lemma 5, the hardness results can now be proved by working directly with (**P1'**) - (**P4'**). In particular, these problems are NP-hard as stated in the following theorem.

Theorem 20. *For a fixed $p_e \in [0,1]$ and number of classes $|\mathcal{Y}|=2$, **(P1') - (P4')** are NP-hard. Moreover, we have that*

- *Approximating 2^{R_k} in problems **(P1')**, **(P2')** within $O(N^{1-\epsilon})$ is NP-hard $\forall \epsilon > 0$.*
- *Approximating 2^{R_k} in problems **(P3')**, **(P4')** within $O(N^{\frac{1}{2}-\epsilon})$ is NP-hard $\forall \epsilon > 0$ assuming the Small Set Expansion Hypothesis (SSEH) and that NP $\nsubseteq$ BPP.*

Proof. We prove each statement by reduction from an NP-Complete problem. In particular, we prove the result for (**P1'**), (**P2'**) by reduction from the vertex coloring problem, and for (**P3'**), (**P4'**) by reduction from the maximum balanced biclique problem. The proof is delegated to Appendix E.4. □

6.5.3 Optimal Quantizer for Linear Classifiers in 2D

In the previous section, we proved that, for problem (**P4**) when $n > 3$, it is NP-hard to find an optimal quantization system. In this subsection, we propose an optimal polynomial-time algorithm for (**P4**) when the number of features is $n = 2$ under some structural restrictions on the encoders. Specifically, we consider a system with two distributed nodes. Each node $k \in [1{:}2]$, observes one feature x_k, and aims to quantize x_k using R bits. We assume that we have two classes ($|\mathcal{Y}| = 2$) to distinguish among and that the data is linearly separable, namely, $\mathcal{C}$ is a linear classifier with output $\widehat{y}(\mathbf{x}) = y(\mathbf{x})$. Moreover, without loss of generality, we assume that the features are scaled and translated such that the line $x_1{=}x_2$ separates the data. This can be performed during encoding at each distributed node and reverted in the decoder $\mathcal{D}$ at the central node. Note that in this case, since $\mathcal{X}_{\Omega_k}{=}\mathbb{R}, \forall k \in [1{:}2]$ and an $\mathbb{R}$-bin is an interval $[a, b]$ for some a, b, then the encoder/quantizer at node

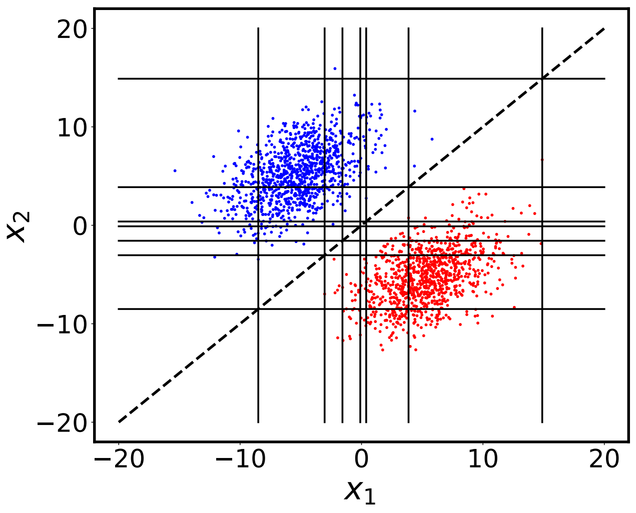


Figure 6.4: Example of `on-the-line` quantizer, where boundaries for x_1 and x_2 intersect along the $x_1 = x_2$ line.

k divides $\mathcal{X}_{\Omega_k} = \mathbb{R}$ into 2^R intervals by introducing the quantization boundaries $(d_{k,1}, \cdots, d_{k,2^R-1})$. We here further restrict our attention to the class of **on-the-line** quantizers, where the horizontal and vertical lines defining $d_{k,i}$ meet along the line $x_1 = x_2$ (as in Fig. 6.4). In other words, nodes 1 and 2 use a common encoder design $\mathcal{E}_1 = \mathcal{E}_2$. This implies that $d_{0,k} = d_{1,k} = d_k$, and thus we simply need to find the $2^R - 1$ quantization boundaries $(d_1, \cdots, d_{2^R-1})$.

Remark 6. Note that although d_k can take any value in $\mathbb{R}$, only $2N$ values can make a difference in the misclassification loss in (6.7): the $2N$ values corresponding to either coordinate of the training data points $\{\mathbf{x} | (\mathbf{x}, y(\mathbf{x})) \in \mathcal{T}\}$. Indeed, these are the only boundaries that can change the bin to which a training point belongs[4].

To find the on-the-line optimal quantizer, we could simply do an exhaustive search over all possible $2N$ values (recall Remark 6) that each of the boundaries d_k can take which costs a complexity of $O\left(\binom{2N}{2^R}\right)$, which is not efficient. Instead, we use a recursive approach, that is based on the following two key observations.

[4]If a point lies on a boundary, we assume it belongs to the bin preceding that boundary.

Remark 7. The encoders/quantizers $\mathcal{E}_1, \mathcal{E}_2$ decompose $\mathbb{R}^2$ into 2^{2R} $\mathbb{R}^2$-bins. However, since the data points are assumed to be linearly separable by the line $x_1 = x_2$, then we only need to consider the $\mathbb{R}^2$-bins crossed by $x_1 = x_2$ as the sources of misclassification. In particular, any other $\mathbb{R}^2$-bin is completely populated by data points from the same class.

Remark 8. Assume that a vector s lists the $2N$ possible boundary values in ascending order, i.e., $s_j \leq s_i$ for all $j \leq i \leq 2N$. Let $\mathcal{T}_{s_i}$ denote the subset of the dataset $\mathcal{T}$ such that both coordinates of $\mathbf{x}$ are upper bounded by s_i, i.e.,

$$\mathcal{T}_{s_i} = \{(\mathbf{x}, y(\mathbf{x})) \in \mathcal{T} | x_1, x_2 \leq s_i\}, \ \forall i \in [1 : 2N]. \tag{6.14}$$

Note that $\mathcal{T}_{s_j} \subseteq \mathcal{T}_{s_i}, \forall i < j$ and that $\mathcal{T}_{s_{2N}} = \mathcal{T}$. Then, the optimal quantizer with b boundaries on $\mathcal{T}_{s_i}$ shares $b-1$ boundaries with the optimal one with $b-1$ boundaries on $\mathcal{T}_{s_j}$ for some $j<i$.

Remark 8 is restated and proved in Appendix E.1. Observations in Remark 7 and Remark 8 lead to a polynomial-time dynamic-programming algorithm to design the optimal quantization boundaries $(d_1, \cdots, d_{2^R-1})$.

The algorithm's pseudo code is given in Algorithm 10 and implements the following logic: Given an ordered list of the possible $2N$ boundary values s, let $E(s_i, b)$ be the minimum number of misclassified points over the dataset subset $\mathcal{T}_{s_i}$ when using b boundaries and $A(s_i, b)$ be the set of boundaries that achieve this. Then in each iteration $i \in [1 : 2N]$:

1. Find $E(s_i, b), \forall b \in [1 : 2^R - 1]$ by trying to augment $A(s_j, b-1), \forall j < i$ with one extra boundary at s_j (Remark 8). The additional number of misclassified points is only a result of the points in $\{(\mathbf{x}, y(\mathbf{x})) \in \mathcal{T} | s_j < x_1 \leq s_i, s_j < x_2 \leq s_i\}$ (Remark 7);
2. Retain the best augmentation $A(s_i, b)$ to be used in the following iteration;
3. After $2N$ iteration, the optimal quantization boundaries are stored in $A(s_{2N}, 2^R - 1)$.

In the worst case the algorithm does $2N \times 2^R$ iterations over the whole dataset $\mathcal{T}$ resulting in a time-complexity of $O(N^2 2^R)$. The optimality of Algorithm 10 is proved in Appendix E.1 by proving the observation in Remark 8.

Algorithm 10 Optimal on-the-line quantizer for linearly separable data in $\mathbb{R}^2$

```
Input: (a) Training set $\{(\mathbf{x}^{(i)},, y(\mathbf{x}^{(i)}))\}_{i=1}^N$;
(b) Quantization bits/feature $R$;
(c) Ordered set s of potential boundaries.
By $\mathbf{x} \preceq p$ we express that all elements in $\mathbf{x}$ are less than $p$; $p \prec \mathbf{x}$ means they all exceed $p$
Output: Quantization boundaries $(d_1, d_2, \cdots, d_{2^R-1})$ to use for features $x_1$ and $x_2$
Initialize:
    $E(s,0) \leftarrow \min_{c\in\{1,2\}} |\{j|\mathbf{x}^{(j)} \preceq s,\ y^{(j)} = c\}|$    for $s \in \mathbf{s}$
for $i \in [1 : |\mathbf{s}|]$ do
    for $b \in [1 : 2^R - 1]$ do
        $E(s_i, b) \leftarrow \min_{\ell<i} \Big\{ E(s_\ell, b-1)$
                $+ \min_{c\in\{1,2\}} |\{j|s_\ell \prec \mathbf{x}^{(j)} \preceq s_i,\ y^{(j)} = c\}| \Big\}$
        $\ell^\star \leftarrow$ index $\ell$ that gave the minimum value for $E(s_i, b)$ in the previous expression
        $A(s_i, b) \leftarrow A(s_{\ell^\star}, b-1) \cup \{s_{\ell^\star}\}$
return $A(s_{|\mathbf{s}|}, 2^R - 1)$
```

In the following section, we introduce an approach for designing the encoders and decoder in a more general setting, i.e., when data points are not necessarily linearly separable and the number of nodes and features are ≥ 2.

6.6 Greedy Boundary Insertion (GBI) Quantizer

We refer the reader to Table 6.1 for the system notation used in this section. Here, we propose our Greedy Boundary Insertion (GBI) algorithm to design encoders/quantizers $\mathcal{E}_k, \forall k \in [1 : K]$, that can be executed in polynomial-time in the dataset size N and the number of features n for any number of classes. For the decoder $\mathcal{D}$, we use the optimal decoder derived in Section 6.5.1. GBI extends the intuition in the observations in Subsection 6.5.3 to a more general case, where the classifier is arbitrary, and where each distributed node k observes Ω_k features and can have arbitrary rate R_k. We design encoders/quantizers such that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ (see (6.4)) is a single $\mathcal{X}_{\Omega_k}$-bin. Note that, since we are not constrained to use the same boundaries for each feature as in the on-the-line case, we consider N possible boundary values per feature, the values taken at that feature by the N training points.

The logic behind `GBI` is as follows. `GBI` iteratively adds quantization boundaries selected greedily: at each iteration it selects to add one of the possible N boundaries to one of the n features, the one that minimizes the misclassification loss in (6.8) given the choice of boundaries in the previous iterations[5]. A feature i can accept a new boundary, if $i \in \Omega_k$ for some node k and introducing a new boundary for feature i does not cause node k to have more than 2^{R_k} $\mathbb{R}^{|\Omega_k|}$-bins. The algorithm terminates when none of the features can accept a new boundary. If two or more possible boundaries lead to the same loss (something that happened surprisingly often in our experiments), then instead of breaking ties at random, it makes a significant performance difference to break ties by using a non-linear criterion. This criterion penalizes a boundary that leaves $\mathbb{R}^n$-bins with high individual misclassification to correct classification ratio. This is discussed in more detail in Appendix E.2.

The pseudo code for `GBI` is presented in Algorithm 11. The losses computed in Algorithm 11 assume that the optimal decoder (in Lemma 4) for the designed encoders is used.

Complexity of `GBI`. At each iteration of the algorithm, we compute the reduction in misclassification error associated with every potential boundary and pick the boundary with the most reduction. This involves $O(N)$ operations per boundary. Thus to add a single boundary, $O(nN^2)$ operations are needed in the worst-case. This results in time-complexity of $O(nN^2 2^{R_{\max}})$, where $R_{\max} = \max_k R_k$. Recall that our focus is on cases where the number of bits used are much lower than required for full precision (32 bits). As a result, the contribution of R_k in the complexity term can be subsumed into the notation $O(nN^2)$.

[5]Since GBI adds a boundary for one feature at a time, instead of a function of the features $\mathbf{x}_{\Omega_k}$, we end up with an encoder of the form of a rectangular-grid, where each region is assigned to a value of $\mathbf{z}_k$.

Algorithm 11 `GBI` Algorithm

```
Variables: •$\{\mathbf{d}\}_{f=1}^{n}$ : Boundaries for feature $f\in[1{:}n]$;
•$B_k$ : Number of $\mathbb{R}^{\Omega_k}$-bins used by node $k \in [1{:}K]$ using boundaries $\{\mathbf{d}_f\}_{f\in\Omega_k}$;
•$\mathcal{L}(\mathbf{d}_1, ..., \mathbf{d}_n)$ : misclassification loss using encoders defined by $\{\mathbf{d}\}_{f=1}^{n}$;
•$\widehat{B}_k(f, \{\mathbf{d}_j\}_{j\in\Omega_k})$ : New $B_k$, if a new boundary is added for feature $f$.
Input: (a) Training set $\{(\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)}))\}_{i=1}^{N}$;
        (b) Quantization bits/node $R_k, \forall k \in [1 : K]$.
Output: Boundaries $\mathbf{d}_f{=}\{d_{f,1}, d_{f,2}, \cdots\}, \forall f \in [1 : n]$
Initialize:
  1) $\mathbf{s}_f \leftarrow \{x_f^{(i)}\}_{i=1}^{N}$ (potential boundaries for feature $f$)
  2) $B_k = 0,\ \forall k \in [1 : K], \quad \mathbf{d}_f = \phi,\ \forall f \in [1 : n]$
while $\mathcal{K}{=}\{k : \min_{f\in\Omega_k} \widehat{B}_k(f, \{\mathbf{d}_j\}_{j\in\Omega_k}){\le}2^{R_k}\} \neq \phi$ do
    - Among $\forall k \in \mathcal{K}$, find $\widehat{f} \in \Omega_k,\ \widehat{d} \in \mathbf{s}_{\widehat{f}}$ that minimizes $\mathcal{L}(\mathbf{d}_1, ..., \mathbf{d}_{\widehat{f}} \cup \widehat{d}, ..., \mathbf{d}_n)$.
    - Break ties using a non-linear criterion
     (See Appendix E.2)
    Update: $\mathbf{d}_{\widehat{f}} \leftarrow \mathbf{d}_{\widehat{f}} \cup \{\widehat{d}\}$
            $B_k \leftarrow B_k + \Delta_k(\widehat{f}, \{\mathbf{d}_j\}_{j\in\Omega_k})$, where $\widehat{f} \in \Omega_k$
```

Remark 9. Despite the fact that `GBI` is a polynomial-time algorithm, we are interested in approaches with linear complexity in N, as the number of available data points (as well as features) in a dataset can be large. To overcome the effect of quadratic complexity in N, `GBI` can be applied stochastically by randomly sampling a subset of the dataset $\mathcal{T}$ to use at each iteration (instead of evaluating the decrease in misclassification over the whole dataset $\mathcal{T}$).

Remark 10. A possible drawback of `GBI` is that boundaries are directly introduced on the native features without transformation. Thus, as aforementioned, the resulting encoder $\mathcal{E}_k$ at node k would always have a rectangular grid structure where each area in the grid would be assigned to some $\mathbf{z}_k$. It is not difficult to see that allowing a transformation on the features available at node k (i.e., Ω_k), can allow more elaborate encoder designs. We study how to design such a transformation before

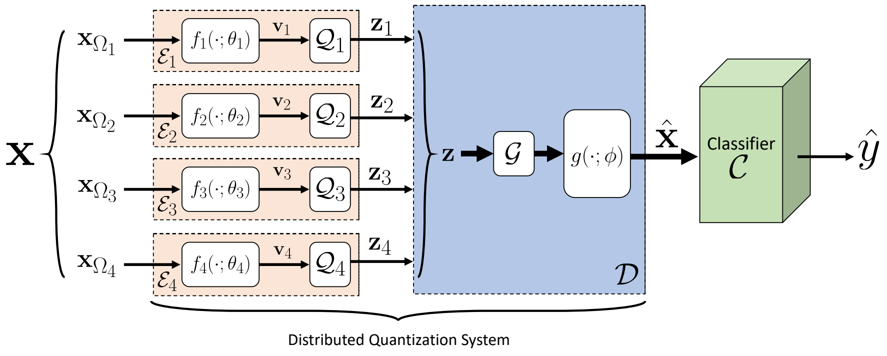


Figure 6.5: An example with $K = 4$ nodes showing the components of encoders and decoder.

applying `GBI` as part of the deep learning approach proposed in the following section.

6.7 Distributed Quantization for Classification Tasks Using Neural Representations

In this section, we explore a learning based approach for the distributed quantization problem introduced in Section 6.4. We consider a quantization system where the encoders $\{\mathcal{E}_k\}_{k=1}^K$ and decoder $\mathcal{D}$ are neural networks, followed by a pretrained classifier $\mathcal{C}$ that is subdifferentiable.

The structure of the encoders and decoder is shown in Fig. 6.5. In particular, the encoder $\mathcal{E}_k(\cdot)$ is decomposed into a neural network parameterized by θ_k, which implements a function $f_k(\cdot;\theta_k) : \mathcal{X}_{\Omega_k} \to \mathbb{R}^{m_k}$, followed by a quantizer $\mathcal{Q}_k : \mathbb{R}^{m_k} \to \mathcal{M}_k$ that maps the output of the neural network to a discrete set $\mathcal{M}_k \subseteq \mathbb{R}^{m_k}$ of size at most 2^{R_k}. That is,

$$\begin{aligned} \mathbf{v}_k &= f_k(\mathbf{x}_k;\theta_k) \in \mathbb{R}^{m_k},\ \forall k \in [1:K], \\ \mathbf{z}_k &= \mathcal{Q}_k(\mathbf{v}_k),\ \forall k \in [1:K]. \end{aligned} \tag{6.15}$$

Given $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K]$ as input, the decoder $\mathcal{D}$ first applies an initial mapping $\mathcal{G}$ that takes $\mathbf{z}$ to $\mathbb{R}^{\overline{m}}$, where $\overline{m} = \sum_{k=1}^K m_k$. This serves as a *combiner* for the values $\{\mathbf{z}_k\}_{k=1}^K$ received from the different encoders. Afterwards, a neural network $g(\cdot;\phi)$, parameterized by ϕ, is applied on the output of $\mathcal{G}$ (see Fig. 6.5) before feeding the output $\widehat{\mathbf{x}}$ to the classifier $\mathcal{C}$. Thus, we have

$$\mathcal{G} : \prod_{k=1}^{K} \mathcal{M}_k \to \mathbb{R}^{\overline{m}} \quad , \quad g(\cdot;\phi) : \mathbb{R}^{\overline{m}} \to \mathcal{X}^n$$
$$\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{z}) = g(\mathcal{G}(\mathbf{z});\phi). \tag{6.16}$$

Our objective is to minimize the misclassification loss

$$\min_{\theta_k, \mathcal{Q}_k, \mathcal{G}, \phi} \frac{1}{N} \sum_{i=1}^{N} [\mathbb{I}(\widehat{y}(\widehat{\mathbf{x}}^{(i)}) \neq y(\mathbf{x}^{(i)}))]. \tag{6.17}$$

Instead of minimizing the $0/1$ loss in (6.17), we construct a distribution from the output of the classifier $\mathcal{C}$ using a `softmax` layer, and then apply the cross entropy (CE) loss to find the maximum likelihood estimator of $y(\mathbf{x})$ [Bis06] using the dataset $\mathcal{T}$. Hence, our objective is to $\min_{\theta_k, \mathcal{Q}_k, \mathcal{G}, \phi} \{\mathcal{L}^c\}$, where

$$\mathcal{L}^c = \frac{1}{N} \sum_{i=1}^{N} -\log\left(\texttt{softmax}[\mathcal{C}(\widehat{\mathbf{x}}^{(i)})]_{y(\mathbf{x}^{(i)})}\right), \tag{6.18}$$

and $[\texttt{softmax}(\mathbf{u})]_j = \exp(u_j)/\sum_{i=1}^{|\mathcal{Y}|} \exp(u_i)$.

We next discuss a challenge in applying standard backpropagation techniques for training our neural networks. Since the classifier $\mathcal{C}$ is subdifferentiable, it is possible to compute the gradient of the (CE) loss in (6.18) with respect to the decoder parameters ϕ. However, regardless of how the quantizers $\mathcal{Q}_k$ are designed, the only subgradient of the quantizers is all zeros. As a result, it is not possible to apply backpropagation methods [OVK17, ROV19, STS19] to update the encoders parameters $\{\theta_k\}_{k=1}^K$. In the following two subsections, we introduce two different approaches for designing $\{\mathcal{Q}_k\}_{k=1}^K$ and the combiner $\mathcal{G}$, and discuss how to incorporate their design in the learning framework of the neural network parameters $\{\theta_k\}_{k=1}^K$ and ϕ.

Remark 11. Note that we do not optimize the classifier $\mathcal{C}$ as it is assumed to be pretrained and fixed. However, since the approaches discussed in this section are gradient-based, they can be directly extended to the case where the classifier $\mathcal{C}$ is trainable as well, i.e., we can update the parameters of the classifier $\mathcal{C}$ as we update the parameters of the networks f_k and g of the encoders and decoders.

In a preliminary work [EFD19], this has been shown to improve the performance of the trained system as the end-to-end system could be adapted for the special partitioning of features in the encoders.

6.7.1 Discrete Distributed Neural Representation for Classification:

In the first approach, we explicitly design the encoders to produce binary string representations of $\mathbf{z}_k$. In particular, for each encoder $\mathcal{E}_k$, the neural network $f_k(\cdot;\theta_k)$ outputs a vector with R_k entries ($m_k{=}R_k$), and we constrain the range of the elements of this vector to be in $[-1{,}1]$. We achieve this by selecting the activations of the last layer of the neural network $f_k(\cdot;\theta_k)$ to be a function that has the range $[-1{,}1]$ (we used the $\tanh(.)$ function in our numerical evaluation). We then simply quantize the output values, by applying the quantizer $\mathcal{Q}_k$ as

$$\mathcal{Q}_k(\mathbf{u}) = 2 * \mathbb{I}(\mathbf{u} \geq 0) - 1, \quad \forall k \in [1:K], \tag{6.19}$$

where the indicator function $\mathbb{I}$ is applied elementwise. For the combiner $\mathcal{G}$ in the decoder we simply use an identity function.

Remark 12. As discussed above the $\mathcal{Q}_k$ function prevents the backpropagation of the gradient to the encoder network $f_k(\cdot;\theta_k)$. To alleviate this, a straight-through approach is to only use the quantizer blocks in the forward pass and treat them as an identity during backpropagation [OVK17, STS19]. This approach works well in some applications [OVK17], however, we observed in our experiments that such an approach prevented the encoder parameters $\{\theta_k\}_{k=1}^K$ from having meaningful gradient updates, and the end-to-end system had a classification performance close to random guessing in the CIFAR10 dataset. In particular, this can happen as when applying the chain rule during backpropagation, we would like to have the derivative $\partial\mathcal{L}^c/\partial\mathbf{v}_k$ to update the parameters θ_k, where $\mathbf{v}_k = f_k(\mathbf{x}_{\Omega_k};\theta_k)$. Instead, the straight-through approach would use the gradient of a different point in space

$$\frac{\nabla\partial\mathcal{L}^c}{\partial\mathbf{z}_k} = \frac{\partial\mathcal{L}^c}{\partial(\mathbf{v}_k + (\mathbf{z}_k - \mathbf{v}_k))},$$

to update θ_k, where $\mathbf{z}_k = \mathcal{Q}_k(\mathbf{v}_k)$ (as in Fig. 6.5). This can be very different from the intended gradient depending on how $\mathcal{L}^c$ looks as a function of $\mathbf{z}_k$ and how big is the second term $(\mathbf{z}_k - \mathbf{v}_k)$. As an illustration, if one choice of θ_k results in $v_k = 10^{-6}$ (very close to 0), it would get quantized to $\mathbf{z}_k = 1$, resulting in quantization noise $\mathbf{z}_k - \mathbf{v}_k = 1 - 10^{-6}$; if a different θ_k results in $\mathbf{v}_k$ being very close to 1, it would again get quantized to 1, in this case with negligible quantization noise. Both parameters θ_k would be updated by the same gradient, even though in the first case, $\mathbf{v}_k$ was orders of magnitude smaller. Thus, when skipping the quantizer in the backpropagation, the calculated gradients may not be useful if the quantization noise is large.

Regularization for quantization: Based on the observation in Remark 12, we opted to facilitate gradient-based optimization by dropping the quantizers blocks $\{\mathcal{Q}_k\}_{k=1}^K$ during training (both in the forward and backward passes) and instead nudge the network to naturally output values close to quantized ones. In particular, we penalize the output values that are far from both -1 and $+1$, by introducing an additional term to the loss in (6.18), termed *quantization loss*, and calculated as

$$\mathcal{L}^{\mathrm{q}} = -\frac{1}{KN}\sum_{i=1}^{N}\sum_{k=1}^{K}\left\| f_k(\mathbf{x}_{\Omega_k}^{(i)};\theta_k)\right\|_2^2. \tag{6.20}$$

Note that since we choose the activations of the last layer in each encoder to have the range $[-1, 1]$, $\mathcal{L}^{\mathrm{q}}$ is minimized (achieves the optimal value $-\sum_k R_k/K$) only when $f_k(\mathbf{x}_{\Omega_k}^{(i)};\theta_k) \in \{-1, 1\}^{R_k}$ $\forall i \in [1 : N]$, $k \in [1 : K]$. Thus, the total training loss becomes

$$\mathcal{L} = \mathcal{L}^{\mathrm{c}} + \beta\mathcal{L}^{\mathrm{q}}, \tag{6.21}$$

where: (i) β is a hyperparameter that controls the contribution of $\mathcal{L}^{\mathrm{q}}$; and (ii) $\mathcal{L}^{\mathrm{c}}$ is the misclassification loss in (6.18).

For large enough β, minimizing $\mathcal{L}$ can be interpreted as minimizing the classification loss under the constraint that the encoders outputs are very close to $-1/1$, which results in $\|f_k(\mathbf{x}_{\Omega_k}^{(i)};\theta_k) - \mathcal{Q}_k(f_k(\mathbf{x}_{\Omega_k}^{(i)};\theta_k))\|_2$ being very small. That is, the outputs without quantization differ by only a small amount from the outputs with quantization which can be treated as negligible quantization noise during testing.

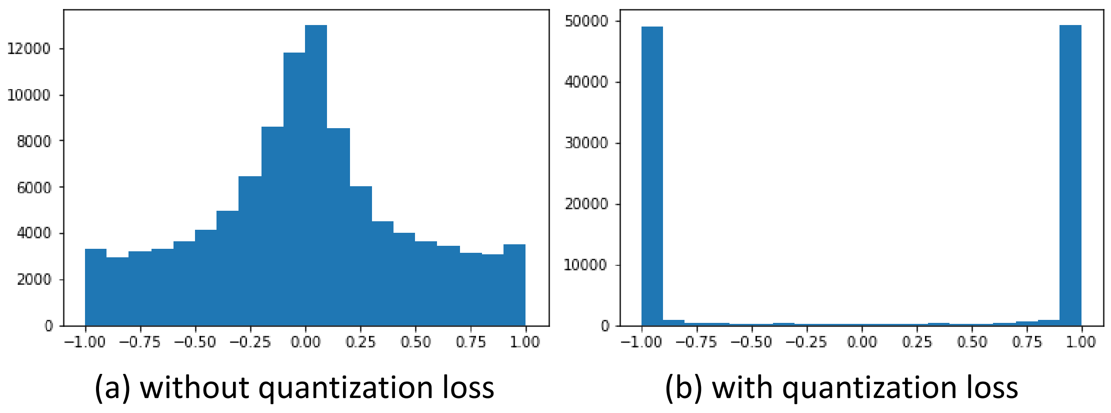


Figure 6.6: Distribution of the decoder inputs after training for 50 epochs on the CIFAR10 dataset.

To illustrate the impact of the *quantization loss* on the distribution of the encoder outputs, Fig. 6.6 shows the empirical distribution of the encoders outputs after 50 training epochs on the CIFAR-10 dataset, with and without using $\mathcal{L}^{\mathrm{q}}$. While the classification loss tries to direct the parameters of the encoders and decoder $\{\theta_k\}, \phi$ to improve the classification accuracy, the quantization loss adjusts the parameters to push the encoders outputs to be close to $-1/1$.

In the approach discussed in this subsection, we have integrated the quantization during the training phase by modifying the loss function to favor models that have small added noise due to quantization. Instead of modifying the objective function, in the following subsection, we introduce a multi-phase approach, where we first learn continuous representations for classification and then learn a quantizer on these continuous representations using our previously introduced GBI algorithm.

Remark 13. Note that the learning approach described in this subsection has computational complexity $O(N * \texttt{num\_epochs})$, where $\texttt{num\_epochs}$ is typically much smaller than N.

6.7.2 Distributed Neural Representation Using GBI:

The main idea in this approach is to use the GBI algorithm to design the quantizers $\{\mathcal{Q}_k\}_{k=1}^{K}$ and the initial decoder $\mathcal{G}$. We first use the neural network $f_k(\cdot;\theta_k)$ to map the features of node k from $\mathbf{x}_{\Omega_k} \in \mathbb{R}^{|\Omega_k|}$ to $\mathbf{v}_k \in \mathbb{R}^{m_k}$. We select m_k to be as small as possible while maintaining a good classification accuracy; that is, the networks $f_k(\cdot;\theta_k)$ essentially perform dimensionality reduction at the encoders before applying the quantization step. We then apply `GBI` on the output of the encoder neural network f_k. The main benefit is that, by decreasing the number of dimensions of the input to `GBI` algorithm, we alleviate the complexity of `GBI`, that grows with the number of dimensions (see Section 6.6).

Given that the neural network $f_k(\cdot;\theta_k)$ is potentially a universal function approximator [Csa01], it is not difficult to see that even using a naive uniform quantizer $\mathcal{Q}_k$, we could potentially implement any encoder $\mathcal{E}_k = \mathcal{Q}_k(f_k(\cdot;\theta_k)) : \mathbb{R}^{|\Omega_k|} \to \mathcal{M}_k$. Hence, ideally, the choice of the quantizer $\mathcal{Q}_k$ should not play a significant role. However, due to the fact that neural networks tend to work well only if the initialization is close to a good solution, the choice of the quantizer becomes important. In the following, we propose a method that operates in three phases:

Phase 1. We first train the encoders and decoder neural networks without any quantization units (i.e., without $\mathcal{Q}_k$ and $\mathcal{G}$) until we get classification accuracy that is close to the classifier's accuracy. Note that for $m_k \geq |\Omega_k|$, we can reconstruct the classifier accuracy. Effectively, in this step, we are following the example structure shown in Fig. 6.5, assuming the blocks $\mathcal{Q}_k$ and $\mathcal{G}$ are identities.

Phase 2. With the parameters θ_k, ϕ learned in **Phase 1**, we now design the quantization components $\{\mathcal{Q}_k\}_{k=1}^{K}$ and $\mathcal{G}$ based on the outputs $\{\mathbf{v}_k\}$ of the neural networks $f_k(\cdot;\theta_k)$. If the quantizer maps data points that have different labels to the same quantized value, the quantized value cannot be used to classify the points correctly. Hence, the objective of the quantizer is to map points that have different labels to different quantized values. We do this by introducing boundaries in the space using our `GBI` quantizer, described earlier in Section 6.6.

Phase 3. Finally, we continue training the encoders and decoder neural networks (f_k and g) with the quantizers designed in **Phase 2**. To do so we skip the quantizers blocks in the back propagation and consider them only in the forward pass. We observed in our experiments that this skip does not cause the network to behave randomly as the initialization is designed carefully. The parameters learned in **Phase 1** act as initializations for θ_k, ϕ in this phase. Phase 3 enables to fine tune the network parameters given that we have already learned the quantizer components earlier in **Phase 2**.

6.8 Experimental Evaluation

In this section, we present various experimental results, comparing the behaviour of our proposed distributed quantization approaches with quantization approaches for reconstruction. We find that tailoring the quantizers to the classification task can offer significant savings: we can achieve more than a factor of two reduction in terms of the number of bits communicated, for the same classification accuracy. Moreover, our algorithms retain reasonable classification performance even when constrained to use a very small number of bits per encoder; for instance, for 2 bits per encoder, we achieve approximately two to four times the classification accuracy of alternative approaches. Additionally, we also compare to centralized quantization approaches for classification and show that despite our distributed setup, we are still able to achieve a competitive performance in terms of classification accuracy.

Note that although in the upcoming experiments, we uniformly use R bits/sample at each encoder, the framework presented in Section 6.7 can be applied when the number of bits used by each encoder are not equal.

6.8.1 Performance on Electromyography Sensor Measurements:

We start with experiments on a dataset of surface electromyographic (sEMG) signals [LKK18]. Each data point represents measurements recorded from 8 sensors that are used to differentiate between 6 different hand gestures. For our classifier $\mathcal{C}$, we use a Multi-Layer Perceptron (MLP) [Bis06] architecture with fully connected layers of the form $8 - 100 - 200 - 200 - 200 - 6$ and ReLu activations. The classifier was pretrained on an unquantized training set of $15,345$ measurements, and yielded a baseline accuracy of 98.66% on a test set of $6,578$ measurements.

For our distributed quantization framework, we assume that we have $K = 4$ encoders, where each encoder has access to measurements from only two of the sensors (i.e., we have four feature groups each consisting of two features). We use MLPs for our encoders and decoder, while the quantizers are either trained using the quantization loss regularization **(NN-REG)** or with the GBI

	bits per encoder (R)				
	1	**2**	**3**	**4**	**5**
k**-means** [Mac67]	18.77%	44.18%	56.47%	66.71%	75.19%
NN-REG	54.50%	63.04%	82.90%	94.72%	97.73%
NN-GBI	55.49%	72.35%	91.12%	97.30%	98.21%

Table 6.2: Classification accuracy on the sEMG test set. Each system uses R bits per encoder, $K = 4$ encoders and a pretrained classifier with 98.66% accuracy.

algorithm **(NN-GBI)** as described in Section 6.7. The hidden layers structures of encoders/decoders, and the hyper parameters (learning rate and regularization weight β) are described in Appendix E.3.

Comparison with quantization for reconstruction. We demonstrate that our approaches achieve competitive classification results with smaller number of bits as compared to distributed approaches aiming at reconstructing the input. We compare against the k-means algorithm [Mac67] as a representative for unsupervised vector quantization algorithms. In particular, in the sEMG dataset, each k-means encoder maps a point in $\mathbb{R}^2$ to the nearest centroid point among 2^R choices. The decoder is treated as an identity in this case, that passes its input $\mathbf{z}$ vector to the classifier.

The results are shown in Table 6.2. We see that our approaches outperform the unsupervised distributed quantization. For example, using 4 bits for each encoder, we can achieve a classification accuracy of $> 95\%$ while the unsupervised approach achieves a performance of 66%.

Comparison with LVQ for classification. To benchmark the performance of our distributed quantization system for classification against centralized approaches, a natural candidate for comparison is the centralized Generalized Learning Vector Quantization approach (GLVQ) [SY96]. In this case, the output of the algorithm is a Voronoi tessellation in the space, where each centroid is now associated with a class. Thus, by mapping a point in space to its nearest centroid, a classification is also performed by picking the class associated to the selected centroid. We compare the performance of our distributed quantization approaches against the quantizer-classifier learned by the

LVQ3 centralized quantizer [SY96] with learning rate 10^{-4} for 200 epochs. Since our distributed quantization system with $K = 4$ encoders uses 2 bits per encoder, we allow LVQ3 to use 8 bits (i.e., 64 centroids) to keep the total number of bits across the nodes constant. Our **NN-GBI** approach, yielded 71.59% accuracy, while the centralized LVQ classifier gave an accuracy of 75.53%.

Although LVQ gives a better classification accuracy, the learned Voronoi boundaries are not decomposable to be applied on distributed nodes. In particular, in the described setting, when inspecting the values of the centroids learned by the LVQ algorithm, we found that although 2^8 centroids are used in $\mathbb{R}^8$, restricting the values of the centroids to any one dimension of the 8, gave 64 distinct values which would require each of the 4 encoders to at least use 8 bits to represent these quantized values. Recall that from Table 6.2, we are able to achieve much higher accuracies than 75%, when only 5 bits are used at each encoder.

6.8.2 Performance on CIFAR10 Images:

In this set of experiments, we evaluate the performance of our proposed algorithms on the CIFAR10 dataset, where each input $\mathbf{x}$ is a $32 \times 32 \times 3$ image. Each image in the CIFAR10 dataset is associated with one of 10 classes. We assume a distributed quantization system with $K = 4$ distributed encoders, that each have access to a quadrant of the image. For the classifier, we use a pretrained VGG-13 classifier [SZ15] with 94.27% accuracy on the CIFAR10 test dataset.

Comparison to VQ-VAE. The VQ-VAE [OVK17] framework is used to learn discrete neural representations of a dataset for reconstruction. We compare against this framework implemented both in a centralized and a distributed fashion. In particular, for the centralized VQ-VAE, a single encoder has access to the full image. We use the same VQ-VAE network structure from [OVK17] for the CIFAR10 dataset and ensure that the total number of bits used by the VQ-VAE encoder is 4 times what our system would use for a single encoder. In the distributed setting, a VQ-VAE encoder is applied on each image quadrant and then a common decoder is used for reconstruction. VQ-VAE structures were trained with 2×10^{-4} learning rate, 200 epochs and 64 batch size.

	bits per encoder (R)				
	1	**2**	**3**	**4**	**5**
VQ-VAE (C)	13.82%	14.36%	15.87%	17.52%	18.18%
VQ-VAE	10.12%	10.46%	11.03%	11.61%	12.28%
NN-REG	48.63%	63.32%	68.07%	73.43%	78.12%
NN-GBI	48.33%	60.88%	65.16%	71.57%	81.18%

Table 6.3: Classification accuracy on CIFAR10 test set. All distributed systems use R bits/encoder. The centralized VQ-VAE system [VQ-VAE(C)] uses $4R$ bits at the single encoder.

The results are summarized in Table 6.3. We find that although VQ-VAE has shown great success in reconstructing images from discrete representations, it does not perform well with a low number of bits even in the centralized case. To get classification accuracy of around 50%, the centralized VQ-VAE required 200 bits (equivalent to 50 bits/encoder in the distributed setting), while our algorithms could get more than 70% accuracy with 3 bits per encoder.

6.9 Conclusion and Open Research Directions

In this chapter, we introduced the problem of data-driven distributed data quantization for classification. We proved that in many cases, designing an optimal quantization system is an NP-hard problem that is also hard to approximate. For a case that is not NP-hard, we proposed an optimal polynomial-time algorithm for designing the quantizer under some structural restrictions. For the NP-hard cases, we proposed a polynomial time greedy approach and two learning based approaches. Numerical results on the sEMG and CIFAR10 datasets[6] indicate that tailoring the quantizers to the classification task can offer significant savings: more (and in some cases much more) than a factor of two reduction in the number of bits communicated, for the same classification accuracy.

[6] Follow up works validated our algorithms on other datasets [LHF23].

Moreover, our algorithms retain reasonable classification performance even when constrained to use a very small number of bits.

We leave the following as open direction:

- Developing an approach that can adapt to different levels of compression without the need to retrain the encoders/decoders.
- How to design the encoders/decoders architectures for multi-modal datasets, e.g., speech, text, and images?
- Adapting the schemes for other compression tasks, e.g., large model compression.

Acknowledgement

The work in this chapter has been done in collaboration with Yahya H. Ezzeldin, during his PhD at UCLA, and Tara Sadjadpour during her undergraduate studies at UCLA.

APPENDIX A

Appendix for Chapter 2: A Reduction from Contextual to Linear Bandits

A.1 Reduction When Context is a Product Set

Algorithm 12 Reduction from stochastic contexts to no context for product context sets

```
1: Input: an algorithm $\Lambda$ for linear contextual bandits with action set
       $$\mathcal{X} = \{a' \in \{0,1\}^{2d} | \, a'_{2i-1} + a'_{2i} = 1 \, \forall i \in [d]\} \tag{A.1}$$
2: for $t = 1 : T$ do
3:     Ask $\Lambda$ for an arm to play $a'_t \in \mathcal{X}$ given the history $r_1, ..., r_{t-1}$.
4:     Play arm $a_t$ with $(a_t)_i = \max\{a \in \mathcal{A}_t^{(i)}\}$ if $a'_{2i-1} = 1$ and $(a_t)_i = \min\{a \in \mathcal{A}_t^{(i)}\}$ otherwise.
       Receive reward $r_t$.
```

We here consider the special case where the action distribution $\mathcal{D}$ is unknown, but the action space has a coordinate-wise product structure, i.e.,

Assumption 3. *$\mathcal{A}_t = \prod_{i=1}^{d} \mathcal{A}_t^{(i)}$, where $\mathcal{A}_t^{(i)} \subset \mathbb{R}$.*

This is an important hard instance that appears in many lower bounds [LS20]. We will show that in this case, the d-dimensional stochastic contextual bandits can be reduced to a linear bandit problem with no misspecification, although the distribution $\mathcal{D}$ is unknown, but where now the parameter vector is over $2d$ dimensions.

The main idea of the reduction is that $g(\theta)$ can be factored into a coordinate-wise product between an unknown vector that only depends on the context distribution and a known vector that only depends on θ. The unknown vector can be incorporated with $\theta_\star$ reducing the contextual instance to a linear bandit instance but now with a different unknown parameter $\theta'_\star$. In particular, we can write $\langle g(\theta), \theta_\star\rangle = \langle a'(\theta), \theta'_\star\rangle$, where $a'(\theta)$ is a vector in $\mathbb{R}^{2d}$ that does not depend on the distribution, and equals

$$(a'(\theta))_{2i} = \begin{cases} 1 & \text{if } (\theta)_i < 0 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad (a'(\theta))_{2i-1} = 1 - (a'(\theta))_{2i} \quad \forall i = 1, \ldots, d, . \tag{A.2}$$

Thus, we can follow the same reduction algorithm as in Section 2.4.1, but where now we call a $2d$-dimensional linear bandit algorithm Λ and provide Λ with the fixed action set

$$\mathcal{X} = \{a' \in \{0,1\}^{2d} |\ a'_{2i-1} + a'_{2i} = 1\ \forall i \in [d]\} \tag{A.3}$$

as the pseudocode Algorithm 12 describes.

Theorem 21. *Let Λ be any algorithm for linear bandits, I be a contextual linear bandit instance with stochastic contexts that satisfy Assumption 3 with unknown parameter $\theta_\star$, and r_t be the rewards generated as described in Algorithm 12. It holds that*

- *The rewards r_t are generated, by pulling arm a'_t, from a linear bandit instance L with action set $\mathcal{X}$ in* (A.1)*, and unknown parameter $\theta'_\star \in \mathbb{R}^{2d}$ with $\|\theta'_\star\|_2 \le 2$.*
- *With probability at least $1 - \delta$ it holds that*

$$|R_T(I) - R_T^\Lambda(L)| \le c\sqrt{T \log(1/\delta)}, \tag{A.4}$$

where $R_T(I)$ is the regret of Algorithm 12 over the instance I and $R_T^\Lambda(L)$ is the regret of algorithm Λ over the instance L, and c is a universal constant.

Proof. The proof follows from Theorem 1 using the observation that

$$(g(\theta))_i = \begin{cases} \mathbb{E}_{\mathcal{A}\sim\mathcal{D}}[\max_{a\in\mathcal{A}}(a)_i] & \text{if } (\theta)_i > 0 \\ \mathbb{E}_{\mathcal{A}\sim\mathcal{D}}[\min_{a\in\mathcal{A}}(a)_i] & \text{if } (\theta)_i < 0 \end{cases} . \tag{A.5}$$

and thus, given (A.3), we can define $\theta'_\star \in \mathbb{R}^{2d}$ as

$$(\theta'_\star)_i = \begin{cases} \mathbb{E}[\max_{a\in\mathcal{A}}(a)_{(i+1)/2}](\theta_\star)_{(i+1)/2} & \text{if } i \text{ is odd} \\ \mathbb{E}[\min_{a\in\mathcal{A}}(a)_{\lfloor(i+1)/2\rfloor}](\theta_\star)_{\lfloor(i+1)/2\rfloor} & \text{if } i \text{ is even.} \end{cases} \tag{A.6}$$

By definition of $a'(\theta), \theta'_\star$, we have that $\langle g(\theta), \theta_\star\rangle = \langle a'(\theta), \theta'_\star\rangle$. To see that $\|\theta'_\star\| \le 2$, we observe that

$$\sum_{i=1}^{d} |(\theta'_\star)_{2i-1}| = \sum_{i=1}^{d} |\mathbb{E}[\max_{a\in\mathcal{A}}(a)_i](\theta_\star)_i| \le \langle |g(\mathbf{1}/\sqrt{d})|, |\theta_\star|\rangle \le 1, \tag{A.7}$$

where absolute value of a vector is defined as a vector with $(|\theta_\star|)_i = |(\theta_\star)_i|$. Similarly, $\sum_{i=1}^{d} |(\theta'_\star)_{2i}| \le 1$. Hence, $\|\theta'_\star\|_2 \le \|\theta'_\star\|_1 \le 2$.

As before, to construct the linear bandit instance, we need rewards that follow the stochastic linear bandits model. The result follows from Proposition 2 and the fact that $\langle g(\theta_t), \theta_\star\rangle = \langle a_t, \theta'_\star\rangle$.

□

A.2 Proof of Theorem 1: Reduction for Known Context Distribution

Theorem (Restatement of Theorem 1)**.** *Let Λ be any algorithm for linear bandits and I be a contextual linear bandit instance with stochastic contexts, unknown parameter $\theta_\star$ and rewards r_t generated as described in the reduction in Section 2.4.1. It holds that*

• The reward r_t is generated, by pulling the arm $g(\theta_t)$, from a linear bandit instance L with action set $\mathcal{X}$, and unknown parameter $\theta_\star$.

• The reduction results in an algorithm $\mathcal{M}$ for contextual linear bandits such that with probability at least $1-\delta$ we have

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| \le c\sqrt{T\log(1/\delta)}, \tag{A.8}$$

where $R_T^{\Lambda}(L)$ is the regret of Λ over the constructed linear bandit instance L, $R_T^{\mathcal{M}}(I)$ is the regret of algorithm $\mathcal{M}$ over the instance I and c is a universal constant.

Proof. Following the reduction described in the section, we start by showing that $R_T^{\Lambda}(L)$, the regret of the algorithm Λ on a linear bandit instance, is at most $\tilde{O}(\sqrt{T})$ away from $R_T^{\mathcal{M}}(I)$ with high probability. Recall that the regret $R_T^{\Lambda}(L)$ is defined as

$$R_T^{\Lambda}(L) = \sum_{t=1}^{T} \max_{\theta \in \Theta} \langle g(\theta), \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle. \tag{A.9}$$

We notice that the function g depends on the context distribution $\mathcal{D}$. In the following we assume for simplicity that for each $\theta \in \Theta$, there is a unique $a_t \in \mathcal{A}_t$ that satisfies $\langle a_t, \theta \rangle = \sup_{a \in \mathcal{A}_t} \langle a, \theta \rangle$ almost surely. We discuss how to remove this assumption at the end of the proof.

The regret $R_T^{\mathcal{M}}(I)$ of the contextual algorithm can be decomposed as

$$\begin{aligned}
R_T^{\mathcal{M}}(I) &= \sum_{t=1}^{T} \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle - \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle \\
&= \sum_{t=1}^{T} \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle - \langle \mathbb{E}[\arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle], \theta_\star \rangle \\
&\qquad + \langle \mathbb{E}[\arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle | \theta_t], \theta_\star \rangle - \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle \\
&\qquad + \langle \mathbb{E}[\arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle], \theta_\star \rangle - \langle \mathbb{E}[\arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle | \theta_t], \theta_\star \rangle \\
&= \sum_{t=1}^{T} \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle - \mathbb{E}[\langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_\star \rangle, \theta_\star \rangle] \\
&\qquad + \mathbb{E}[\langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle | \theta_t] - \langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta_t \rangle, \theta_\star \rangle \\
&\qquad + \langle g(\theta_\star), \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle,
\end{aligned} \tag{A.10}$$

where the expectation is with respect to the randomness in the context generation $\mathcal{A}_t$. In the following we will first show that $\langle g(\theta_\star), \theta_\star \rangle = \max_{\theta \in \Theta'} \langle g(\theta), \theta_\star \rangle$.

Indeed,we observe that $\forall \theta', \theta'' \in \Theta$ we have

$$\begin{aligned}
\max_{\theta \in \Theta} \langle g(\theta), \theta' \rangle \geq \langle g(\theta'), \theta' \rangle &= \mathbb{E}[\max_{a \in \mathcal{A}_t} \langle a, \theta' \rangle] \\
&\geq \mathbb{E}[\langle \arg\max_{a \in \mathcal{A}_t} \langle a, \theta'' \rangle, \theta' \rangle] = \langle g(\theta''), \theta' \rangle.
\end{aligned} \tag{A.11}$$

The above inequalities have to be met with equality since we can select $\theta'' = \arg\max_{\theta \in \Theta} \langle g(\theta), \theta' \rangle$ making the first and last terms equal. Hence, we have proved that

$$\langle g(\theta'), \theta' \rangle = \max_{\theta \in \Theta} \langle g(\theta), \theta' \rangle, \quad \forall\, \theta' \in \Theta. \tag{A.12}$$

Substituting in the last line of (A.10) using the triangle inequality, we get that

$$\begin{aligned}|R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| &\overset{(i)}{=} |\sum_{t=1}^{T} \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle] \\ &\quad + \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle | \theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle| \\ &\leq |\sum_{t=1}^{T} \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle]| \\ &\quad + |\sum_{t=1}^{T} \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle | \theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle|, \end{aligned} \tag{A.13}$$

where (i) follows by definition of $R_T^{\Lambda}(L)$ and (A.12).

We next bound the quantity

$$\Sigma_T := \sum_{t=1}^{T} \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle | \theta_t] - \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle. \tag{A.14}$$

Let $\mathcal{F}'_t = \sigma\{\theta_1, r_1, ..., \theta_t\}$ be the filtration of all historic information of the linear bandit problem up to time t. we notice that

$$\begin{aligned}\mathbb{E}[\Sigma_t|\mathcal{F}'_t] &= \mathbb{E}[\Sigma_{t-1}|\mathcal{F}'_t] + \mathbb{E}[\mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle | \theta_t]|\mathcal{F}'_t] - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_t\rangle, \theta_\star\rangle | \mathcal{F}'_t] \\ &= \Sigma_{t-1}. \end{aligned} \tag{A.15}$$

Hence, Σ_t is a martingale with a bounded difference (by boundedness of $\Theta, \mathcal{A}_t$). By Azuma–Hoeffding inequality [Wai19], we have that $|\Sigma_T| \leq c\sqrt{T \log 1/\delta}$ with probability at least $1 - \frac{\delta}{2}$. For completeness, we state a special case of the Azuma-Hoeffding inequality at the end of our proof. The summation $\Sigma'_T = \sum_{t=1}^{T} \langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle - \mathbb{E}[\langle \arg\max_{a\in\mathcal{A}_t} \langle a, \theta_\star\rangle, \theta_\star\rangle]$ can be bounded similarly. Hence, by (A.13), we have that with probability at least $1 - \delta$

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| \leq c\sqrt{T \log 1/\delta}. \tag{A.16}$$

We have shown that the regret of Algorithm 1 over the instance I is $O(\sqrt{T \log 1/\delta})$ away from $R_T^{\Lambda}(L)$ with probability at least $1 - \delta$. It remains to show that the rewards r_t generated by the described interaction with the instance I, are generated from L by the interaction of algorithm Λ.

Proposition 2. *The reward r_t can be rewritten as*

$$r_t = \langle g(\theta_t), \theta_\star \rangle + \eta'_t, \tag{A.17}$$

where $\mathbb{E}[\eta'_t|\mathcal{F}'_t] = 0, \mathbb{E}[\exp(\lambda\eta'_t)|\mathcal{F}'_t] \leq \exp(2\lambda^2) \quad \forall\lambda \in \mathbb{R}$, *and where* $\mathcal{F}'_t = \sigma\{\theta_1, r_1, ..., \theta_t\}$ *is the filtration of historic information up to time t.*

Proof. We have that

$$\begin{aligned} r_t &= \langle a_t, \theta_\star \rangle + \eta_t = \langle \arg\max_{b\in\mathcal{A}_t} \langle b, \theta_t \rangle, \theta_\star \rangle + \eta_t \\ &= \langle g(\theta_t), \theta_\star \rangle + \left(\eta_t + \langle \arg\max_{b\in\mathcal{A}_t} \langle b, \theta_t \rangle, \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle \right). \end{aligned} \tag{A.18}$$

We let $\eta'_t = \eta_t + \langle \arg\max_{b\in\mathcal{A}_t} \langle b, \theta_t \rangle, \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle$. The proof that $\mathbb{E}[\eta'_t|\mathcal{F}_t] = 0$ follows similarly to (A.15)

$$\begin{aligned} \mathbb{E}[\eta'_t|\mathcal{F}_t] &= \mathbb{E}[\eta_t|\mathcal{F}_t] + \mathbb{E}[\langle \arg\max_{b\in\mathcal{A}_t} \langle b, \theta_t \rangle, \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle|\mathcal{F}_t] \\ &= \mathbb{E}[\langle \arg\max_{b\in\mathcal{A}_t} \langle b, \theta_t \rangle, \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle|\theta_t] = 0. \end{aligned} \tag{A.19}$$

Lastly, $\mathbb{E}[\exp(\lambda\eta'_t)|\mathcal{F}_t] \leq \exp(2\lambda^2)\forall\lambda \in \mathbb{R}$ follows by boundedness of η'_t which follows by boundedness of $\eta_t, \Theta, \mathcal{A}_t$. □

This concludes the proof.

Lemma 13. [Azuma's Inequality [Wai19]] Let $\Sigma_0, \Sigma_1, ...$ be a martingale with respect to filtration $\mathcal{F}_0, \mathcal{F}_1, ...$ such that $|\Sigma_i - \Sigma_{i-1}| \leq c$ almost surely. Then for all $\epsilon > 0$, we have that

$$\mathbb{P}[|\Sigma_n - \Sigma_0| > \epsilon] \leq 2\exp(-\frac{\epsilon^2}{2nc^2}). \tag{A.20}$$

□

A.2.1 When $\sup_{a\in\mathcal{A}_t} \langle a, \theta \rangle$ is not Unique

One solution is to choose a_t as any action $\langle a_t, \theta \rangle \geq \sup_{a\in\mathcal{A}_t} \langle a, \theta \rangle - \delta$ for some $\delta > 0$. The error arising from performing this step can be controlled by δ, e.g., by setting $\delta = \exp(-T)$. Our proofs will follow by choosing a_t as described above, using any deterministic or random choice function, as long as the action a_t is measurable.

A.3 Proof of Theorem 2: Reduction for Unknown Context Distribution

Theorem (Restatement of Theorem 2)**.** *Let Λ_ϵ be an algorithm for linear bandits with ϵ misspecification and I be a contextual linear bandit instance with stochastic contexts, unknown parameter $\theta_\star$ and rewards r_t are generated as described in Algorithm 1. The following holds:*

• Conditioned on $\mathcal{H}_{t^{(m)}} = \sigma(\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_{t^{(m)}}, a_{t^{(m)}}, r_{t^{(m)}})$: in epoch m, the rewards r_t are generated, by pulling arm $g^{(m)}(\theta_t)$, from a misspecified linear bandit instance L_m for $t = t^{(m)} + 1, ..., t^{(m+1)}$, action set $\mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}$, unknown parameter $\theta_\star$, mean rewards $\langle g(\theta), \theta_\star\rangle$, and unknown misspecification ϵ'_m.

• The misspecification ϵ'_m is bounded by the known quantity ϵ_m with probability at least $1 - c\delta/M$.

• With probability at least $1-\delta$ we have that $|R_T(I) - \sum_{m=1}^{M} R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)| \le c\sqrt{T\log(1/\delta)}$, where $R_T(I)$ is the regret of Algorithm 1 over the instance I, $R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$ is the regret of algorithm Λ_{ϵ_m} over the bandit instance L_m in phase m, $T_m = t^{(m+1)} - t^{(m)}$, and c is a universal constant.

Proof. Let L_m be a bandit instance with actions $\mathcal{X}_m = \langle g^{(m)}(\theta) | \theta \in \Theta'\rangle$ indexed by the set Θ', mean rewards $\langle g(\theta), \theta_\star\rangle \forall\theta \in \Theta'$, and $t \in \{t^{(m)} + 1, ..., t^{(m+1)}\}$. Let

$$\mathcal{H}_{t^{(m)}} = \sigma\{\mathcal{A}_1, a_1, r_1, ..., \mathcal{A}_{t^{(m)}}, a_{t^{(m)}}, r_{t^{(m)}}\}$$

be the filtration of all historic information before epoch m. Note that $g^{(m)}$ is defined in line 7 of Algorithm 1 and is the empirical estimate of g using history $\mathcal{H}_{t^{(m)}}$, hence, $g^{(m)}$ is $\mathcal{H}_{t^{(m)}}$-predictable. Conditioned on $\mathcal{H}_{t^{(m)}}$, we have that L_m is a misspecified linear bandit instance with misspecification, $\epsilon'_m = \sup_{\theta\in\Theta'} \langle g(\theta) - g^{(m)}(\theta), \theta_\star\rangle$. And the regret of the algorithm Λ_{ϵ_m} over L_m is the random quantity

$$R_{T_m}^{\Lambda_{\epsilon_m}}(L_m) := \sum_{t=t^{(m)}+1}^{t^{(m+1)}} \max_{\theta\in\Theta'} \langle g(\theta), \theta_\star\rangle - \langle g(\theta_t), \theta_\star\rangle.$$

As we have shown in Proposition 2, conditioned on $\mathcal{H}_{t^{(m)}}$, r_t is generated from L_m by pulling arm $g^{(m)}(\theta_t)$. Define the event $\mathcal{E}_m = \{\epsilon'_m > \epsilon_m\}$ be the bad event that the random quantity ϵ'_m is greater than ϵ_m defined in Algorithm 1. We will show in Proposition 3 that $\sum_{m=1}^{M} \mathbb{P}[\mathcal{E}_m] \leq \delta$.

We next bound the regret of Algorithm 1 in terms of the random quantities $\{R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)\}_{m=1}^{M}$. By choosing δ sufficiently small, it it will be enough to bound $R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$ conditioned on $\mathcal{H}_{t^{(m)}}$ and the good event $\mathcal{G}_m = \{\epsilon'_m \leq \epsilon_m\}$. Let us define the random quantity $R_T^{\Lambda_\epsilon}(L_\epsilon) = \sum_{m=1}^{M} R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$. We show that $R_T^{\Lambda_\epsilon}(L_\epsilon)$ is at most $\tilde{O}(\sqrt{T})$ away from $R_T^{\mathcal{M}}$ with high probability. By definition of $R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$ we have that

$$R_T^{\Lambda_\epsilon}(L_\epsilon) = \sum_{t=1}^{T} \max_{\theta \in \Theta'} \langle g(\theta), \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle. \tag{A.21}$$

As in the proof of Theorem 1, we assume for simplicity that for each $\theta \in \Theta$, there is a unique $a_t \in \mathcal{A}_t$ that satisfies $\langle a_t, \theta \rangle = \sup_{a \in \mathcal{A}_t} \langle a, \theta \rangle$ almost surely. This can be relaxed in the same way as in Theorem 1.

Recall that L is the linear bandit instance in Theorem 1 with access to the function g, and actions in $\mathcal{X} = \{g(\theta) | \theta \in \Theta\}$. The regret $R_T^{\mathcal{M}}(I)$ of the contextual algorithm can be bounded as

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda_\epsilon}(L_\epsilon)| \leq |R_T^{\mathcal{M}}(I) - R_T^{\Lambda}(L)| + |R_T^{\Lambda}(L) - R_T^{\Lambda_\epsilon}(L_\epsilon)|, \tag{A.22}$$

where $R_T^{\Lambda}(L)$ is defined as follows

$$R_T^{\Lambda}(L) = \sum_{t=1}^{T} \max_{\theta \in \Theta} \langle g(\theta), \theta_\star \rangle - \langle g(\theta_t), \theta_\star \rangle,$$

and $\{\theta_t\}$ are the actions played by $\{\Lambda_{\epsilon_m}\}$. The first term in (A.22) is bounded in Theorem 1. In the following, we focus on bounding $|R_T^{\Lambda}(L) - R_T^{\Lambda_\epsilon}(L_\epsilon)|$. To that end, we first show that

$$|\langle g(\theta_\star), \theta_\star \rangle - \max_{\theta \in \Theta'} \langle g(\theta), \theta_\star \rangle| \leq 2/T. \tag{A.23}$$

We recall from (A.12) that

$$\langle g(\theta'), \theta' \rangle = \max_{\theta \in \Theta} \langle g(\theta), \theta' \rangle, \quad \forall\, \theta' \in \Theta. \tag{A.24}$$

From $1/T$-net properties, we also have that there exists $\phi \in \Theta'$ such that $\|\theta_\star - \phi\|_2 \leq 1/T$. Hence,

$$\begin{aligned}
\max_{\theta\in\Theta'} \langle g(\theta), \theta_\star\rangle &\overset{(i)}{\leq} \max_{\theta\in\Theta} \langle g(\theta), \theta_\star\rangle \overset{(ii)}{=} \langle g(\theta_\star), \theta_\star\rangle \\
&\overset{(iii)}{\leq} \langle g(\theta_\star), \phi\rangle + 1/T \\
&\leq \max_{\theta\in\Theta} \langle g(\theta), \phi\rangle + 1/T \\
&\overset{(iv)}{=} \langle g(\phi), \phi\rangle + 1/T \overset{(v)}{\leq} \langle g(\phi), \theta_\star\rangle + 2/T \\
&\overset{(vi)}{\leq} \max_{\theta\in\Theta'} \langle g(\theta), \theta_\star\rangle + 2/T, \qquad \text{(A.25)}
\end{aligned}$$

where (i) follows from $\Theta' \subseteq \Theta$, (ii) follows from (A.12), (iii) follows from $\langle g(\theta_\star), \theta_\star - \phi\rangle \leq \|g(\theta_\star)\|_2\|\theta_\star - \phi\|_2 \leq 1/T$, (iv) follows from (A.12), (v) follows as in (iii), and (vi) follows from $\phi \in \Theta'$. Eq. (A.23) follows. Note that in this part it is important to have $\Theta' \subseteq \Theta$.

As a result $|R_T^\Lambda(L) - R_T^{\Lambda_\epsilon}(L_\epsilon)|$ can be bounded as

$$\begin{aligned}
|R_T^\Lambda(L) - R_T^{\Lambda_\epsilon}(L_\epsilon)| &= |\sum_{t=1}^T \max_{\theta\in\Theta'} \langle g(\theta), \theta_\star\rangle - \max_{\theta\in\Theta} \langle g(\theta), \theta_\star\rangle| \\
&\leq \sum_{t=1}^T |\max_{\theta\in\Theta'} \langle g(\theta), \theta_\star\rangle - \max_{\theta\in\Theta} \langle g(\theta), \theta_\star\rangle| \overset{(i)}{\leq} 2, \qquad \text{(A.26)}
\end{aligned}$$

where (i) follows uses (A.23) and (A.12).

Hence, by (A.22), Theorem 1, (A.26) and union bound, we have that with probability at least $1 - \delta$

$$|R_T^{\mathcal{M}}(I) - R_T^{\Lambda_\epsilon}(L_\epsilon)| \leq c\sqrt{T \log 1/\delta}. \qquad \text{(A.27)}$$

Since $R_T^{\Lambda_\epsilon}(L_\epsilon) = \sum_{m=1}^M R_{T_m}^{\Lambda_{\epsilon_m}}(L_m)$, we have proved the second part of the theorem. It remains to show that in each epoch m the rewards r_t are generated from the for linear bandit instance L_m and to bound the amount of misspecification. By Proposition 2, it suffices to show the following.

Proposition 3. *For each $m \in [M]$, we have that with probability at least $1 - \delta/M$ it holds that*

$$|\langle g(\theta), \theta'\rangle - \langle g^{(m)}(\theta), \theta'\rangle| \leq 2\sqrt{\frac{\log(2M|\Theta'|/\delta)}{t^{(m)}}} \forall\theta \in \Theta', \theta' \in \Theta. \qquad \text{(A.28)}$$

Proof. Since for a fixed θ' we have that $\langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta\rangle, \theta'\rangle$ is $1/4$-subgaussian with mean $\langle g(\theta),\theta'\rangle$, we have that with probability at least $1-\frac{\delta}{M|\Theta'|^2}$ it holds that

$$\begin{aligned} |\langle g(\theta),\theta'\rangle - \langle g^{(m)}(\theta),\theta'\rangle| &= |\langle g(\theta),\theta'\rangle - \frac{1}{t^{(m+1)}}\sum_{t=1}^{t^{(m+1)}} \langle \arg\max_{a\in\mathcal{A}_t} \langle a,\theta\rangle,\theta'\rangle| \\ &\le 2\sqrt{\frac{\log(2M|\Theta'|/\delta)}{t^{(m)}}}. \end{aligned} \tag{A.29}$$

By the union bound, the following holds with probability at least $1-\delta/M$

$$|\langle g(\theta),\theta'\rangle - \langle g^{(m)}(\theta),\theta'\rangle| \le 2\sqrt{\frac{\log(2M|\Theta'|/\delta)}{t^{(m)}}}\forall\theta,\theta'\in\Theta'. \tag{A.30}$$

Let us pick arbitrary $\theta\in\Theta',\theta'\in\Theta$. We have that there is $\phi'\in\Theta'$ such that $\|\theta'-\phi'\|_2\le 1/T$. Hence, by Cauchy-Schwartz and the triangle inequality, we have that the following holds with probability at least $1-\delta/M$

$$\begin{aligned} |\langle g(\theta),\theta'\rangle - \langle g^{(m)}(\theta),\theta'\rangle| &\le |\langle g(\theta),\phi'\rangle - \langle g^{(m)}(\theta),\phi'\rangle| + \frac{2}{T} \\ &\le 2\sqrt{\frac{\log(2M|\Theta'|/\delta)}{t^{(m)}}} + \frac{2}{T}\forall\theta\in\Theta',\theta'\in\Theta. \end{aligned} \tag{A.31}$$

□

This concludes the proof. □

A.4 Proof of Lemma 1: Computational Complexity

Lemma. Consider a given $m\in[M]$ and let $g^{(m)}(\theta)=\frac{1}{t^{(m)}}\sum_{t=1}^{t^{(m)}}\arg\max_{a\in\mathcal{A}_t}\langle a,\theta\rangle$, $\mathcal{X}_m=\{g^{(m)}(\theta)|\theta\in\Theta'\}$, where $t^{(m)}$ is the length of phase m and $\Theta'=\{[\theta]_q|\theta\in\Theta\}$ is a discretization of Θ, $[\theta]_q=q\lfloor\theta\sqrt{d}/q\rfloor/\sqrt{d}$ and q is the discretization parameter. For any $\theta\in\{a\in\mathbb{R}^d|\|a\|_2\le 1\}$, $\epsilon\in\mathbb{R}^+$, if $q\le\epsilon/2$, we have that

$$\langle g^{(m)}([\theta/\|\theta\|_2]_q),\theta\rangle \ge \sup_{a\in\mathcal{X}_m}\langle a,\theta\rangle - \epsilon\|\theta\|_2. \tag{A.32}$$

Proof. We first observe that

$$0 \leq \theta - [\theta]_q = \theta - \frac{\lfloor \theta\sqrt{d}/q \rfloor}{\sqrt{d}/q} \leq q/\sqrt{d}\mathbf{1}. \tag{A.33}$$

It follows that $\|\theta - [\theta]_q\|_2 \leq q$. We notice that

$$\begin{aligned}\langle g^{(m)}(\theta), \theta\rangle = \frac{1}{t^{(m)}}\sum_{t=1}^{t^{(m)}} \langle \mathcal{O}(\mathcal{A}_t;\theta), \theta\rangle &\geq \frac{1}{t^{(m)}}\sum_{t=1}^{t^{(m)}} \langle \mathcal{O}(\mathcal{A}_t;\theta'), \theta\rangle \\ &= \langle g^{(m)}(\theta'), \theta\rangle, \quad \forall \theta' \in \Theta. \end{aligned}\tag{A.34}$$

Hence,

$$\langle g^{(m)}(\theta), \theta\rangle \geq \sup_{\theta'\in\Theta'} \langle g^{(m)}(\theta'), \theta\rangle. \tag{A.35}$$

We also have that

$$\begin{aligned}\langle g^{(m)}([\theta]_q), \theta\rangle &= \langle g^{(m)}([\theta]_q), [\theta]_q\rangle + \langle g^{(m)}([\theta]_q), \theta - [\theta]_q\rangle \\ &\geq \langle g^{(m)}([\theta]_q), [\theta]_q\rangle - \|g^{(m)}([\theta]_q)\|_2\|\theta - [\theta]_q\|_2 \\ &\geq \langle g^{(m)}([\theta]_q), [\theta]_q\rangle - q \\ &\overset{(i)}{\geq} \langle g^{(m)}(\theta), [\theta]_q\rangle - q \\ &\geq \langle g^{(m)}(\theta), \theta\rangle - \|g^{(m)}(\theta)\|_2\|\theta - [\theta]_q\|_2 - q \\ &\geq \langle g^{(m)}(\theta), \theta\rangle - 2q \overset{(ii)}{\geq} \sup_{\theta'\in\Theta'} \langle g^{(m)}(\theta'), \theta\rangle - 2q = \sup_{a\in\mathcal{X}_m} \langle a, \theta\rangle - 2q, \end{aligned}\tag{A.36}$$

where (i) and (ii) follow from (A.35). As (A.36) holds $\forall \theta \in \mathbb{R}^d$, we get that

$$\langle g^{(m)}([\theta/\|\theta\|_2]_q), \theta/\|\theta\|_2\rangle \geq \sup_{a\in\mathcal{X}_m} \langle a, \frac{\theta}{\|\theta\|_2}\rangle - 2q = \frac{\sup_{a\in\mathcal{X}_m}\langle a, \theta\rangle}{\|\theta\|_2} - 2q. \tag{A.37}$$

It follows that

$$\langle g^{(m)}([\theta/\|\theta\|_2]_q), \theta\rangle \geq \sup_{a\in\mathcal{X}_m} \langle a, \theta\rangle - 2q\|\theta\|_2. \tag{A.38}$$

□

APPENDIX B

Appendix for Chapter 3: A Computationally-Efficient Algorithm for Batched Linear Contextual Bandits

B.1 Table of Notations

The notations of Chapter 3 are collected in Table B.1.

Symbol	Description
T	time horizon
$\theta_\star$	unknown parameter vector in $\mathbb{R}^d$
$\mathcal{A}$	action set if fixed over time
d	dimension of actions and unknown parameter
μ_a	mean of arm a: $\langle a, \theta_\star \rangle$
$a^\star$	best action: $\arg\max_{a \in \mathcal{A}} \mu_a$
Δ_a	gap: $\mu_{a^\star} - \mu_a$
$\Delta_{\min}$	minimum gap: $\inf_{a \in \mathcal{A}: \Delta_a > 0} \Delta_a$
$\mathcal{O}(\mathcal{A}; .)$	linear optimization oracle for the set $\mathcal{A}$
$\mathcal{O}^+_\epsilon(\mathcal{A}; .)$	approximate linear optimization oracle with additive error ϵ for the set $\mathcal{A}$
$\mathcal{O}^\times_\alpha(\mathcal{A}; .)$	approximate linear optimization oracle with multiplicative error α for the set $\mathcal{A}$
$\mathcal{T}_{\text{opt}}$	time complexity of optimization oracle $\mathcal{O}(\mathcal{A}; .)$
$\mathcal{M}_{\text{opt}}$	space complexity of optimization oracle $\mathcal{O}(\mathcal{A}; .)$
a_t	pulled action at time t

R_T	regret: $\sum_{t=1}^{T} \Delta_{a_t}$
η_t	noise at time t
r_t	reward at time t
M	number of batches
T_m	length of batch m
H_m	set of time slots for batch m
λ	least squares regularization parameter
$\boldsymbol{V}_m$	least squares matrix at batch m: $\lambda \mathbf{I} + \sum_{t \in H_m} a_t a_t^\top$
θ_{m+1}	least squares estimate at end of batch m: $\boldsymbol{V}_m^{-1} \sum_{t \in H_m} r_t a_t$
$a_m^\star$	estimates best action at batch m: $\mathcal{O}^+_{1/T}(\mathcal{A}; \theta_m)$
$\Delta_m(a)$	estimated gap at batch m: $\langle a_m^\star - a, \theta_m \rangle$
C_L	approximate optimal design parameter: $e^8 d$
γ	$2d\sqrt{C_L(\log(1/\delta) + \log T)}$
$\phi_m(a)$	scaled action at batch m: $\frac{a}{1+\sqrt{T_{m-1}}\Delta_m(a)/(8\gamma)}$
$\mathcal{A}'$	extended action set: $\mathcal{A}' = \mathcal{A} \cup \mathcal{B}_{1/T}$
$\tilde{\mathcal{A}}_m$	weighted action set: $\{\phi_m(a) \mid a \in \mathcal{A}'\}$
δ	confidence parameter
$\mathcal{C}_m$	set of size d such that $\{\phi_m(a) \mid a \in \mathcal{C}_m\}$ is a barycentric spanner for $\tilde{\mathcal{A}}_m$
W	parameter: $3 \log T$
C	universal constant
$\mathcal{B}_r$	$\{a \in \mathbb{R}^d \mid \|a\|_2 \leq r\}$
$\mathcal{S}_r$	$\{a \in \mathbb{R}^d \mid \|a\|_2 = r\}$
$\|a\|_{\boldsymbol{V}}$	$\sqrt{a^\top \boldsymbol{V} a}$
$\mathbf{1}(E)$	indicator function: returns 1 if E holds and 0 otherwise
$[n], n \in \mathbb{N}$	$\{1, \cdots, n\}$

Table B.1: Table with notation for the linear bandit setting

B.2 Proof of Lemma 2: Approximate Inverse Gap Weighted Optimization

Lemma. Let $T \geq 3, \eta \in \mathbb{R}, \hat{a} \in \mathbb{R}^d, \hat{\theta} \in \mathcal{B}_T$ be given parameters, and $\mathcal{A}$ be a given set. Let $\Delta(a), \phi(a)$ denote $\Delta(a) = \langle \hat{a} - a, \hat{\theta} \rangle, \phi(a) = a/(1 + \eta\Delta(a))$. If $\mathcal{B}_{1/T} \subseteq \mathcal{A} \subseteq \mathcal{B}_1, |\eta| \leq T$ and $1/2 \leq 1 + \eta\Delta(a) \leq T^2,\ \forall a \in \mathcal{A}$, then for any $\theta \in \mathcal{S}_1$, LW-ArgMax (Algorithm 5) outputs an element $a \in \mathcal{A}$ such that

$$\langle \phi(a), \theta \rangle \geq \exp(-3) \sup_{b \in \{\phi(b') | b' \in \mathcal{A}\}} \langle b, \theta \rangle. \tag{B.1}$$

Proof. To simplify notations we define the modified gap as

$$\tilde{\Delta}(a) := 1 + \eta\Delta(a). \tag{B.2}$$

We also define the function $L_z : \mathcal{A} \to \mathbb{R}$ as

$$L_z(a) = (1 + 1/W)z\langle a, \theta \rangle - z^{1+1/W}\tilde{\Delta}(a), \tag{B.3}$$

where W is a parameter and we have set it to $3\log T$. The main part of the proof shows that the optimizer of L_z for some z is an optimizer of $\langle \phi(a), \theta \rangle (\langle a, \theta \rangle)^{1/W}$, which, as we also prove, is a good approximation of $\langle \phi(a), \theta \rangle$ for $W = 3\log T$. Towards that we first aim to show $\sup_{a \in \mathcal{A}} \langle \phi(a), \theta \rangle (\langle a, \theta \rangle)^{1/W} = (W \sup_{a \in \mathcal{A}, z \geq 0} L_z(a))^{1/W}$. The following boundedness properties will be repeatedly used in the proof

$$|\langle a, \theta \rangle| \leq \|a\|_2 \|\theta\|_2 \leq 1 \tag{B.4}$$

and by assumption we have that the modified gap can be bounded as

$$1/2 \leq \tilde{\Delta}(a) \leq T^2. \tag{B.5}$$

We start by proving the following property about the function L_z.

Claim 1.

$$\sup_{z \geq 0} L_z(a) = \begin{cases} \frac{1}{W}(\langle \phi(a), \theta \rangle)^W \langle a, \theta \rangle & \text{if } \langle a, \theta \rangle \geq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{B.6}$$

Proof. We notice the following fact about the function L_z. For any $a \in \mathcal{A}$ with $\langle a, \theta \rangle \geq 0$ we have that $L_z(a)$ is a concave function of z for $z \geq 0$, hence, by setting the derivative to 0, we observe that

$$\sup_{z \geq 0} L_z(a) = \frac{1}{W} \left(\frac{\langle a, \theta \rangle}{\tilde{\Delta}(a)} \right)^W \langle a, \theta \rangle, \tag{B.7}$$

where the supremum is attained by

$$z_a = \left(\frac{\langle a, \theta \rangle}{\tilde{\Delta}(a)} \right)^W . \tag{B.8}$$

We also notice that for any $a \in \mathcal{A}$ with $\langle a, \theta \rangle < 0$, since for all $a \in \mathcal{A}$, $\tilde{\Delta}(a) \geq 0$ we have that

$$\sup_{z \geq 0} L_z(a) = 0, \tag{B.9}$$

where the supremum is attained by $z_a = 0$. The result follows by combining (B.7) and (B.9). $\square$

The following fact follows from Claim 1.

Claim 2.

$$\sup_{a \in \mathcal{A}} \langle \phi(a), \theta \rangle \langle a, \theta \rangle^{1/W} = (W \sup_{a \in \mathcal{A}, z \geq 0} L_z(a))^{1/W}. \tag{B.10}$$

Proof. We notice that since $\|\theta/T\|_2 = 1/T$ since $\|\theta\|_2 = 1$ by assumption. Then $\theta/T \in \mathcal{B}_{1/T} \subseteq \mathcal{A}$. We also have that $\langle \theta/T, \theta \rangle = 1/T > 0$, hence, from Claim 1 we have

$$\sup_{a \in \mathcal{A} : \langle a, \theta \rangle \geq 0} \sup_{z \geq 0} L_z(a) \geq \sup_{z \geq 0} L_z(\theta/T) \overset{(i)}{=} \frac{1}{W} \left(\frac{\langle \theta/T, \theta \rangle}{\tilde{\Delta}(\theta/T)} \right)^W \langle \theta/T, \theta \rangle \overset{(ii)}{\geq} \frac{1}{W} \left(\frac{1/T}{T^2} \right)^W /T > 0, \tag{B.11}$$

where (i) follows from Claim 1 and (ii) uses $\tilde{\Delta}(a) \leq T^2$. It follows from (B.6) that

$$\begin{aligned} \sup_{a \in \mathcal{A}, z \geq 0} L_z(a) = \max \left\{ 0, \sup_{a \in \mathcal{A} : \langle a, \theta \rangle \geq 0} \sup_{z \geq 0} L_z(a) \right\} &= \sup_{a \in \mathcal{A} : \langle a, \theta \rangle \geq 0} \sup_{z \geq 0} L_z(a) \\ &= \sup_{a \in \mathcal{A} : \langle a, \theta \rangle \geq 0} \frac{1}{W} \left(\frac{\langle a, \theta \rangle}{\tilde{\Delta}(a)} \right)^W \langle a, \theta \rangle. \end{aligned} \tag{B.12}$$

Moreover we have that $\frac{1}{W} \left(\frac{\langle a, \theta \rangle}{\tilde{\Delta}(a)} \right)^W \langle a, \theta \rangle \leq 0$ whenever $\langle a, \theta \rangle \leq 0$. We can also see that

$$\sup_{a \in \mathcal{A} : \langle a, \theta \rangle \geq 0} \frac{1}{W} \left(\frac{\langle a, \theta \rangle}{\tilde{\Delta}(a)} \right)^W \langle a, \theta \rangle > 0 \tag{B.13}$$

by noticing that $\theta/T \in \mathcal{A}$ and has a positive objective value. Hence, we have that

$$\sup_{a\in\mathcal{A}, z\geq 0} L_z(a) = \sup_{a\in\mathcal{A}:\langle a,\theta\rangle\geq 0} \frac{1}{W}\left(\frac{\langle a,\theta\rangle}{\tilde{\Delta}(a)}\right)^W \langle a,\theta\rangle = \sup_{a\in\mathcal{A}} \frac{1}{W}\left(\frac{\langle a,\theta\rangle}{\tilde{\Delta}(a)}\right)^W \langle a,\theta\rangle. \tag{B.14}$$

It follows that

$$\sup_{a\in\mathcal{A}} \langle \phi(a),\theta\rangle\langle a,\theta\rangle^{1/W} = (W \sup_{a\in\mathcal{A},z\geq 0} L_z(a))^{1/W}. \tag{B.15}$$

□

In the following we assume that $\sup_{a\in\mathcal{A}} \langle \phi(a),\theta\rangle\langle a,\theta\rangle^{1/W}$ is attained by some $b^\star \in \mathcal{A}$ and also that $\sup_{a\in\mathcal{A}} \langle \phi(a),\theta\rangle$ is attained by some $a^\star \in \mathcal{A}$. The proofs can be extended to the case where the supremums are not attained by using sufficiently small approximation.

The proof continues as following

- We show that the algorithm uses z_i that is close to $z_{b^\star} = (\langle \phi(b^\star),\theta\rangle)^W$ (the optimizer of $L_z(b^\star)$ in (B.8)) in some iteration i.
- We show that the solution of $\sup_{a\in\mathcal{A}} L_{z_i}(a)$, namely $\tilde{a}$ satisfying $L_{z_i}(\tilde{a}) = \sup_{a\in\mathcal{A}} L_{z_i}(a)$, is an approximate optimizer of the function $\langle \phi(a),\theta\rangle\langle a,\theta\rangle^{1/W}$.
- We finally show that an approximate optimizer of $\langle \phi(a),\theta\rangle\langle a,\theta\rangle^{1/W}$ is also an approximate optimizer of $\langle \phi(a),\theta\rangle$.

Towards the first step, we start by finding an upper and lower bound on $z_{b^\star} = (\langle \phi(b^\star),\theta\rangle)^W$. From (B.4) and (B.5), we have that

$$\begin{aligned} 2^W \geq z_{b^\star} = (\langle \phi(b^\star),\theta\rangle)^W &\overset{(i)}{\geq} (\langle \phi(b^\star),\theta\rangle\langle b^\star,\theta\rangle^{1/W})^W \\ &\overset{(ii)}{\geq} (\langle \phi(\theta/T),\theta\rangle\langle \theta/T,\theta\rangle^{1/W})^W \\ &= \frac{1/T^{1+W}}{\tilde{\Delta}(\theta/T)} \geq \frac{1}{T^{3+W}}, \end{aligned} \tag{B.16}$$

where (i) follows from $|\langle b^\star,\theta\rangle| \leq 1$ and $\langle \phi(b^\star),\theta\rangle > 0$ (see (B.13)), and (ii) follows by definition of $b^\star$ as the maximizer of $\langle \phi(a),\theta\rangle\langle a,\theta\rangle^{1/W}$ over the set $\mathcal{A}$ and the fact that $\theta/T \in \mathcal{B}_{1/T} \subseteq \mathcal{A}$.

Then, we find an upper and lower bound on the values of z used by the algorithm. Recall that Algorithm 5 starts with $z = 2^W$ where $W = 3\log T$ and decreases z with a factor of $s = 1 - \frac{1}{6\log T}$ for $N = 36W\log^2 T$ iterations. We have that

$$\begin{aligned} 2^W s^N &= 2^W(1 - \frac{1}{6\log T})^N \leq \exp(W - N/(6\log T)) = \exp(W(1 - 6\log T)) \\ &\leq \exp(-3W\log(T)) = \frac{1}{T^{3W}} \leq \frac{1}{T^{3+W}} \end{aligned} \tag{B.17}$$

From (B.16), (B.17), the fact that Algorithm 5 starts with $z = 2^W$ and decreases z by a factor of $s = 1 - \frac{1}{6\log T}$ each step, it follows that there is an iteration i with

$$sz_{b^\star} \leq z_i \leq z_{b^\star}, \tag{B.18}$$

where z_i is the value of the variable z in iteration i. Now we consider the function L_{z_i}. We aim to show that an approximate optimizer of L_{z_i} is an approximate optimizer of $\langle\phi(a), \theta\rangle\langle a, \theta\rangle^{1/W}$. This is proved in the following lemma.

Lemma 14. Consider given $\eta \in \mathbb{R}, \theta \in \mathbb{R}^d, \hat{\theta} \in \mathbb{R}^d, \hat{a} \in \mathbb{R}^d$ and let $W = 3\log T, \Delta(a) = \langle \hat{a} - a, \hat{\theta}\rangle, \tilde{\Delta}(a) = 1 + \eta\Delta(a)$, $\phi(a) = a/\tilde{\Delta}(a)$, $L_z(a) = (1 + 1/W)z\langle a, \theta\rangle - z^{1+1/W}\tilde{\Delta}(a)$.
Let i be an iteration of Algorithm 5 with $sz_{b^\star} \leq z_i \leq z_{b^\star}$. If $\mathcal{B}_{1/T} \subseteq \mathcal{A} \subseteq \mathcal{B}_1$ and $1/2 \leq \tilde{\Delta}(a) \leq T^2\ \forall a \in \mathcal{A}$, then we have that

$$\langle\phi(a_i), \theta\rangle\langle a_i, \theta\rangle^{1/W} \geq \exp(-1)\sup_{b\in\mathcal{A}} \langle\phi(b), \theta\rangle\langle b, \theta\rangle^{1/W}, \tag{B.19}$$

where a_i is the approximate optimizer defined in step 7 of Algorithm 5 at iteration i.

Proof. To utilize Claim 1 to relate the optimizer of L_{z_i} to the optimizer of $\langle\phi(a), \theta\rangle\langle a, \theta\rangle^{1/W}$, we first show that $\sup_{a\in\mathcal{A}} L_{z_i}(a) > 0$. We have that (recall that $b^\star$ is the optimizer of $\langle\phi(a), \theta\rangle\langle a, \theta\rangle^{1/W}$)

$$\begin{aligned} L_{z_i}(b^\star) &= (1 + 1/W)z_i\langle b^\star, \theta\rangle - z_i^{1+1/W}\tilde{\Delta}(b^\star) \\ &\overset{(i)}{\geq} (1 + 1/W)sz_{b^\star}\langle b^\star, \theta\rangle - z_{b^\star}^{1+1/W}\tilde{\Delta}(b^\star) \\ &\overset{(ii)}{=} (\langle\phi(b^\star), \theta\rangle\langle b^\star, \theta\rangle^{1/W})^W((1 + 1/W)s - 1), \end{aligned} \tag{B.20}$$

where (i) follows from $\langle b^\star, \theta\rangle > 0$ (see (B.13)) and $\tilde{\Delta}(b^\star) \geq 0$, and (ii) follows by substituting $z_{b^\star} = (\langle b^\star, \theta\rangle / \tilde{\Delta}(b^\star))^W$. We denote

$$\beta := (1 + 1/W)s - 1. \tag{B.21}$$

We next lower bound β as follows (recall that $T \geq 3$ and $s = 1 - 1/6 \log T$)

$$(W\beta)^{1/W} = (1/2 - 1/(6\log T))^{1/(3\log T)} \geq (1/4)^{1/(3\log T)} \geq \exp(-0.5/\log T) \geq \exp(-0.5). \tag{B.22}$$

It follows that

$$\beta \geq \exp(-0.5W)/(W) \geq 1/(3T^2), \tag{B.23}$$

where the last inequality uses $T \geq 3$, hence, $\log T \leq \sqrt{T}$. Substituting in (B.20) we get that

$$\sup_{b\in\mathcal{A}} L_{z_i}(b) \geq L_{z_i}(b^\star) \geq \frac{(\langle \phi(b^\star), \theta\rangle \langle b^\star, \theta\rangle^{1/W})^W}{3T^2} \geq \frac{1}{3T^{2+12\log T}}, \tag{B.24}$$

where the last inequality follows by definition of $b^\star$ as the maximizer of $\langle \phi(a), \theta\rangle \langle a, \theta\rangle^{1/W}$ over the set $\mathcal{A}$ and the fact that $\theta/T \in \mathcal{B}_{1/T} \subseteq \mathcal{A}$. In the algorithm, we do not construct an optimizer for L_{z_i}; instead we use an approximate optimizer a_i of the linear function given in step 7 of Algorithm 5. In the following we will use (B.24) to show that $L_{z_i}(a_i) > 0$. We notice that

$$\begin{aligned} L_{z_i}(b) &= (1 + 1/W)z_i\langle b, \theta\rangle - z_i^{1+1/W}(1 + \eta\langle \hat{a} - b, \hat{\theta}\rangle) \\ &= \langle b, (1 + 1/W)z_i\theta + z_i^{1+1/W}\eta\hat{\theta}\rangle - z_i^{1+1/W}(1 + \eta\langle \hat{a}, \hat{\theta}\rangle) \\ &= \langle b, \tilde{\theta}_i\rangle - z_i^{1+1/W}(1 + \eta\langle \hat{a}, \hat{\theta}\rangle), \end{aligned} \tag{B.25}$$

where $\tilde{\theta}_i = (1+1/W)z_i\theta + z_i^{1+1/W}\eta\hat{\theta}$. It follows that $\sup_{b\in\mathcal{A}} L_{z_i}(b) = (\sup_{b\in\mathcal{A}} \langle b, \tilde{\theta}_i\rangle) - z_i^{1+1/W}(1 + \eta\langle \hat{a}, \hat{\theta}\rangle)$. Hence, by definition of a_i in Algorithm 5 we get that

$$\begin{aligned} L_{z_i}(a_i) &\geq \sup_{b\in\mathcal{A}} L_{z_i}(b) - \frac{1 - \exp(-1)}{12T^{7+12\log T}}\|\tilde{\theta}_i\|_2 \\ &\overset{(i)}{\geq} \sup_{b\in\mathcal{A}} L_{z_i}(b) - \frac{1 - \exp(-1)}{3T^{2+12\log T}} \\ &\overset{(ii)}{\geq} \sup_{b\in\mathcal{A}} L_{z_i}(b) - (1 - \exp(-1)) \sup_{b\in\mathcal{A}} L_{z_i}(b) \end{aligned}$$

$$= \exp(-1) \sup_{b \in \mathcal{A}} L_{z_i}(b), \tag{B.26}$$

where (i) follows from $\|\tilde{\theta}_i\|_2 \le (1 + 1/W) z_i \|\theta\|_2 + |\eta| z_i^{1+1/W} \|\hat{\theta}\|_2 \le 2T^2 2^{W+1} \le 4T^5$ (recall that $|\eta| \le T, \hat{\theta} \in \mathcal{B}_T$ and $z_i \le 2^W$, $W = 3 \log T$) and (ii) follows from (B.24). It follows from (B.24) that $L_{z_i}(a_i) > 0$. Hence, from (B.6) we get that $\langle a_i, \theta \rangle \ge 0$. From (B.6) again it follows that

$$\begin{aligned}
\frac{1}{W} \left(\frac{\langle a_i, \theta \rangle}{\tilde{\Delta}(a_i)} \right)^W \langle a_i, \theta \rangle &= \sup_{z \ge 0} L_z(a_i) \ge L_{z_i}(a_i) \\
&\overset{(i)}{\ge} \exp(-1) \sup_{b \in \mathcal{A}} L_{z_i}(b) \\
&\ge \exp(-1) L_{z_i}(b^\star) \overset{(ii)}{\ge} \exp(-1) (\langle \phi(b^\star), \theta \rangle \langle b^\star, \theta \rangle^{1/W})^W \beta
\end{aligned} \tag{B.27}$$

where (i) follows from (B.26) and (ii) follows from (B.20). Hence, we have that

$$\begin{aligned}
\langle \phi(a_i), \theta \rangle \langle a_i, \theta \rangle^{1/W} &\ge \exp(-1/W) \langle \phi(b^\star), \theta \rangle \langle b^\star, \theta \rangle^{1/W} (W\beta)^{1/W} \\
&\ge \exp(-0.5) \langle \phi(b^\star), \theta \rangle \langle b^\star, \theta \rangle^{1/W} (W\beta)^{1/W} \\
&\overset{(i)}{\ge} \exp(-1) \langle \phi(b^\star), \theta \rangle \langle b^\star, \theta \rangle^{1/W},
\end{aligned} \tag{B.28}$$

where (i) follows from (B.22). □

The last part of the proof shows that an approximate optimizer for $\langle \phi(a), \theta \rangle \langle a, \theta \rangle^{1/W}$ is also an approximate optimizer for $\langle \phi(a), \theta \rangle$. We lower bound $\langle \phi(a_i), \theta \rangle$ as follows (recall that $a^\star$ is the optimizer of $\langle \phi(a), \theta \rangle$)

$$\begin{aligned}
\frac{\langle \phi(a_i), \theta \rangle}{\langle \phi(a^\star), \theta \rangle} &= \frac{\langle \phi(a_i), \theta \rangle \langle a_i, \theta \rangle^{1/W}}{\langle \phi(a^\star), \theta \rangle \langle a^\star, \theta \rangle^{1/W}} (\frac{\langle a^\star, \theta \rangle}{\langle a_i, \theta \rangle})^{1/W} \\
&\overset{(i)}{\ge} \exp(-1) \frac{\langle \phi(b^\star), \theta \rangle \langle b^\star, \theta \rangle^{1/W}}{\langle \phi(a^\star), \theta \rangle \langle a^\star, \theta \rangle^{1/W}} (\frac{\langle a^\star, \theta \rangle}{\langle a_i, \theta \rangle})^{1/W} \\
&\overset{(ii)}{\ge} \exp(-1) (\frac{\langle a^\star, \theta \rangle}{\langle a_i, \theta \rangle})^{1/W} \\
&\overset{(iii)}{\ge} \exp(-1) \langle a^\star, \theta \rangle^{1/W} = \exp(-1) \langle \phi(a^\star), \theta \rangle^{1/W} \tilde{\Delta}(a^\star)^{1/W} \\
&\overset{(iv)}{\ge} \exp(-1) \langle \phi(a^\star), \theta \rangle^{1/W} 0.5^{1/W}
\end{aligned}$$

$$
\begin{aligned}
&\geq \exp(-1.5)\langle \phi(a^\star), \theta\rangle^{1/W} \\
&\overset{(v)}{\geq} \exp(-1.5)\langle \phi(\theta/T), \theta\rangle^{1/W} = \exp(-1.5)(\frac{1/T}{\tilde{\Delta}(\theta/T)})^{1/W} \\
&\overset{(vi)}{\geq} \exp(-1.5)\left(\frac{1/T}{T^2}\right)^{1/W} = \exp(-1.5 - 3\log T/W) = \exp(-2.5). \qquad \text{(B.29)}
\end{aligned}
$$

where (i) follows from Lemma 14, (ii) follows by definition of $b^\star$ as the maximizer of $\langle \phi(a), \theta\rangle$ $\langle a, \theta\rangle^{1/W}$, (iii) follows from $\langle a^\star, \theta\rangle > 0, \langle a_i, \theta\rangle > 0$ and $|\langle a_i, \theta\rangle| \leq 1$, (iv) follows from (B.5), (v) uses the fact that $\theta/T \in \mathcal{A}$ and definition of $a^\star$ to attain the supremum of $\langle \phi(a), \theta\rangle$, and (vi) follows from (B.5). The proof is concluded by noticing that a_i is one of the candidates in the return statement of Algorithm 5, hence, if a is the output of Algorithm 5, then $\langle \phi(a), \theta\rangle \geq \langle \phi(a_i), \theta\rangle \geq \exp(-3)\langle \phi(a^\star), \theta\rangle$, where the last inequality follows from (B.29). □

B.3 Proof of Lemma 3: Barycentric Spanner

We here prove that Algorithm 2 can efficiently find a barycentric spanner.

Lemma 2. *Let $\eta \in \mathbb{R}, \hat{a} \in \mathbb{R}^d, \hat{\theta} \in \mathbb{R}^d$ be given parameters, and $\mathcal{A}$ be a given set. Let $\Delta(a), \phi(a)$ denote $\Delta(a) = \langle \hat{a} - a, \hat{\theta}\rangle, \phi(a) = a/(1 + \eta\Delta(a))$. Suppose that for any $\theta \in \mathcal{S}_1$, LW-ArgMax (Algorithm 5) with inputs $\mathcal{A}, \theta, \eta, \hat{a}, \hat{\theta}$, outputs $a_\theta \in \mathcal{A}$ with $\langle \phi(a_\theta), \theta\rangle \geq \alpha \sup_{a\in\mathcal{A}} \langle \phi(a), \theta\rangle$, then Algorithm 4 computes a C/α-approximate barycentric spanner for the set $\tilde{\mathcal{A}} = \{\phi(a) | a \in \mathcal{A}\}$ with at most $O(d^2 \log_C(d/\alpha))$ calls to LW-ArgMax.*

Proof. The proof is a simple modification of the proof of Proposition 2.5 in [AK08]; the difference is that we replace exact linear optimization oracles with approximate ones, and show that the resulting vectors still have the good properties we want.

We note that Lemma 2 holds for any generic action set $\mathcal{A}$ used to call Algorithm 2; however, since Algorithm 1 calls Algorithm 2 (and Algorithm 3) with input action set $\mathcal{A}'$, for consistency we will use $\mathcal{A}'$ as the input action set in the following, e.g., we interpret the lemma statement assumption as:

$$\langle \phi(\text{LW-ArgMax}(\theta)), \theta \rangle \geq \alpha \sup_{a \in \mathcal{A}'} \langle \phi(a), \theta \rangle.$$

From this asumption and the fact that $\max_{a \in \tilde{\mathcal{A}}} |\langle a, \theta \rangle| = \max\{\max_{a \in \tilde{\mathcal{A}}} \langle a, \theta \rangle, \max_{a \in \tilde{\mathcal{A}}} \langle a, -\theta \rangle\}$, we have that step 7 (and similarly step 12) in Algorithm 4 outputs a with

$$|\langle \phi(a), \theta \rangle| \geq \alpha \sup_{\tilde{a} \in \tilde{\mathcal{A}}} |\langle \tilde{a}, \theta \rangle|, \text{ for some } 0 < \alpha < 1. \tag{B.30}$$

We next show that if Algorithm 4 terminates then $\{\phi(a_1), \cdots, \phi(a_d)\}$ is a C/α-approximate barycentric spanner. We have that if there exists $a' \in \mathcal{A}'$ with $|\text{det}((\phi(a'), \boldsymbol{A}_{-i}))| \geq C/\alpha |\text{det}(\boldsymbol{A})|$ for some i, then from (B.30), in step 12 we have an a with $|\text{det}((\phi(a), \boldsymbol{A}_{-i}))| \geq C|\text{det}(\boldsymbol{A})|$, hence, the algorithm will continue. As a result when Algorithm 4 terminates we have that

$$\sup_{a \in \tilde{\boldsymbol{A}}} |\text{det}((a, \boldsymbol{A}_{-i}))| \leq C/\alpha |\text{det}(\boldsymbol{A})|, \quad \forall i \in [d]. \tag{B.31}$$

In the proof of Lemma 2 we showed that $\sup_{a \in \tilde{\mathcal{A}}} \langle a, \theta \rangle > 0, \ \forall \theta \neq 0$. This shows that at every step of Algorithm 4, the matrix $\boldsymbol{A}$ has non-zero determinent. Hence, $\{a_1, \cdots, a_d\}$ span $\mathbb{R}^d$. As a result for any $\tilde{a} \in \tilde{\mathcal{A}}$ we have that $\tilde{a} = \sum_{i=1}^{d} w_i a_i$ for some $\{w_i\}_{i=1}^{d}$. We have that

$$|\text{det}(\tilde{a}, \boldsymbol{A}_{-i})| = |\text{det}(\sum_{i=1}^{d} w_i a_i, \boldsymbol{A}_{-i})| = |w_i||\text{det}(\boldsymbol{A})|. \tag{B.32}$$

Hence, from (B.31) we get that

$$|w_i| \leq C/\alpha. \tag{B.33}$$

This implies that $\{\phi(a_1), \cdots, \phi(a_d)\}$ is a C/α-approximate barycentric spanner for $\tilde{\mathcal{A}}$. It remains to show that Algorithm 4 terminates in $O(d^2 \log_C d)$ iterations. The number of iterations of the first for loop is d. To bound the number of iterations of the second for loop, we notice that for each repetition of the for loop (which takes at most d iterations), det($\boldsymbol{A}$) increases by a factor of C. Let $\boldsymbol{M}_i = [\tilde{a}_1, \cdots, \tilde{a}_i, e_{i+1}, \cdots, e_d]$ be the value of the matrix $\boldsymbol{A}$ at the end of the i-th iteration of the first for loop. As the determinant of $\boldsymbol{A}$ increases by at least factor of C each repetition, then if N is the number of repetitions of the second for loop, we have that $C^N \leq |\text{det}(\boldsymbol{A})/\text{det}(\boldsymbol{M}_d)|$, where $\boldsymbol{A}$ is the matrix at the end of the N-th repetition of the second for loop. Hence, to prove the theorem it suffices to show that $|\det(\boldsymbol{A})/\det(\boldsymbol{M}_d)| \leq (1/\alpha)^d d^{d/2}$. Let $u_i^T = e_i^\top \boldsymbol{M}_i^{-1}$ and define $\boldsymbol{U}$ to be the matrix whose i-th row is u_i. We observe that

$$\langle u_i, a\rangle = \frac{\det(a, \boldsymbol{M'}_{-i})}{\det(\boldsymbol{M}_i)}, \quad \forall a \in \tilde{\mathcal{A}}, \tag{B.34}$$

by noticing that both sides are linear functions of a and equality holds for all columns of $\boldsymbol{M}_i$ which form a basis for $\mathbb{R}^d$. It follows from (B.30) that $|u_i^\top a| \leq 1/\alpha$. As each entry of $\boldsymbol{U}\boldsymbol{A}$ is $u_i^\top a$ for some $i \in [d], a \in \tilde{\mathcal{A}}$, all the entries of $\boldsymbol{U}\boldsymbol{A}$ lie in $[-1/\alpha, 1/\alpha]$. Hence, $\det(\boldsymbol{U}\boldsymbol{A}) \leq (1/\alpha)^d d^{d/2}$ as the determinant of a matrix is upper bounded by the product of the L^2-norms of its columns. We also notice that if $\boldsymbol{M}_d = [\tilde{a}_1, \cdots, \tilde{a}_d]$, then by definition of u_i we have $\langle u_i, \tilde{a}_j\rangle$ is zero if $j < i$, and $\langle u_i, \tilde{a}_i\rangle = 1,\ \forall i \in [d]$. Hence, $\boldsymbol{U}\boldsymbol{M}_d$ is upper triangular matrix with unit diagonal, implying $\det(\boldsymbol{U}\boldsymbol{M}_d) = 1$. We have that

$$\frac{\det(\boldsymbol{A})}{\det(\boldsymbol{M}_d)} = \frac{\det(\boldsymbol{U}\boldsymbol{A})}{\det(\boldsymbol{U}\boldsymbol{M}_d)} \leq (1/\alpha)^d d^{d/2}. \tag{B.35}$$

This conlcudes the proof. □

B.4 Proof of Theorem 8: Regret Analysis for Linear Bandits

Theorem 1. *Consider a linear bandit instance with action set $\mathcal{A} \subseteq \mathbb{R}^d$ and horizon T. There exists a universal constant C and a choice for the batch lengths such that Algorithm 3 finishes in at most $M = \lceil \log\log T\rceil + 1$ batches with regret bounded as*

$$R_T \leq C\gamma\sqrt{T}\log\log T \text{ with probability at least } 1-\delta, \tag{B.36}$$

where $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$, $C_L = e^8 d$ and δ is a parameter. Moreover, if for any $a \in \mathcal{A}$ with $\Delta_a > 0$ we have $\Delta_a \geq \Delta_{\min}$, then there exists a choice of batch lengths so that Algorithm 3 finishes in at most $M = \log_4 T$ batches with regret bounded as

$$R_T \leq C\frac{\gamma^2}{\Delta_{\min}}\log T \text{ with probability at least } 1-\delta. \tag{B.37}$$

Proof. Note that in Algorithm 3, we end batch m if the total number of pulls reaches T_m. Hence, it is not guaranteed that the number of pulls for arm a_i in batch m reaches $n_m(i)$, which complicates the analysis of the concentration for the least squares estimate parameters. To handle this, we first analyze a variant of Algorithm 3 that completes all $n_m(i)$ pulls for each action a_i, $i \in [d]$. We bound the regret of the variant algorithm when a good event $\tilde{G}$ (that we define later) holds, and show that $\mathbb{P}[\tilde{G}] \geq 1 - \delta$. Then, we show that conditioned on $\tilde{G}$, it holds that $\sum_{i=1}^{d} n_m(i) \leq T_m$, for all batches $m \in [M]$ (see (B.68)), which implies that Algorithm 3 coincides with the variant algorithm on $\tilde{G}$ in this case. In the following, we refer to the variant algorithm as Algorithm 3 for simplicity.

To invoke Lemma 2, and hence, Lemma 3, we first verify that the conditions of Lemma 2 hold for all batches m. We note that as a result of using the definition of $a_m^\star = \mathcal{O}_{1/T}^{+}(\mathcal{A}; \theta_m)$, due to the use of an approximate oracle and doing the maximization only over $\mathcal{A}$ (not the bigger set $\mathcal{A}'$), the value of $\Delta_m(a)$ can be negative, however, by definition of $\Delta_m = \langle a_m^\star - a_i, \theta_m \rangle$ and the fact that $\mathcal{A}' = \mathcal{A} \cup \mathcal{B}_{1/T}$, we have that

$$\Delta_m(a) \geq -1/T, \quad \forall a \in \mathcal{A}'. \tag{B.38}$$

Hence, we have that

$$\begin{aligned} 1/2 \overset{(i)}{\leq} 1 - \eta_m/T &\overset{(ii)}{\leq} 1 + \eta_m \Delta_m(a) \\ &\overset{(iii)}{\leq} 1 + 2\eta_m T \overset{(iv)}{\leq} T^2 \end{aligned} \tag{B.39}$$

where (i) follows from $\eta_m = \sqrt{T_{m-1}}/(8\gamma) \leq \sqrt{T}/(8\gamma)$, (ii) follows from (B.38), (iii) follows from $|\theta_m| = |\boldsymbol{V}_{m-1}^{-1} \sum_{t=1}^{T_{m-1}} \tilde{a}_t r_t| \leq \sum_{t=1}^{T_{m-1}} \|\tilde{a}_t r_t\|_2 \leq T$ since $\boldsymbol{V}_m \geq \boldsymbol{I}$, $|r_t| \leq 1, \|\tilde{a}_t\|_2 \leq 1$ (recall that $\boldsymbol{V}_m = \boldsymbol{I} + \sum_{i=0}^{d} n_{m-1}(i) a_i a_i^\top \mathbf{1}[a_i \notin \mathcal{B}_{1/T}]$, $\tilde{a}_t$ is the pulled action at the t-th iteration of the previous batch, $n_{m-1}(i)$ is the number of pulls for action a_i in the previous batch, $\{a_i\}_{i=1}^{d}$ is the set of actions for the approximate optimal design from previous batch), and (iv) uses $\eta_m \leq \sqrt{T}/(8\gamma)$. This shows that Lemma 2 applies to all calls to Algorithm 5, hence, Lemma 3; namely in each batch $m \geq 2$, Algorithm 4 finds an $\exp(4)$ ($C = \exp(1), \alpha = \exp(-3)$) barycentric spanner of the set $\{\phi_m(a) | a \in \mathcal{A}'\}$. For the first batch, we note that Algorithm 3 and Algorithm 4 do not use the same action gaps. Algorithm 4 uses $\theta_1 = 0$ and thus uses $\Delta(a) = 0$ and $\phi(a) = a$. Hence, it finds $\mathcal{C}_1$, an $\exp(4)$-barycentric spanner of $\mathcal{A}'$. Algorithm 3 sets $\Delta_1(a) = 1, \ \forall a \in \mathcal{A}'$, and thus $\tilde{\mathcal{A}}_1 = \{ca | a \in \mathcal{A}'\}$, $c = 1/(1 + 1/(8\gamma))$ is a scaled version of $\mathcal{A}'$. Hence, $\{\phi_1(a) = ca | a \in \mathcal{C}_1\}$ is a barycentric spanner for $\tilde{\mathcal{A}}_1$ as well. Thus we conclude that for $m = 1$ as well as all other $m \in [M]$, $\{\phi_m(a) | a \in \mathcal{C}_m\}$ is a barycentric spanner for $\tilde{\mathcal{A}}_m$.

We next prove the following lemma that shows the concentration of the estimates $\langle \phi_m(a), \theta_{m+1} \rangle$, $\forall a \in \mathcal{A}'$.

Lemma 15. Let $T \geq 2$, and θ_{m+1} be the regularized least squares estimate of $\theta_\star$ at the end of batch m in Algorithm 3. Let the event $\mathcal{G}$ be the event

$$\mathcal{G} : |\langle \phi_m(a), \theta_{m+1} - \theta_\star \rangle| \leq \gamma / \sqrt{T_m}, \quad \forall a \in \mathcal{A}', m \in [M], \tag{B.40}$$

where $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$. Then, we have that $\mathbb{P}[\mathcal{G}] \geq 1 - \delta$.

Proof. We note that the regularized least squares matrix $\boldsymbol{V}_{m+1}$ at the end of batch m can be bounded as (recall that the considered variant of Algorithm 3 finishes all $n_m(i)$ pulls $\forall i \in [d]$ and $\forall m \in [M]$)

$$\begin{aligned}
\boldsymbol{V}_{m+1} &\geq \lambda \boldsymbol{I} + \sum_{i=1}^{d} \left\lceil \frac{\pi(i) T_m / 8}{(1 + \sqrt{T_{m-1}} \Delta_m(a_i)/(8\gamma))^2} \right\rceil a_i a_i^\top \mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\
&\geq \lambda \boldsymbol{I} + \sum_{i=1}^{d} \frac{\pi(i) T_m / 8}{(1 + \sqrt{T_{m-1}} \Delta_m(a_i)/(8\gamma))^2} a_i a_i^\top \mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\
&= \lambda \boldsymbol{I} + \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \phi_m(a_i) \phi_m(a_i)^\top \mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\
&= \lambda \boldsymbol{I} + \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \phi_m(a_i) \phi_m(a_i)^\top - \boldsymbol{E},
\end{aligned} \tag{B.41}$$

where $\boldsymbol{E} = \sum_{i=1}^{d} \pi(i) T_m \phi_m(a_i) \phi_m(a_i)^\top \mathbf{1}[a_i \in \mathcal{B}_{1/T}]$. Hence, using (B.38), for any $a \in \mathcal{B}_{1/T}, T \geq 2$ we have that

$$\|\phi_m(a)\|_2 = \frac{\|a\|_2}{1 + \Delta_m(a)} \leq \frac{1/T}{1 - 1/T} \leq 2/T. \tag{B.42}$$

As a result we have that for any $T \geq 2, a \in \mathbb{R}^d$ with $\|a\|_2 \leq 1$

$$\begin{aligned}
a^\top \boldsymbol{E} a = \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} a^\top \phi_m(a_i) \phi_m(a_i)^\top a \mathbf{1}[a_i \in \mathcal{B}_{1/T}] &\leq \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \|a\|_2^2 \|\phi_m(a_i)\|_2^2 \mathbf{1}[a_i \in \mathcal{B}_{1/T}] \\
&\leq \sum_{i=1}^{d} \pi(i)/T \leq 1/T.
\end{aligned} \tag{B.43}$$

From (B.43) and (B.41) we get that for $T \geq 2$

$$\begin{aligned}\boldsymbol{V}_{m+1} &\geq \lambda \boldsymbol{I} + \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \phi_m(a_i)\phi_m(a_i)^\top - \boldsymbol{E} \geq \lambda \boldsymbol{I} + \sum_{i=1}^{d} \pi(i) \frac{T_m}{8} \phi_m(a_i)\phi_m(a_i)^\top - 1/T\boldsymbol{I} \\ &\overset{(i)}{=} (1 - 1/T)\boldsymbol{I} + \frac{T_m}{8}\boldsymbol{V}_{\pi,m} \geq \frac{T_m}{8}\boldsymbol{V}_{\pi,m}, \end{aligned} \tag{B.44}$$

where (i) follows from $\lambda = 1$ and

$$\boldsymbol{V}_{\pi,m} = \sum_{i=1}^{d} \pi(i)\phi_m(a_i)\phi_m(a_i)^\top. \tag{B.45}$$

By Cauchy-Schwartz inequality we have that

$$\begin{aligned}|\langle \phi_m(a), \theta_{m+1} - \theta_\star \rangle| &\leq \|\phi_m(a)\|_{\boldsymbol{V}_{m+1}^{-1}} \|\theta_{m+1} - \theta_\star\|_{\boldsymbol{V}_{m+1}} \overset{(i)}{\leq} \frac{\|\phi_m(a)\|_{\boldsymbol{V}_{\pi,m}^{-1}}}{\sqrt{T_m/8}} \|\theta_{m+1} - \theta_\star\|_{\boldsymbol{V}_{m+1}} \\ &\overset{(ii)}{\leq} 2\sqrt{2C_L d/T_m} \|\theta_{m+1} - \theta_\star\|_{\boldsymbol{V}_{m+1}}, \end{aligned} \tag{B.46}$$

where (i) follows from (B.44), and (ii) follows from the fact that $\{\phi_m(a_i), \pi(i)\}_{i=1}^d$ is a C_L-approximate design for $\tilde{\mathcal{A}}$. By Theorem 20.5 in [LS20], we have that with probability at least $1 - \delta$ it holds that

$$\|\theta_{m+1} - \theta_\star\|_{\boldsymbol{V}_{m+1}} \leq 2\sqrt{\log(1/\delta) + d\log T}, \quad \forall m \in [M]. \tag{B.47}$$

Combining with (B.46) we get that the next inequality holds with probability at least $1 - \delta$

$$|\langle \phi_m(a), \theta_{m+1} - \theta_\star \rangle| \leq 4d\sqrt{2C_L(\log(1/\delta) + \log T)/T_m}, \quad \forall m \in [M]. \tag{B.48}$$

□

Corollary 7 follows from Lemma 15 and the fact that $1 + \eta_m \Delta_m(a) \geq 1 - 1/\sqrt{T} > 0$ for $T > 1$.

Corollary 7. *Let $T \geq 2$, and θ_{m+1} be the regularized least squares estimate of $\theta_\star$ at the end of batch m in Algorithm 3. The following event holds with probability at least $1 - \delta$*

$$\mathcal{G}' : |\langle a, \theta_{m+1} \rangle - \mu_a| \leq \frac{\gamma}{\sqrt{T_m}} + \frac{\Delta_m(a)}{8}\sqrt{\frac{T_{m-1}}{T_m}}, \quad \forall a \in \mathcal{A}', m \in [M], \tag{B.49}$$

where $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$ and $\mu_a = \langle a, \theta_* \rangle$.

We introduce the definition of the gap Δ_a on the set $\mathcal{A}'$ as follows

$$\Delta_a = \sup_{b\in\mathcal{A}} \langle b, \theta_\star\rangle - \langle a, \theta_\star\rangle, \quad \forall a \in \mathcal{A}'. \tag{B.50}$$

We note that with this definition Δ_a may be negative for some $a \in \mathcal{A}'$ as the supremum is taken over the smaller set $\mathcal{A}$. However, we have that $\forall a \in \mathcal{A}'$

$$\Delta_a \geq \min\{0, \sup_{b\in\mathcal{A}} \langle b, \theta_\star\rangle - \sup_{u\in\mathcal{B}_{1/T}} \langle u, \theta_\star\rangle\} \geq -1/T. \tag{B.51}$$

We also have that

$$\Delta_a \leq \max\{1, \sup_{b\in\mathcal{A}} \langle b, \theta_\star\rangle - \inf_{u\in\mathcal{B}_{1/T}} \langle u, \theta_\star\rangle\} \leq 1 + 1/T. \tag{B.52}$$

We can now prove the following lemma about the concentration of $\Delta_m(a)$.

Lemma 16. Suppose that $\mathcal{G}'$ holds and assume $T_m \geq T_{m-1},\ \forall m \in [M]$, then we have that the following events hold

$$\tilde{\mathcal{G}}_m : -4\frac{\gamma}{\sqrt{T_{m-1}}} + \frac{1}{2}\Delta_a \leq \Delta_m(a) \leq 2\Delta_a + 4\frac{\gamma}{\sqrt{T_{m-1}}}, \quad \forall a \in \mathcal{A}', \quad \forall m \in M. \tag{B.53}$$

Proof. We prove the statement by induction on m. For $m = 1$ we have that for any $a \in \mathcal{A}'$

$$\begin{aligned} -4\frac{\gamma}{\sqrt{T_{m-1}}} + \frac{1}{2}\Delta_a \overset{(i)}{=} -4\gamma + \frac{1}{2}\Delta_a \leq \frac{1}{2}\Delta_a &\overset{(ii)}{\leq} \frac{1}{2}(1 + 1/T) \\ &\overset{(iii)}{\leq} \Delta_1(a) \\ &\overset{(iv)}{\leq} 4\gamma - 2/T \overset{(v)}{\leq} 4\gamma + 2\Delta_a = 4\frac{\gamma}{\sqrt{T_{m-1}}} + 2\Delta_a \end{aligned} \tag{B.54}$$

where (i) uses $T_0 = 1$, (ii) follows from (B.52), (iii) follows from $\Delta_1(a) = 1$, (iv) uses $\gamma \geq 1$, and (v) follows from (B.51). Now suppose that $\tilde{\mathcal{G}}_m$ holds. We need to show that $\tilde{\mathcal{G}}_{m+1}$ holds. We have that for any $a \in \mathcal{A}'$

$$\begin{aligned} \Delta_{m+1}(a) &= \langle a^\star_{m+1} - a, \theta_{m+1}\rangle \\ &\overset{(i)}{\leq} \mu_{a^\star_{m+1}} - \mu_a + 2\frac{\gamma}{\sqrt{T_m}} + \left(\frac{\Delta_m(a^\star_{m+1})}{8} + \frac{\Delta_m(a)}{8}\right)\sqrt{\frac{T_{m-1}}{T_m}} \end{aligned}$$

$$
\begin{aligned}
&= \Delta_a - \Delta_{a^\star_{m+1}} + 2\frac{\gamma}{\sqrt{T_m}} + (\frac{\Delta_m(a^\star_{m+1})}{8} + \frac{\Delta_m(a)}{8})\sqrt{\frac{T_{m-1}}{T_m}} \\
&\overset{(ii)}{\le} \Delta_a - \Delta_{a^\star_{m+1}} + 2\frac{\gamma}{\sqrt{T_m}} + (\frac{2\Delta_{a^\star_{m+1}} + 4\frac{\gamma}{\sqrt{T_{m-1}}}}{8} + \frac{2\Delta_a + 4\frac{\gamma}{\sqrt{T_{m-1}}}}{8})\sqrt{\frac{T_{m-1}}{T_m}} \\
&= \Delta_a - \Delta_{a^\star_{m+1}} + 3\frac{\gamma}{\sqrt{T_m}} + (\frac{\Delta_{a^\star_{m+1}}}{4} + \frac{\Delta_a}{4})\sqrt{\frac{T_{m-1}}{T_m}} \\
&= 2\Delta_a + 3\frac{\gamma}{\sqrt{T_m}} + \Delta_a(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1) + \Delta_{a^\star_{m+1}}(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1), \qquad \text{(B.55)}
\end{aligned}
$$

where (i) follows from $\mathcal{G}'$, and (ii) follows by the induction hypothesis. We have that if $\Delta_a \ge 0$, then

$$\Delta_a(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1) \overset{(i)}{\le} \Delta_a(1/4 - 1) \le 0, \qquad \text{(B.56)}$$

where (i) uses the fact that $T_m \ge T_{m-1}$. If $\Delta_a < 0$, then

$$\Delta_a(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1) \le -\Delta_a \overset{(i)}{\le} 1/T, \qquad \text{(B.57)}$$

where (i) follows from (B.51). Hence, from (B.56) and (B.57) we get that

$$\Delta_a(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1) \le 1/T. \qquad \text{(B.58)}$$

Similarly, we have

$$\Delta_{a^\star_{m+1}}(1/4\sqrt{\frac{T_{m-1}}{T_m}} - 1) \le 1/T. \qquad \text{(B.59)}$$

Substituting from (B.58) and (B.59) in (B.55) we get that

$$\Delta_{m+1}(a) \le 2\Delta_a + 3\frac{\gamma}{\sqrt{T_m}} + 2/T \le 2\Delta_a + 4\frac{\gamma}{\sqrt{T_m}}, \qquad \text{(B.60)}$$

where the last inequality uses $T_m \le T$ and $\gamma \ge 2$. We next prove a lower bound on $\Delta_{m+1}(a)$. In the following we assume that $\sup_{a\in\mathcal{A}} \mu_a$ is attained by $a^\star \in \mathcal{A}$, and $\sup_{a\in\mathcal{A}'} \langle a, \theta_{m+1}\rangle$ is attained by $\tilde{a}^\star_{m+1} \in \mathcal{A}'$. The proof can be easily extended when the supremums are not attained by using a small approximation and taking the limit. We have that for any $a \in \mathcal{A}'$

$$
\begin{aligned}
\Delta_{m+1}(a) &= \langle a^\star_{m+1} - a, \theta_{m+1}\rangle \ge \langle \tilde{a}^\star_{m+1} - a, \theta_{m+1}\rangle - 1/T \ge \langle a^\star - a, \theta_{m+1}\rangle - 1/T \\
&\overset{(i)}{\ge} \mu_{a^\star} - \mu_a - 2\frac{\gamma}{\sqrt{T_m}} - (\frac{\Delta_m(a^\star)}{8} + \frac{\Delta_m(a)}{8})\sqrt{\frac{T_{m-1}}{T_m}} - 1/T
\end{aligned}
$$

$$= \Delta_a - 2\frac{\gamma}{\sqrt{T_m}} - (\frac{\Delta_m(a^\star)}{8} + \frac{\Delta_m(a)}{8})\sqrt{\frac{T_{m-1}}{T_m}} - 1/T$$
$$\overset{(ii)}{\geq} \Delta_a - 2\frac{\gamma}{\sqrt{T_m}} - (\frac{2\Delta_{a^\star} + 4\frac{\gamma}{\sqrt{T_{m-1}}}}{8} + \frac{2\Delta_a + 4\frac{\gamma}{\sqrt{T_{m-1}}}}{8})\sqrt{\frac{T_{m-1}}{T_m}} - 1/T$$
$$= \Delta_a - 3\frac{\gamma}{\sqrt{T_m}} - \frac{\Delta_a}{4}\sqrt{\frac{T_{m-1}}{T_m}} - 1/T = \frac{1}{2}\Delta_a - 3\frac{\gamma}{\sqrt{T_m}} + \Delta_a(\frac{1}{2} - \frac{1}{4}\sqrt{\frac{T_{m-1}}{T_m}}) - 1/T. \tag{B.61}$$

where (i) follows from $\mathcal{G}'$, and (ii) follows by the induction hypothesis. We have that if $\Delta_a \geq 0$, then

$$\Delta_a(\frac{1}{2} - \frac{1}{4}\sqrt{\frac{T_{m-1}}{T_m}}) \overset{(i)}{\geq} \frac{1}{4}\Delta_a \geq 0, \tag{B.62}$$

where (i) follows from $T_m \geq T_{m-1}$. If $\Delta_a \leq 0$, then

$$\Delta_a(\frac{1}{2} - \frac{1}{4}\sqrt{\frac{T_{m-1}}{T_m}}) \geq \frac{1}{2}\Delta_a \geq -\frac{1}{2}1/T. \tag{B.63}$$

Substituting from (B.62) and (B.63) in (B.61) we get that

$$\Delta_{m+1}(a) \geq \frac{1}{2}\Delta_a - 3\frac{\gamma}{\sqrt{T_m}} - 2/T \geq \frac{1}{2}\Delta_a - 4\frac{\gamma}{\sqrt{T_m}}, \tag{B.64}$$

where the last inequality uses $T_m \leq T$ and $\gamma \geq 2$. Combining (B.60) and (B.64) we get that $\tilde{\mathcal{G}}_{m+1}$ holds. We conclude by induction that $\tilde{\mathcal{G}}_m$ holds for all $m \in [M]$. □

We are now ready to prove the regret bound. We first upper bound the regret in batch m

$$R^{(m)} = \sum_{t \in H_m} \sup_{a \in \mathcal{A}} \mu_a - \mu_{a_t}, \tag{B.65}$$

where H_m is the set of time slots for batch m, and a_t is the action pulled at time t. The following lemma gives a bound on $R^{(m)}$.

Lemma 17. Suppose that $\tilde{\mathcal{G}}_m$ holds, then we have that

$$R^{(m)} \leq d + 1 + \frac{68\gamma T_m}{\sqrt{T_{m-1}}}. \tag{B.66}$$

Moreover, if $\forall a \in \mathcal{A}$ with $\Delta_a > 0$ we have $\Delta_a \geq \Delta_{\min}$ then

$$R^{(m)} \leq d + 1 + \frac{544\gamma^2 T_m}{\Delta_{\min} T_{m-1}}. \tag{B.67}$$

If $T_1 \geq 2d$ then

$$\sum_{i=1}^{d} n_m(i) \leq T_m. \tag{B.68}$$

Proof. Let $\{\phi_m(a_i), \pi(i)\}_{i=1}^d$ be the C_L-approximate design at batch m and $a_0 = a_m^\star$. The regret at batch m can be bounded as

$$R^{(m)} \leq T_m \Delta_{a_0} + \sum_{i=1}^{d} n_m(a_i)\Delta_{a_i}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \tag{B.69}$$

We first modify the first term in (B.69) to put it in the same form of the terms inside the summation. Towards that, we expand the definition of $n_m(i)$ to include a_0 by letting $\pi(0) = 16$ ($n_m(0)$ and $\pi(0)$ are values used only for analysis and may not reflect the actual number of pulls for action a_0) and

$$n_m(0) = \lceil \frac{\pi(0)T_m/8}{(1+\sqrt{T_{m-1}}\Delta_m(a_0)/(8\gamma))^2} \rceil. \tag{B.70}$$

By definition of $a_0 = a_m^\star$ we also have that $\Delta_m(a_0) \leq 1/T$. Hence, we have that

$$\frac{1}{(1+\sqrt{T_{m-1}}\Delta_m(a_0)/(8\gamma))^2} \geq 1/2. \tag{B.71}$$

Substituting in (B.69), and using $\pi(0) = 16$, we get that

$$R^{(m)} \leq \sum_{i=0}^{d} n_m(a_i)\Delta_{a_i}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \tag{B.72}$$

We notice that on $\tilde{G}_m$ we have

$$\begin{aligned} n_m(i) &= \lceil \frac{\pi(i)T_m/8}{(1+\sqrt{T_{m-1}}\Delta_m(a_i)/(8\gamma))^2} \rceil \leq 1 + \frac{\pi(i)T_m/8}{(1+\sqrt{T_{m-1}}\Delta_m(a_i)/(8\gamma))^2} \\ &\leq 1 + \frac{\pi(i)T_m/8}{(1+\sqrt{T_{m-1}}(1/2\Delta_{a_i} - 4\gamma/\sqrt{T_{m-1}})/(8\gamma))^2} \\ &= 1 + \frac{\pi(i)T_m/8}{(1/2 + 1/16\sqrt{T_{m-1}}\Delta_{a_i}/\gamma)^2} \end{aligned} \tag{B.73}$$

This implies that

$$n_m(i) \leq 1 + \min\{T_m/2, \frac{32\gamma^2 T_m}{T_{m-1}\Delta_{a_i}^2}\}\pi(i) \tag{B.74}$$

The last part of the lemma follows from (B.74) since $\sum_{i=1}^{d} n_m(i) \le d + T_m/2 \sum_{i=1}^{d} \pi(i) = d + T_m/2 \le T_m$, where the last inequality follows from $T_1 \ge 2d$. Substituting in (B.72), we get that

$$\begin{aligned} R^{(m)} &\le \sum_{i=0}^{d} n_m(a_i)\Delta_{a_i}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\ &\le d + 1 + \sum_{i=0}^{d} \pi(i) \min\{\Delta_{a_i}T_m/2, \frac{32\gamma^2 T_m}{\Delta_{a_i}T_{m-1}}\}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \end{aligned} \quad \text{(B.75)}$$

Hence, we have that

$$\begin{aligned} R^{(m)} &\le d + 1 + \sum_{i=0}^{d} \pi(i) \sup_{\Delta_{a_i} \ge 0} \min\{\Delta_{a_i}T_m/2, \frac{32\gamma^2 T_m}{\Delta_{a_i}T_{m-1}}\}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\ &\le d + 1 + \sup_{\Delta \ge 0} \min\{\Delta T_m/2, \frac{32\gamma^2 T_m}{\Delta T_{m-1}}\} \sum_{i=0}^{d} \pi(i) \\ &= d + 1 + 17 \sup_{\Delta \ge 0} \min\{\Delta T_m/2, \frac{32\gamma^2 T_m}{\Delta T_{m-1}}\}. \end{aligned} \quad \text{(B.76)}$$

We have that $\min\{\Delta T_m/2, \frac{32\gamma^2 T_m}{\Delta T_{m-1}}\}$ is maximized when $\Delta T_m/2 = \frac{32\gamma^2 T_m}{\Delta T_{m-1}}$, hence, when $\Delta = \frac{8\gamma}{\sqrt{T_{m-1}}}$. Substituting in (B.76) we get that

$$R^{(m)} \le d + 1 + \frac{68\gamma T_m}{\sqrt{T_{m-1}}}. \quad \text{(B.77)}$$

To prove the gap dependent bound on $R^{(m)}$ we start from (B.75). We have that if $\Delta_a \ge \Delta_{\min} \forall a \in \mathcal{A} : \Delta_a > 0$, then

$$\begin{aligned} R^{(m)} &\le d + 1 + \sum_{i=0}^{d} \pi(i) \min\{\Delta_{a_i}T_m/2, \frac{32\gamma^2 T_m}{\Delta_{a_i}T_{m-1}}\}\mathbf{1}[a_i \notin \mathcal{B}_{1/T}] \\ &\le d + 1 + \frac{32\gamma^2 T_m}{\Delta_{\min}T_{m-1}} \sum_{i=0}^{d} \pi(i) \\ &\le d + 1 + \frac{544\gamma^2 T_m}{\Delta_{\min}T_{m-1}} \end{aligned} \quad \text{(B.78)}$$

This concludes the proof of the lemma. □

To combine the regret across different batches we notice that since $\sum_{m=1}^{M} T_m \ge T$, Algorithm 3 will finish in at most M batches. The following result follows from Lemma 17.

Lemma 18. Suppose $T_m \geq T_{m-1}, \forall m \in [M]$, $\sum_{m=1}^{M} T_m \geq T$ and $T_0 = 1, T_1 \geq 2d$, then there exists a universal constant C such that with probability at least $1-\delta$ the regret of Algorithm 3 is bounded as

$$R_T \leq C \sum_{m=1}^{M} \frac{\gamma T_m}{\sqrt{T_{m-1}}}, \tag{B.79}$$

where $\gamma = 8d\sqrt{C_L(\log(1/\delta) + \log T)}$. Moreover, if $\forall a \in \mathcal{A}$ with $\Delta_a > 0$ we have $\Delta_a \geq \Delta_{\min}$ then with probability at least $1-\delta$ the regret of Algorithm 3 is bounded as

$$R_T \leq C \sum_{m=1}^{M} \frac{\gamma^2 T_m}{\Delta_{\min} T_{m-1}}. \tag{B.80}$$

Finally, we use the two sets of batch lengths proposed in [GHR19]. The first set of batch lengths is suitable for worst case regret bounds. We choose the following batch lengths $\{T_m\}$:

$$T_m = \max\{\lfloor T^{1-2^{-m}} \rfloor, 2d\}, m \in [M-1], T_M = T, M = \lceil \log\log T \rceil + 1. \tag{B.81}$$

We note that $\sum_{m=1}^{M} T_m \geq T$, however, Algorithm 3 finishes whenever the number of rounds reaches T, hence, the number of batches is upper bounded by M. We also notice that $T_1 \geq 2d, T_m \geq T_{m-1} \forall m \in [M]$. To prove the first regret bound we observe that for $T \geq 2$ and $2 \leq m \leq M-1$ we have

$$\frac{T_m}{\sqrt{T_{m-1}}} \leq \frac{\lfloor T^{1-2^{-m}} \rfloor}{\sqrt{\lfloor T^{1-2^{-m+1}} \rfloor}} \leq \frac{T^{1-2^{-m}}}{\sqrt{\lfloor T^{1-2^{-m+1}} \rfloor}} = \frac{\sqrt{T}\sqrt{T^{1-2^{-m+1}}}}{\sqrt{\lfloor T^{1-2^{-m+1}} \rfloor}} \leq 2\sqrt{T}. \tag{B.82}$$

We also have that

$$\frac{T_M}{\sqrt{T_{M-1}}} = \frac{T}{\lfloor T^{1-2^{-\log\log T}} \rfloor} = \frac{T}{\lfloor T/2 \rfloor} \leq 4. \tag{B.83}$$

Hence, in all cases we have $\frac{T_m}{\sqrt{T_{m-1}}} \leq 4\sqrt{T}$. The regret bound follows by noticing that the regret of the first batch can be bounded by $T_1 \leq \max\{2d, \sqrt{T}+1\}$ and substituting in (B.79).

The second set of batch lengths $\{T_m\}$ is suitable for gap dependent regret bounds. We choose the following batch lengths

$$T_m = d4^m, m \in [M], M = \lceil \log_4 T \rceil. \tag{B.84}$$

We notice $T_1 \geq 2d, T_m \geq T_{m-1} \forall m \in [M], \sum_{m=1}^{M} T_m \geq T$ (Algorithm 3 finishes whenever the number of rounds reaches T, hence, the number of batches is upper bounded by M). The gap dependent regret bound directly follows by substituting the batch lengths from (B.84) in (B.80). □

B.5 Proof of Theorem 9: Complexity of Algorithm 3

Theorem. *Algorithm 3 finishes in $\tilde{O}(Td^2 + d^4 M + \mathcal{T}_{opt} d^3 M)$ runtime and uses $\tilde{O}(d^2 + \mathcal{M}_{opt})$ memory, where $\mathcal{T}_{opt}, \mathcal{M}_{opt}$ are the time and space complexity of the linear optimzation oracle for the action set $\mathcal{A}$.*

Proof. We notice that the runtime and space complexity of LW-ArgMax is

$$\mathcal{T}_{\text{LW-ArgMax}} = O((d + \mathcal{T}_{\text{opt}}) \log^3 T), \mathcal{M}_{\text{LW-ArgMax}} = O(d \log^3 T + \mathcal{M}_{\text{opt}}). \tag{B.85}$$

We next upper bound the complexity of Algorithm 4. As the matrix $\boldsymbol{A}$ is invertible in all iterations, we can use the rank-one update formula of the determinent [Mey00] to perform steps 5 and 10 in $O(d)$ runtime and $O(d^2)$ space complexity. Namely

$$\begin{aligned}
\det(a, \boldsymbol{A}_{-i}) &= \det(\boldsymbol{A} + (a - a_i)e_i^\top) = \det(A)(1 + e_i^\top \boldsymbol{A}^{-1}(a - a_i)) \\
&= \langle a, \det(\boldsymbol{A})(\boldsymbol{A}^{-1})^\top e_i \rangle + \det(\boldsymbol{A})(1 - e_i^\top \boldsymbol{A}^{-1} a_i) \\
&= \langle a, \det(\boldsymbol{A})(\boldsymbol{A}^{-1})^\top e_i \rangle,
\end{aligned} \tag{B.86}$$

where the last step follows by noticing that the formula is valid for $a = 0$, $\tilde{a}_i$ is the i-th column of $\boldsymbol{A}$. This requires the inverse of matrix $\boldsymbol{A}$ which can be computed using rank-one updates in $O(d^2)$ time and $O(d^2)$ space [SM50]

$$(\boldsymbol{A} + (a - \tilde{a}_i)e_i^\top)^{-1} = \boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}(a - \tilde{a}_i)e_i^\top \boldsymbol{A}^{-1}}{1 + e_i \boldsymbol{A}^{-1}(a - \tilde{a}_i)}. \tag{B.87}$$

We notice that for each repetition of the second for loop, $\boldsymbol{A}^{-1}$ is updated once while $\det(\boldsymbol{A})$ can be updated at most d times. Hence, the time and space complexity of one repetition of the for loop in Algorithm 4 is $O(\mathcal{T}_{\text{LW-ArgMax}} d + d^2), O(\mathcal{M}_{\text{LW-ArgMax}} + d^2)$ respectively. By Lemma 3, the for loops is repeated at most $O(d^2 \log d)$ times. Hence, the time and space complexity of Algorithm 4 can be bounded as

$$\mathcal{T}_{\text{LWS}} = O\left((d^4 + \mathcal{T}_{\text{opt}} d^3) \log d \log^3 T\right), \mathcal{M}_{\text{LWS}} = O\left(d^2 \log^3 T + \mathcal{M}_{\text{opt}}\right). \tag{B.88}$$

We next upper bound the time and space complexity of Algorithm 3.

- The time and space complexity of finding the barycentric spanner in step 5 is $\mathcal{T}_{\text{LWS}}, \mathcal{M}_{\text{LWS}}$ respectively.
- The computation of the least squares matrix requires $O(T_m d^2)$ time and $O(d^2)$ space, while its inversion requires $O(d^3)$ runtime. Hence, θ_m can be computed in $O(T_m d^2 + d^3)$ time and $O(d^2)$ space.
- Computing the estimated best action in step 11 requires $\mathcal{T}_{\text{opt}}, \mathcal{M}_{\text{opt}}$ time and space respectively.

Hence, in total Algorithm 3 runtime is $O(Td^2 + (d^4 + \mathcal{T}_{\text{opt}} d^3) M \log d \log^3 T)$ while the space complexity is $O\left(d^2 \log^3 T + \mathcal{M}_{\text{opt}}\right)$. □

B.6 Proof of Theorem 10: Regret Analysis for Contextual Bandits

Theorem 3. *Consider a contextual linear bandit instance with $\mathcal{A}_t$ generated from an unknown distribution $\mathcal{D}$. There exists a universal constant C and a choice for batch lengths such that Algorithm 6 , with $q = (1 - \exp(-1))/(24T^{7+12\log T})$, finishes in $O(\log \log T)$ batches with regret upper bounded as*

$$R_T \leq C\gamma\sqrt{T} \log \log T$$

with probability at least $1 - 2\delta$, where $\gamma = 10\sqrt{C_L d(\log(8M/\delta) + 57d \log^2(6T))}$. Moreover, the running time and space complexity are $\tilde{O}(d^4 + \mathcal{T}_{opt} d^3 T)$ and $\tilde{O}(d^2 + \mathcal{M}_{opt})$, respectively.

Proof. Recall that at each round t, Algorithm 6 pulls action a_t associated with a value θ_t (see step 7 in Algorithm 6). To upper bound the regret, we follow a technique proposed in [HYF23] by first upper bounding the quantity

$$R_T^L = \sum_{t=1}^{T} \sup_{\theta \in \Theta} \langle g(\theta) - g(\theta_t), \theta_\star \rangle \tag{B.89}$$

which can be thought of as the regret of the algorithm on a reduced linear bandit instance [HYF23]. Then we can use Theorem 1 in [HYF23] which states that $|R_T - R_T^L| = \tilde{O}(\sqrt{T})$ with high probability to upper bound the regret R_T.

As in the proof of Theorem 8 instead of analyzing Algorithm 6 which ends batch m if the total number of pulls reaches T_m, we analyze a variant algorithm that completes all the required pulls of the actions in the barycentric spanner. We bound the regret of the variant algorithm when a good event $\tilde{G}$ (that we define later) holds, and show that $\mathbb{P}[\tilde{G}] \geq 1 - \delta$. Then, we show that conditioned on $\tilde{G}$, it holds that $\sum_{i=1}^{d} n_m(i) \leq \tau_m$, for all batches $m \in [2M]$ (see (B.101)), which implies that Algorithm 6 coincides with the variant algorithm on $\tilde{G}$ in this case. We also refer to the variant algorithm as Algorithm 6 for simplicity.

Recall that

$$g(\theta) = \mathbb{E}_{\mathcal{A} \sim \mathcal{D}}[\mathcal{O}(\mathcal{A}; \theta)], g^{(m)}(\theta) = \frac{1}{|H_{m-1}|} \sum_{t \in H_{m-1}} \mathcal{O}(\mathcal{A}_t; \theta), \mathcal{X}_m = \{g^{(m)}(\theta) | \theta \in \Theta'\}, \tag{B.90}$$

where H_m is the set of indices for the rounds in batch m and $\Theta' = \{[\theta]_q | \theta \in \Theta\}$ is a discretization of Θ, $[\theta]_q = q\lfloor \theta\sqrt{d}/q \rfloor / \sqrt{d}$ and q is the discretization parameter. Recall also that $\boldsymbol{V}_m$ is the regularized least squares matrix in step 12 of Algorithm 6 with $\lambda = 1$, and denote $\epsilon_m = \sup_{\theta' \in \Theta', \theta \in \Theta} |\langle g^m(\theta') - g(\theta'), \theta \rangle|$ in the extended reals $\mathbb{R} \cup \{\infty\}$. We also denote $\epsilon(t) = \langle g(\theta^{(t)}) - g^{(m)}(\theta^{(t)}), \theta_\star \rangle$, where $g^{(m)}(\theta^{(t)}) \in \mathcal{X}_m, r_t$ are the action and reward at iteration t of batch m. We first upper bound the error in estimating $\mu_{g(\theta')} = \langle g(\theta'), \theta_\star \rangle$ for an action $g(\theta')$ at the end of batch m for $\theta' \in \Theta'$. We have that for any $a \in \mathbb{R}^d$, $|\langle a, \theta_{m+1} - \theta_\star \rangle|$ can be decomposed as

$$\begin{aligned} |\langle a, \theta_{m+1} - \theta_\star \rangle| &= |\langle a, \boldsymbol{V}_m^{-1} \sum_{t=1}^{\tau_m} r_t a_t - \theta_\star \rangle| \overset{(i)}{=} |\langle a, \boldsymbol{V}_m^{-1} \sum_{t=1}^{\tau_m} (\theta_\star^\top g(\theta^{(t)}) + \eta_t') g^{(m)}(\theta^{(t)}) - \theta_\star \rangle| \\ &= |\langle a, \boldsymbol{V}_m^{-1} \sum_{t=1}^{\tau_m} (\theta_\star^\top g^{(m)}(\theta^{(t)}) + \epsilon(t) + \eta_t') g^{(m)}(\theta^{(t)}) - \theta_\star \rangle| \end{aligned}$$

$$
\begin{aligned}
&= |\langle a, \boldsymbol{V}_m^{-1}\left((\boldsymbol{V}_m - \boldsymbol{I})\theta_\star + \sum_{t=1}^{\tau_m}(\epsilon(t) + \eta'_t)g^{(m)}(\theta^{(t)})\right) - \theta_\star\rangle| \\
&= |\langle a, -\boldsymbol{V}_m^{-1}\theta_\star + \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}(\epsilon(t) + \eta'_t)g^{(m)}(\theta^{(t)})\rangle| \\
&\leq |\langle a, -\boldsymbol{V}_m^{-1}\theta_\star\rangle| + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\epsilon(t)g^{(m)}(\theta^{(t)})\rangle| \\
&\qquad + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\eta'_t g^{(m)}(\theta^{(t)})\rangle| \\
&\leq \|a\|_{\boldsymbol{V}_m^{-1}}\|\theta_\star\|_{\boldsymbol{V}_m^{-1}} + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\epsilon(t)g^{(m)}(\theta^{(t)})\rangle| \\
&\qquad + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\eta'_t g^{(m)}(\theta^{(t)})\rangle| \\
&\overset{(ii)}{\leq} \|a\|_{\boldsymbol{V}_m^{-1}} + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\epsilon(t)g^{(m)}(\theta^{(t)})\rangle| + |\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\eta'_t g^{(m)}(\theta^{(t)})\rangle|, \qquad \text{(B.91)}
\end{aligned}
$$

where (i) follows from Theorem 1 in [HYF23], η'_t is a zero mean noise conditioned on the filtration of history and $\theta^{(t)}$ and (ii) uses $\boldsymbol{V}_m \geq \boldsymbol{I}$. We next bound the term $|\langle a, \boldsymbol{V}_m^{-1}\sum_{t\in H_m}\epsilon(t)g^{(m)}(\theta^{(t)})\rangle|$. We have that

$$
\begin{aligned}
|\langle a, \boldsymbol{V}_m^{-1}\sum_{t=1}^{\tau_m}\epsilon(t)g^{(m)}(\theta^{(t)})\rangle| &\leq \sqrt{\tau_m\sum_{t=1}^{\tau_m}\epsilon(t)^2 a^\top \boldsymbol{V}_m^{-1}g^{(m)}(\theta^{(t)})g^{(m)}(\theta^{(t)})^\top \boldsymbol{V}_m^{-1}a} \\
&\overset{(i)}{\leq} \epsilon_m\sqrt{\tau_m\sum_{t=1}^{\tau_m} a^\top \boldsymbol{V}_m^{-1}g^{(m)}(\theta^{(t)})g^{(m)}(\theta^{(t)})^\top \boldsymbol{V}_m^{-1}a} \\
&\leq \epsilon_m\sqrt{\tau_m a^\top \boldsymbol{V}_m^{-1}(\boldsymbol{V}_m - \boldsymbol{I})\boldsymbol{V}_m^{-1}a} \\
&\leq \epsilon_m\sqrt{\tau_m(\|a\|^2_{\boldsymbol{V}_m^{-1}} - \|a\|^2_{\boldsymbol{V}_m^{-2}})} \leq \epsilon_m\sqrt{\tau_m(\|a\|^2_{\boldsymbol{V}_m^{-1}} - \|a\|^2_{\boldsymbol{V}_m^{-2}})} \\
&\leq \epsilon_m\sqrt{\tau_m(\|a\|^2_{\boldsymbol{V}_m^{-1}} - \|a\|^2_{\boldsymbol{V}_m^{-2}})} \leq \epsilon_m\sqrt{\tau_m\|a\|^2_{\boldsymbol{V}_m^{-1}}}, \qquad \text{(B.92)}
\end{aligned}
$$

where (i) follows by the definition of $\epsilon_m = \sup_{\theta'\in\Theta',\theta\in\Theta}|\langle g^m(\theta') - g(\theta'), \theta\rangle|$ and $\epsilon(t) = \langle g(\theta^{(t)}) - g^{(m)}(\theta^{(t)}), \theta_\star\rangle$. We have from Theorem 2 in [HYF23] that the following event holds with probability at least $1 - \delta/(4M)$

$$\mathcal{G}_m^\epsilon : \epsilon_m \le 2\sqrt{\frac{\log(8M|\Theta'|/\delta)}{\tau_{m-1}}}. \tag{B.93}$$

We also have that from eq. (20.2) of [LS20] the following event holds with probability at least $1 - \delta/(4M)$

$$\begin{aligned}\mathcal{G}_m^\eta : |\langle a, \boldsymbol{V}_m^{-1} \sum_{t=1}^{\tau_m} \eta_t' g^{(m)}(\theta^{(t)}) \rangle| &\le \sqrt{2 \sum_{t=1}^{\tau_m} (a^\top \boldsymbol{V}_m^{-1} g^{(m)}(\theta^{(t)}))^2 \log(4M|\Theta'|/\delta)} \\ &\overset{(i)}{\le} \sqrt{2\|a\|_{\boldsymbol{V}_m^{-1}}^2 \log(4M|\Theta'|/\delta)} \forall a \in \tilde{\mathcal{X}}_m, \end{aligned} \tag{B.94}$$

where (i) follows by expanding $(a^\top \boldsymbol{V}_m^{-1} g^{(m)}(\theta^{(t)}))^2$ as in (B.92). From Lemma 1 we have that for $q = (1 - e^{-1})/(24T^{7+12\log T})$, the function $g^{(m)}([\theta]_q)$ is an approximate linear optimization oracle with additive gap at most $(1 - e^{-1})/(12T^{7+12\log T})$. Hence, using Lemma 2[1] and Lemma 3, Algorithm 4 finds a set $\mathcal{C}_m$ such that $\{\phi_m(a) | a \in \mathcal{C}_m\}$ is an e^8 approximate spanner for $\tilde{\mathcal{X}}_m$. By the properties of the C_L-approximate design, similar to (B.44) we have that $\|\phi_m(a)\|_{\boldsymbol{V}_m^{-1}} \le \sqrt{C_L d/\tau_m} \forall a \in \mathcal{X}_m'$, where $C_L = e^8 d$. Hence, substituting from (B.92), (B.93) and (B.94) in (B.91) we get that the following holds on $\mathcal{G}_m^\eta \cap \mathcal{G}_m^\epsilon$

$$|\langle \phi_m(a), \theta_{m+1} - \theta_\star \rangle| \le \sqrt{C_L d/\tau_m} + 4\sqrt{\frac{C_L d \log(8M|\Theta'|/\delta)}{\tau_{m-1}}} \le 5\sqrt{\frac{C_L d \log(8M|\Theta'|/\delta)}{\tau_{m-1}}} \forall a \in \mathcal{X}_m'. \tag{B.95}$$

We notice that for $q = (1 - e^{-1})/(24T^{7+12\log T})$, we have that $|\Theta'| \le 6T^{3d(7+12\log T)}$. Hence, $\log|\Theta'| \le 57d\log^2(6T)$. Hence, $C_L d \log(8M|\Theta'|/\delta) = C_L d(\log(8M/\delta) + 57d\log^2(6T))$. Substituting in (B.95) we get that the following holds on $\mathcal{G}_m^\eta \cap \mathcal{G}_m^\epsilon$

$$|\langle \phi_m(a), \theta_{m+1} - \theta_\star \rangle| \le 5\sqrt{C_L d(\log(8M/\delta) + 57d\log^2(6T))/\tau_{m-1}} \le \frac{\gamma/2}{\sqrt{\tau_{m-1}}}. \tag{B.96}$$

By definition of ϕ_m it follows that the following holds on $\mathcal{G}_m^\eta \cap \mathcal{G}_m^\epsilon$

$$|\langle a, \theta_{m+1} - \theta_\star \rangle| \le \frac{\gamma/2}{\sqrt{\tau_{m-1}}} + \frac{\Delta_m(a)}{8}\sqrt{\frac{\tau_{m-2}}{\tau_{m-1}}} \forall a \in \mathcal{X}_m'. \tag{B.97}$$

[1]The verification of the conditions stated in Lemma 2 is equivalent to the verification conducted at the beginning of the proof of Theorem 8.

Hence, by definition of $\mathcal{G}_m^\epsilon$ in (B.93) the following holds on $\mathcal{G}_m^\eta \cap \mathcal{G}_m^\epsilon$

$$\begin{aligned} |\langle a, \theta_{m+1}\rangle - \mu_a| &\leq |\langle a, \theta_{m+1} - \theta_\star\rangle| + |\langle a, \theta_\star\rangle - \mu_a| \\ &\leq \frac{\gamma/2}{\sqrt{\tau_{m-1}}} + \frac{\Delta_m(a)}{8}\sqrt{\frac{\tau_{m-2}}{\tau_{m-1}}} + \epsilon_m \\ &\leq \frac{\gamma}{\sqrt{\tau_{m-1}}} + \frac{\Delta_m(a)}{8}\sqrt{\frac{\tau_{m-2}}{\tau_{m-1}}} \forall a \in \mathcal{X}_m'. \end{aligned} \tag{B.98}$$

We recall that $\mathbb{P}[\mathcal{G}_m^\epsilon] \geq 1 - \delta/(4M)$, $\mathbb{P}[\mathcal{G}_m^\eta] \geq 1 - \delta/(4M)$. Hence, by the union bound we have that

$$\mathbb{P}[\tilde{\mathcal{G}}] \geq 1 - \delta, \tilde{\mathcal{G}} = \cap_{m \in [2M]}(\mathcal{G}_m^\eta \cap \mathcal{G}_m^\epsilon) \tag{B.99}$$

Then, following the proof of Lemma 16 by replacing every τ_m with τ_{m-1} and every τ_{m-1} with τ_{m-2} we get that the following event hold on $\tilde{\mathcal{G}}$

$$-4\frac{\gamma}{\sqrt{\tau_{m-2}}} + \frac{1}{2}\Delta_a \leq \Delta_m(a) \leq 2\Delta_a + 4\frac{\gamma}{\sqrt{\tau_{m-2}}} \forall a \in \mathcal{A}' \forall m \in M. \tag{B.100}$$

Hence, following the same steps as in Lemma 17 we get that there is a universal constant C such that the following holds on $\tilde{\mathcal{G}}$

$$R_T^L \leq C\sum_{m=1}^{2M} \frac{\gamma\tau_m}{\sqrt{\tau_{m-2}}} = C\sum_{m=1}^{2M} \frac{\gamma T_{m//2}}{\sqrt{T_{m//2-1}}}, \sum_{i=1}^{d} n_m(i) \leq \tau_m, \tag{B.101}$$

where R_T^L is the regret of the algorithm on the linear bandit instance defined in (B.89). Using the batch lengths in (B.81), we get, from (B.82), that the following holds on $\tilde{\mathcal{G}}$

$$R_T^L \leq 8C\gamma\sqrt{T}M \tag{B.102}$$

From Theorem 1 in [HYF23] we have that $|R_T^L - R_T| \leq \sqrt{T\log(T/\delta)}$ with probability at least $1 - \delta$. By the union bound and triangle inequality it follows that

$$R_T \leq 16C\gamma\sqrt{T}M \tag{B.103}$$

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

The complexity result follows from Theorem 9 by observing that computing $g^{(m)}([\theta]_q)$ (our approximate oracle) requires at most T calls to $\mathcal{O}(\mathcal{A}_t; .)$. Hence, the time and space complexity of Algorithm 6 are $O((d^4 + \mathcal{T}_{\text{opt}}d^3T)M\log d\log^3 T)$ and $O\left(d^2\log^3 T + \mathcal{M}_{\text{opt}}\right)$, respectively. $\square$

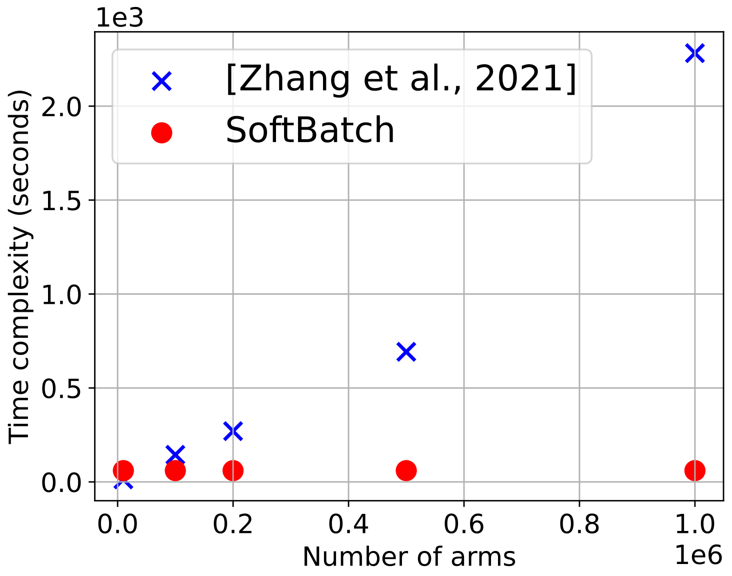


Figure B.1: Complexity of computing our exploration policy versus the state of the art complexity.

B.7 Numerical Comparison of Complexity of Our Scheme

In this appendix we present a small experiment to compare the computational complexity of computing the exploration policy of Algorithm 6 versus the complexity of computing the policy in [ZJZ21] (complexity of one batch). We do not consider other batched algorithms such as [RYZ21] since they are not feasible to implement even for a small number of actions. We used $d = 5$ dimensions and a batch of size 100 iterations. For simplicity we use a fixed action set (unit sphere), however, this knowledge is not revealed to the algorithms, i.e., the algorithms assume that the action set may change over time. As the policy of [ZJZ21] requires to solve a non-convex optimization problem, it is not feasible to implement it for infinite number of actions. Instead, we solve the optimization problem over a finite subset of k actions sampled uniformly at random from the action set. In contrast, our algorithm can be directly applied for the infinite action set, hence, the computational complexity will not depend on k. In Fig. B.1, we plot the time complexity versus the sampled number of actions (on Intel(R) Xeon(R) CPU @ 2.20GHz, 56MB cache). We observe that for moderately large number of actions, our algorithm achieves significant savings in computational complexity as compared to the scheme of [ZJZ21].

APPENDIX C

Appendix for Chapter 4: Reward Compression for Bandits

C.1 Proof of Proposition 1: Properties of the Quantized Rewards

Proof. We start by proving that $\hat{r}_t$ is an unbiased estimate of μ_{A_t}. If $-3 \le r_t \le -4$, we have that $\hat{r}_t$ takes the value $\lceil r_t \rceil$ with probability $r_t - \lfloor r_t \rfloor$, and the value $\lfloor r_t \rfloor$ with probability $\lceil r_t \rceil - r_t$. Hence, $\mathbb{E}[\hat{r}_t | r_t] = r_t$. For all the other cases we have that

$$\begin{aligned}
\mathbb{E}[\hat{r}_t|r_t] &= \mathbb{E}[M_t(s_t(e_t^{(q)} + \ell_t + 3.5s_t + 0.5) + \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor)|r_t] \\
&= \mathbb{E}[M_t\mathbb{E}[s_t(e_t^{(q)} + \ell_t + 3.5s_t + 0.5) + \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor \\
&\qquad + s_t\ell_t | r_t, \hat{\mu}(t), M_t]|r_t] \\
&\overset{(i)}{=} \mathbb{E}\left[M_t\left(\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor + s_t(-[3.5s_t + 0.5 + \ell_t] \right.\right. \\
&\qquad \left.\left. + s_t\ell_t + 3.5s_t + 0.5) + \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor\right) |r_t\right] \\
&= r_t, \qquad (C.1)
\end{aligned}$$

where (i) follows from the fact that the stochastic quantization (SQ) that we use gives an unbiased estimate of the input. We note that from Algorithm 8, e_t encodes $|\bar{r}_t| - (|a| + 1) - \ell_t$, where $|a| + 1 = 3$ when $s_t = -1$, and $|a| + 1 = 4$ when $s_t = 1$, i.e., $|a| + 1 = 3.5s_t + 0.5$. Hence, in all cases we have that

$$\begin{aligned}
\mathbb{E}[\hat{r}_t|A_t] &= \mathbb{E}[\mathbb{E}[\hat{r}_t|r_t, A_t]|A_t] = \mathbb{E}[\mathbb{E}[\hat{r}_t|r_t]|A_t] \\
&= \mathbb{E}[r_t|A_t] = \mu_{A_t} \qquad (C.2)
\end{aligned}$$

The bound on $|r_t - \hat{r}_t|$ follows from the fact that the distance between the quantization levels for which we use the randomized quantization is 1, hence, in all cases we have that

$$1 \geq |s_t e_t^{(q)} - (\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor - s_t \ell_t)| = \frac{|\hat{r}_t - r_t|}{M_t}. \tag{C.3}$$

We note that this implies

$$\begin{aligned}\mathbb{E}[|\hat{r}_t - \mu_{A_t}|^2 | A_t] &= \mathbb{E}[|\hat{r}_t - r_t + r_t - \mu_{A_t}|^2 | A_t] \\ &= \mathbb{E}[|\hat{r}_t - r_t|^2 | A_t] + \mathbb{E}[|r_t - \mu_{A_t}|^2 | A_t] \\ &\qquad + 2\mathbb{E}[(r_t - \mu_{A_t})(\hat{r}_t - r_t)|A_t] \\ &\leq (1+\epsilon^2)\sigma^2 \\ &\quad + 2\mathbb{E}[(r_t - \mu_{A_t})\mathbb{E}[(\hat{r}_t - r_t)|A_t, r_t]|A_t] \\ &= (1+\epsilon^2)\sigma^2. \end{aligned} \tag{C.4}$$

To see that conditioned on A_t, $\hat{r}_t$ is conditionally independent on the history $A_1, \hat{r}_1, ..., A_{t-1}, \hat{r}_{t-1}$, we notice that since we replace $\frac{\hat{\mu}(t)}{M_t}$ by an integer, $\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor$ and since the distance between the quantization levels is 1, we have that the two nearest quantization levels to $\frac{r_t}{M_t}$ are at $\lfloor \frac{r_t}{M_t} \rfloor, \lceil \frac{r_t}{M_t} \rceil$. Hence, conditioned on M_t, $\hat{r}_t$ takes the value $M_t \lceil \frac{r_t}{M_t} \rceil$ with probability $\frac{r_t}{M_t} - \lfloor \frac{r_t}{M_t} \rfloor$, and the value $M_t \lfloor \frac{r_t}{M_t} \rfloor$ with probability $\lceil \frac{r_t}{M_t} \rceil - \frac{r_t}{M_t}$. This shows that despite the fact that the encoding of $\hat{r}_t$ is a function of $r_1, ..., r_t$, the value of $\hat{r}_t$ is a function of r_t only, since M_t is generated independently of the history. As a result, given A_t, $\hat{r}_t$ is conditionally independent on the history $A_1, \hat{r}_1, ..., A_{t-1}, \hat{r}_{t-1}$.

The fact that $\hat{r}_t$ is subgaussian can be proven by Cauchy-Schwartz

$$\begin{aligned}\mathbb{E}[e^{\lambda(\hat{r}_t - \mu_{A_t})}|A_t] &= \mathbb{E}[e^{\lambda(\hat{r}_t - r_t + r_t - \mu_{A_t})}|A_t] \\ &\leq \mathbb{E}[e^{p\lambda(\hat{r}_t - r_t)}|A_t]^{\frac{1}{p}} \\ &\qquad \mathbb{E}[e^{(1-p)\lambda(r_t - \mu_{A_t})}|A_t]^{\frac{1}{1-p}} \\ &\leq e^{\lambda^2 \frac{\sigma^2(1+\frac{\epsilon}{2})^2}{2}}, \end{aligned} \tag{C.5}$$

where $p = 1 + \frac{2}{\epsilon}$. To bound the expected regret after quantization we observe that $R_T = \sum_{t=1}^T \mathbb{E}(\mu_t^* - r_t) = \sum_{t=1}^T \mathbb{E}(\mu_t^* - \hat{r}_t) = (1 + \frac{\epsilon}{2}) \sum_{t=1}^T \mathbb{E}(\frac{\mu_t^* - \hat{r}_t}{1+\frac{\epsilon}{2}})$. We have that $\frac{\hat{r}_t}{(1+\frac{\epsilon}{2})}$ is σ^2-subgaussian. Applying the bandit algorithm using $\frac{\hat{r}_t}{(1+\frac{\epsilon}{2})}$ results in $\sum_{t=1}^{T} \mathbb{E}(\frac{\mu_t^* - \hat{r}_t}{(1+\frac{\epsilon}{2})}) \leq R_T^U(\{\Delta_i/(1+\frac{\epsilon}{2})\})$, hence

$$R_T \leq (1+\frac{\epsilon}{2}) R_T^U(\{\Delta_i/(1+\frac{\epsilon}{2})\}). \tag{C.6}$$

□

C.2 Proof of Theorem 11: Upper Bounds on the Number of Bits

Proof. We have that B_t can be bounded as

$$\begin{aligned}
B_t &\leq 3 + \mathbf{1}[\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor > 3] + \mathbf{1}[\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor - \frac{r_t}{M_t} > 2] \\
&\quad + 2(\mathbf{1}[\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor > 4] \lceil \log(\frac{r_t}{M_t} - \lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor - 3) \rceil) \\
&\quad + 2(\mathbf{1}[\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor - \frac{r_t}{M_t} > 3] \lceil \log(\lfloor \frac{\hat{\mu}(t)}{M_t} \rfloor - \frac{r_t}{M_t} - 2) \rceil) \\
&\leq 3 + \mathbf{1}[|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| > 2] + 2(\mathbf{1}[|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| > 3]) \\
&\quad + 2(\mathbf{1}[|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| > 3] \log(|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| - 2)).
\end{aligned} \tag{C.7}$$

Hence for each $\delta > 0$, we have

$$\begin{aligned}
B_t &\leq 3 + \mathbf{1}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 2(1-\delta)] + \mathbf{1}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 2\delta] \\
&\quad + 2(\mathbf{1}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 3(1-\delta)] + \mathbf{1}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 3\delta]) \\
&\quad + 2(\mathbf{1}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 3]) \log(|\frac{r_t - \hat{\mu}(t)}{\sigma}| - 2).
\end{aligned} \tag{C.8}$$

Taking the expectation of both sides, we get that

$$\begin{aligned}
\mathbb{E}[B_t] \leq 3 &+ \mathbb{P}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 2(1-\delta)] \\
&+ \mathbb{P}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 2\delta] \\
&+ 2(\mathbb{P}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 3(1-\delta)] \\
&+ \mathbb{P}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 3\delta])
\end{aligned}$$

$$+ 2\mathbb{E}[(\mathbf{1}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 3])\log(|\frac{r_t - \hat{\mu}(t)}{\sigma}| - 2)]. \tag{C.9}$$

Hence, there are universal constants C, C' such that

$$\begin{aligned}
\mathbb{E}[B_t] &\leq 3.32 + C'\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|] \\
&\quad + 2\mathbb{E}[\mathbf{1}[|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| > 3](|\frac{r_t}{M_t} - \frac{\hat{\mu}(t)}{M_t}| - 3)] \\
&\leq 3.32 + C'\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|] \\
&\quad + 2\mathbb{E}[\mathbf{1}[|\frac{r_t - \mu_{A_t}}{\sigma}| > 3(1-\delta)]||\frac{r_t - \mu_{A_t}}{\sigma}| - 3|] \\
&\quad + 2\mathbb{E}[\mathbf{1}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 3\delta]||\frac{r_t - \mu_{A_t}}{\sigma}| - 3|] \\
&\quad + 2\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|] \\
&\leq 3.32 + (C' + 2)\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|] \\
&\quad + 2\sum_{i=3}^{\infty} |i(1-\delta) - 3|\mathbb{P}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > i(1-\delta)] \\
&\quad + 2\mathbb{E}[\mathbf{1}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| > 3\delta]]\mathbb{E}[||\frac{r_t - \mu_{A_t}}{\sigma}| - 3|] \\
&\quad + 2\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|] \\
&\leq 3.4 + C\mathbb{E}[|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|]
\end{aligned} \tag{C.10}$$

From (C.8), $\mathbb{E}[|r_t - \mu_{A_t}|^2 | A_t] \leq \sigma^2$, Markov property and the strong law of large numbers for martingales, we also have that there is a universal constant C such that

$$\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} B_t \leq 3.4 + \lim_{T\to\infty} \frac{C}{T}\sum_{t=1}^{T} |\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}| \quad \text{almost surely.} \tag{C.11}$$

It then remains to analyze $|\mu_{A_t} - \hat{\mu}(t)|$ for the three proposed choices of $\hat{\mu}(t)$.

• avg-pt ($\hat{\mu}(t) = \frac{1}{t-1}\sum_{j=1}^{t-1} \hat{r}_j$):

We have that for $t > 1$

$$\frac{|\mu_{A_t} - \hat{\mu}(t)|}{\sigma} \leq \frac{|\mu_{A_t} - \mu^*|}{\sigma} + \frac{|\mu^* - \hat{\mu}(t)|}{\sigma}$$

$$
\begin{aligned}
&= \frac{\Delta_{A_t}}{\sigma} + |\frac{\sum_{j=1}^{t-1} \mu^* - \mu_{A_j} + \mu_{A_j} - \hat{r}_j}{(t-1)\sigma}| \\
&\leq \frac{\Delta_{A_t}}{\sigma} + |\frac{\sum_{j=1}^{t-1} \mu^* - \mu_{A_j}}{(t-1)\sigma}| \\
&\quad + |\frac{\sum_{j=1}^{t-1} \mu_{A_j} - \hat{r}_j}{(t-1)\sigma}| \\
&= \frac{\Delta_{A_t}}{\sigma} + \frac{\sum_{i=1}^{k} \Delta_i T_i(t-1)}{(t-1)\sigma} \\
&\quad + |\frac{\sum_{j=1}^{t-1} \mu_{A_j} - \hat{r}_j}{(t-1)\sigma}|. \qquad \text{(C.12)}
\end{aligned}
$$

For $t = 1$ we have

$$
\begin{aligned}
\frac{|\mu_{A_t} - \hat{\mu}(t)|}{\sigma} &\leq \frac{|\mu_{A_t} - \mu^*|}{\sigma} + \frac{|\mu^* - \hat{\mu}(t)|}{\sigma} \\
&= \frac{\Delta_{A_1}}{\sigma} + \frac{|\mu^*|}{\sigma}. \qquad \text{(C.13)}
\end{aligned}
$$

We then have that

$$
\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T} \log(1+|\frac{\mu_{A_t} - \hat{\mu}(t)}{\sigma}|) &\leq \frac{\log(1 + \frac{|\mu^*|}{\sigma})}{T\sigma} \\
&\quad + \frac{1}{T}\sum_{t=1}^{T} \log(1 + \frac{\Delta_{A_t}}{\sigma}) \\
&\quad + \frac{1}{T}\sum_{t=2}^{T} \log(1 + \frac{\sum_{i=1}^{k} \Delta_i T_i(t-1)}{(t-1)\sigma}) \\
&\quad + \log(1 + |\frac{\sum_{j=1}^{t-1} \mu_{A_j} - \hat{r}_j}{(t-1)\sigma}|) \\
&\leq \frac{\log(1 + \frac{|\mu^*|}{\sigma})}{T\sigma} + \frac{1}{T}\left(\frac{\sum_{i=1}^{k} \Delta_i T_i(T)}{\sigma}\right. \\
&\quad \left. + \sum_{t=1}^{T-1} \frac{\sum_{i=1}^{k} \Delta_i T_i(t)}{t\sigma} + |\frac{\sum_{j=1}^{t} \mu_{A_j} - \hat{r}_j}{t\sigma}|\right). \qquad \text{(C.14)}
\end{aligned}
$$

We have that since $\mathbb{E}[|r_t - \mu_{A_t}|^2 | A_t] \leq \sigma^2$, and Markov property, then by the strong law of large numbers for martingales $\lim_{t\to\infty} \frac{\sum_{j=1}^{t-1} \mu_{A_j} - \hat{r}_j}{(t-1)\sigma} = 0$ almost surely. We then have that if the limit of average regret is 0 almost surely (or in probability), then from (C.11) and (C.14) we get that

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}B_t\leq 3.4 \quad \text{almost surely (or in probability).} \tag{C.15}$$

By observing that we can generate a long sequence of rewards from each arm before the process starts and since $\mathbb{E}[|r_t-\mu_{A_t}|^2|A_t]\leq\sigma^2$, then by the triangle inequality we have that

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[|\frac{\sum_{j=1}^{t-1}\mu_{A_j}-\hat{r}_j}{(t-1)\sigma}|] &\overset{(i)}{\leq}\frac{2}{T}\sum_{t=1}^{T}\frac{1}{\sqrt{t}}\\ &=\frac{2}{T}\sum_{t=1}^{T}\frac{1}{\sqrt{t}}\\ &\leq\frac{2}{T}(1+\int_{t=1}^{T}\frac{1}{\sqrt{t}}dt)\\ &\leq\frac{4}{\sqrt{T}},\end{aligned} \tag{C.16}$$

where (i) follows from the fact that $\mu_{A_j}-\hat{r}_j,\mu_{A_i}-\hat{r}_i$ are uncorrelated for all $i<j$ since

$$\begin{aligned}&\mathbb{E}[(\mu_{A_j}-\hat{r}_j)(\mu_{A_i}-\hat{r}_i)]\\ &\qquad=\mathbb{E}[\mathbb{E}[(\mu_{A_j}-\hat{r}_j)(\mu_{A_i}-\hat{r}_i)|A_j,A_i,\hat{r}_i]]=0.\end{aligned} \tag{C.17}$$

We conclude that there is a universal constant C such that

$$\begin{aligned}\hat{B}(T)\leq 3.4+(C/T)\,&(1+\log(1+|\mu^*|/\sigma)\\ &+R_T/\sigma+\sum_{t=1}^{T-1}R_t/(\sigma t)\Bigg)+C/\sqrt{T}\end{aligned} \tag{C.18}$$

• avg-arm-pt ($\hat{\mu}(t)=\hat{\mu}_{A_t}(t-1)$):

We have that for $T_{A_t}(t-1)>0$

$$\frac{|\mu_{A_t}-\hat{\mu}(t)|}{\sigma}=|\frac{\sum_{j=1}^{t-1}(\mu_{A_t}-\hat{r}_j)\mathbf{1}(A_j=A_t)}{T_{A_t}(t-1)\sigma}|. \tag{C.19}$$

For $T_{A_t}(t-1)=0$, we have that $\hat{\mu}(t)=0$. Then

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\log(1+\frac{|\mu_{A_t}-\hat{\mu}(t)|}{\sigma})]$$

$$\overset{(i)}{\leq} \frac{1}{T}\sum_{i=1}^{k} \log(1+\frac{|\mu_i|}{\sigma}) + \frac{2}{T}\sum_{t=1}^{T}\frac{1}{\sqrt{t}}$$

$$\overset{(ii)}{\leq} \frac{1}{T}\sum_{i=1}^{k} \log(1+\frac{|\mu_i|}{\sigma}) + \frac{4}{\sqrt{T}} \tag{C.20}$$

where (ii) is as in (C.16), and (i) can be seen by observing that we can generate a long sequence of rewards from each arm before the process starts, from the fact that $\hat{r}_j - \mu_{A_j}, \hat{r}_i - \mu_{A_i}$ are uncorrelated for all $i \neq j$ and since $\mathbb{E}[|r_t - \mu_{A_t}|^2 | A_t] \leq \sigma^2$.

We conclude that there is a universal constant C such that

$$\hat{B}(T) \leq 3.4 + \frac{C}{T}\sum_{i=1}^{k} \log(1+\frac{|\mu_i|}{\sigma}) + \frac{C}{\sqrt{T}}. \tag{C.21}$$

The fact that $\lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} B_t \leq 3.4$ almost surely, can be seen using the strong law of large numbers by observing that we can generate a long sequence of rewards from each arm before the process starts, the number of arms is finite, and if $\lim_{T\to\infty} T_i(T) < \infty$ then the contribution of arm i in the number of bits decays to zero almost surely as $T \to \infty$.

• stochastic linear bandits ($\hat{\mu}(t) = \langle \theta_t, A_t \rangle$):

The results follow directly from (C.8), (C.10), (C.11) and choice of $\hat{\mu}(t)$.

For the case where $\epsilon \neq 1$, it is easy to see that for small values of ϵ, the number of transmitted bits increases by $2\log(\frac{1}{\epsilon})$ bits. This can be further decreased to $\log(\frac{1}{\epsilon}) + \log(\log(\frac{1}{\epsilon}))$ bits using the encoding in Section 4.5. □

C.3 Proof of Theorem 12: High Probability Bound on the Number of Bits

From Section 4.5, we have that

$$\begin{aligned}B_t \leq 3 &+ \mathbf{1}[\frac{r_t}{M_t} - \lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor > 3] + \mathbf{1}[\lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor - \frac{r_t}{M_t} > 2] \\ &+ \mathbf{1}[\frac{r_t}{M_t} - \lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor > 4]\left(\lceil\log(\frac{r_t}{M_t} - \lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor - 3)\rceil \right.\\ &\left. + \lceil\log(\log(\frac{r_t}{M_t} - \lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor - 3))\rceil\right)\end{aligned}$$

$$+\mathbf{1}[\lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor-\frac{r_t}{M_t}>3]\left(\lceil\log(\lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor-\frac{r_t}{M_t}-2)\rceil\right.$$
$$\left.+\lceil\log(\log(\lfloor\frac{\hat{\mu}(t)}{M_t}\rfloor-\frac{r_t}{M_t}-2))\rceil\right)$$
$$\leq 4+\log(\frac{\hat{\mu}(t)}{\sigma}-\frac{r_t}{\sigma}-2)+\log(\log(\frac{\hat{\mu}(t)}{\sigma}-\frac{r_t}{\sigma}-2)). \tag{C.22}$$

Let the event G be that $\forall t \in \{1, ..., T\} : |r_t - \mu_{A_t}| \leq \sigma\sqrt{4\log(T)}$. From the subgaussian assumption and applying the union bound we have that

$$\mathbb{P}[G] > 1-\sum_{t=1}^{T} e^{-2\log(T)}. \tag{C.23}$$

We have that if G holds then for t with $T_t(A_t) > 0$, we have that $|\hat{\mu}(t) - \mu_{A_t}| \leq \sigma, |r_t - \mu_{A_t}| \leq \sigma$. Hence, $|\hat{\mu}(t) - r_t| \leq 2\sigma$. Substituting in (C.22), we get the desired result.

C.4 Proofs of Lemma 4 and Theorem 13: The Lower Bound

C.4.1 Proof of Lemma 4

Proof. To simplify notation, we omit the time index t and only mention it when it is necessary.

Let P, P' denote reward distributions with means μ_1 and μ_2, respectively. We have that, for any given quantizer Q, either:
Case 1: $\forall\ P, P'$ with $\mu_1 \neq \mu_2$, we have that $\mathbb{E}_P[Q(r)] \neq \mathbb{E}_{P'}[Q(r)]$; or
Case 2: $\exists\ P, P'$ with $\mu_1 \neq \mu_2$, and $\mathbb{E}_P[Q(r)] = \mathbb{E}_{P'}[Q(r)]$.

We will first show that any quantizer Q satisfying **Case 1** must saisfy $\mathbb{E}[Q(r)|r] = c_1 r + c_2$ for some constants c_1, c_2. To do so, we first construct distributions P and P' as follows. Let $\{x_i, p_i, p'_i\}_{i=1}^3$ be real values such that $x_1 \neq x_2, \sum_{i=1}^3 p_i = \sum_{i=1}^3 p'_i = 1$ and $p_i,\ p'_i \geq 0,\ \forall\ i \in \{1, 2, 3\}$. We design P to be the distribution of a random variable that takes the value x_i with probability p_i, and P' be the distribution of a random variable that takes the value x_i with probability p'_i for $i = 1, 2, 3$.

For Case 1, it is necessary that $\mathbb{E}_P[Q(r)] = \mathbb{E}_{P'}[Q(r)]$ *only if* $\sum_{i=1}^3 p_i x_i = \sum_{i=1}^3 p'_i x_i$. Or equivalently,

$$\sum_{i=1}^{3}(p_i - p_i')\mathbb{E}[Q(r)|r = x_i] = 0 \textit{ only if } \sum_{i=1}^{3}(p_i - p_i')x_i = 0. \tag{C.24}$$

This implies that the right null space of the matrix

$$\mathbf{E} = \begin{bmatrix} \mathbb{E}[Q(r)|r = x_1] & \mathbb{E}[Q(r)|r = x_2] & \mathbb{E}[Q(r)|r = x_3] \\ 1 & 1 & 1 \end{bmatrix}$$

is subset of the right null space of the matrix

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & x_3 \\ 1 & 1 & 1 \end{bmatrix}$$

(note that $\sum_i (p_i - p_i') = 0$). This is because for any vector $\mathbf{a}$ in the nullspace of $\mathbf{E}$, there exist vectors $\mathbf{p}, \mathbf{p}'$ such that $\mathbf{p}, \mathbf{p}' \geq 0$, $\mathbf{1}^\top \mathbf{p} = \mathbf{1}^\top \mathbf{p}' = 1$, and $\mathbf{a} = c(\mathbf{p} - \mathbf{p}')$ for some constant c; in particular, $\mathbf{p} = \frac{\mathbf{a}^+}{\mathbf{1}^\top \mathbf{a}^+}, \mathbf{p}' = \frac{|\mathbf{a}^-|}{\mathbf{1}^\top |\mathbf{a}^-|}$, where $\mathbf{a}^+$ is the same as $\mathbf{a}$ with the negative entries replaced by zeros, while in $\mathbf{a}^-$ the positive entries of $\mathbf{a}$ are replaced by zeros. Note that $\mathbf{1}^\top \mathbf{a}^+ = \mathbf{1}^T |\mathbf{a}^-|$ since $\mathbf{a}$ is in the right null space of $\mathbf{E}$, hence, $\mathbf{1}^\top \mathbf{a} = 0$. Thus, by (C.24), the same vector $\mathbf{a}$ also belongs in the nullspace of $\mathbf{X}$.

We also observe that since $x_1 \neq x_2$ and $\mathbb{E}[Q(r)|r = x_1] \neq \mathbb{E}[Q(r)|r = x_2]$ (as we assumed in **Case 1**); hence the ranks of $\mathbf{E}$ and $\mathbf{X}$ equal to 2. Therefore the dimension of the right null space of each of the matrices $\mathbf{E}, \mathbf{X}$ is exactly one. This, together with the fact that the right null space of $\mathbf{E}$ is a subset of the right null space of $\mathbf{X}$, imply that the right null spaces of these two matrices are *exactly the same* (and one-dimensional). We note that the right null space of $\mathbf{X}$ includes the vector

$$\mathbf{a} = \begin{bmatrix} \frac{x_3 + x_2}{x_1 - x_2} \\ \frac{x_3 + x_1}{x_2 - x_1} \\ 1 \end{bmatrix}.$$

Hence, we have that $\mathbf{Ea} = \mathbf{0}$ which implies that (from the first row of $\mathbf{Ea} = \mathbf{0}$)

$$\mathbb{E}[Q(r)|r = x_3] = \left(\frac{\mathbb{E}[Q(r)|r = x_1] - \mathbb{E}[Q(r)|r = x_2]}{x_2 - x_1} \right) x_3$$

$$+\frac{x_2\mathbb{E}[Q(r)|r=x_1]-x_1\mathbb{E}[Q(r)|r=x_2]}{x_2-x_1}$$

As x_3 was arbitrary, we have that, for all $x \in \mathbb{R}$

$$\mathbb{E}[Q(r)|r=x]=c_1x+c_2,$$

where $c_1=\frac{\mathbb{E}[Q(r)|r=x_1]-\mathbb{E}[Q(r)|r=x_2]}{x_1-x_2}$, $c_2=\frac{x_2\mathbb{E}[Q(r)|r=x_1]-x_1\mathbb{E}[Q(r)|r=x_2]}{x_1-x_2}$. This completes the proof for Case 1.

For **Case 2**, if we consider a MAB instance with two arms with distributions P, P' that witness the property in **Case 2**, then even if we have infinite samples from the quantization scheme we cannot achieve better than $O(|\mu_1-\mu_2|T)$ regret. $\square$

C.4.2 Proof of Theorem 13

Proof. To simplify notation, we omit the time index t and only mention it when it is necessary. Normalizing the rewards by σ, it suffices to consider the case where $\sigma=1$.

We first show that it suffices to consider schemes with deterministic quantization levels. Let us consider a quantizer Q with encoder $\mathcal{E}:\mathbb{R}\to\mathbb{N}$ and decoder $D:\mathbb{N}\to\mathbb{R}$, where $\mathcal{E}, D$ can both be random. We note that as D is allowed to be random, the set of quantization levels is now random. Let us consider a new decoder D' defined as

$$D'(i)=\mathbb{E}[D(i)]. \tag{C.25}$$

We now consider the quantizer Q' defined by $\mathcal{E}, D'$ as an encoder-decoder pair. We note that the decoder D' is a deterministic function, hence, the set of quantization levels for the quantizer Q' is deterministic. We will show that: (a) $\mathbb{E}[Q(r_t)|r_t]=\mathbb{E}[Q'(r_t)|r_t]$ and (b) if Q results in sub-Gaussian quantized rewards conditioned on r_t, then Q' also results in sub-Gaussian quantized rewards conditioned on r_t with the same sub-Gaussian parameter as Q. Properties (a) and (b) will allow us to switch D with D' in the rest of our proofs without affecting the encoder $\mathcal{E}$ (hence without affecting the number of bits). To show $\mathbb{E}[Q(r_t)|r_t]=\mathbb{E}[Q'(r_t)|r_t]$, we observe that

$$\mathbb{E}[Q(r_t)|r_t]=\mathbb{E}[D(\mathcal{E}(r_t))|r_t]=\mathbb{E}[\mathbb{E}[D(i)|r_t,\mathcal{E}(r_t)=i]|r_t]$$

$$= \mathbb{E}[D'(\mathcal{E}(r_t))|r_t] = \mathbb{E}[Q'(r_t)|r_t]. \tag{C.26}$$

To show the second property we observe that

$$\begin{aligned}
&\mathbb{E}[\exp(\lambda(Q(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t, \mathcal{E}(r_t)] \\
&\overset{(i)}{=} \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t, \mathcal{E}(r_t)] \\
&\quad \mathbb{E}[\exp(\lambda(Q(r_t) - Q'(r_t)))|r_t, \mathcal{E}(r_t)] \\
&\overset{(ii)}{\geq} \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t, \mathcal{E}(r_t)] \\
&\quad \exp(\lambda\mathbb{E}[(Q(r_t) - Q'(r_t))|r_t, \mathcal{E}(r_t)]) \\
&= \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t, \mathcal{E}(r_t)] \\
&\quad \exp(\lambda\mathbb{E}[(D(\mathcal{E}(r_t)) - D'(\mathcal{E}(r_t)))|r_t, \mathcal{E}(r_t)]) \\
&\overset{(iii)}{=} \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t, \mathcal{E}(r_t)] \\
&\overset{(iv)}{=} \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q'(r_t)|r_t]))|r_t, \mathcal{E}(r_t)]
\end{aligned} \tag{C.27}$$

where (i) follows by the fact that $Q'(r_t) = D'(\mathcal{E}(r_t))$ is a deterministic function of $\mathcal{E}(r_t)$, (ii) follows by Jensen's inequality and non-negativity of the $\exp$ function, (iii) follows by definition of D', and (iv) follows from (C.26). By taking the conditional expectation given r_t of both sides in (C.27) we get that

$$\begin{aligned}
&\mathbb{E}[\exp(\lambda(Q(r_t) - \mathbb{E}[Q(r_t)|r_t]))|r_t] \\
&\qquad \geq \mathbb{E}[\exp(\lambda(Q'(r_t) - \mathbb{E}[Q'(r_t)|r_t]))|r_t].
\end{aligned} \tag{C.28}$$

Proof of Statetment 1 &2 in Theorem 13 To prove 1,2, we consider the following distribution that takes values on $2z \forall z \in \mathbb{Z}$ with

$$\mathbb{P}[r_t = 2z] = \begin{cases} \mathbb{P}[\mathcal{N}(0,(4\sigma)^2) \in [2z, 2(z+1)]] & \text{if } z > 0 \\ \mathbb{P}[\mathcal{N}(0,(4\sigma)^2) \in [2(z-1), 2z]] & \text{if } z < 0 \\ \mathbb{P}[\mathcal{N}(0,(4\sigma)^2) \in [-2, 2]] & \text{if } z = 0, \end{cases} \tag{C.29}$$

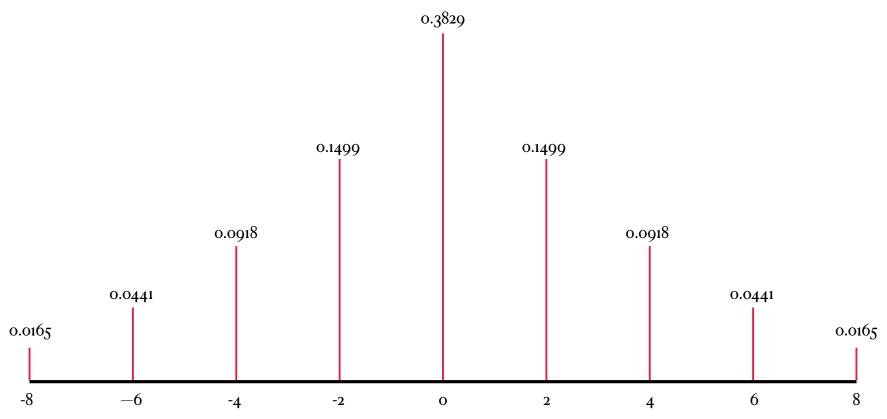


Figure C.1: Illustration of reward distribution.

where $\mathcal{N}(0,(4\sigma)^2)$ is a random variable with Gaussian distribution with zero mean and standard deviation 4σ. An illustration of the distribution is depicted in Fig. C.1.

By construction of the distribution, we have that r_t is $(4\sigma)^2$-subGaussian, since every value in the Gaussian distribution is mapped to one that is closer to the mean in the distribution of r_t. We next prove 1. Suppose towards a contradiction that $\exists b, t$ such that $\mathbb{P}[B_T \leq b] = 1 \forall T > t$. Pick T arbitrary large, we have that b can describe at most 2^b quantization levels. We note that the maximum distance between any consecutive quantization levels cannot exceed 4, lest there is a reward r, that is in the middle of the two quantization levels, mapped to $\hat{r}$ with $|\hat{r} - r| \geq 2$ almost surely which violates the fact that $\mathbb{P}[|\hat{r} - r| \geq 2 | r = z] \leq \exp(-\frac{2^2}{2(\sigma/2)^2}) < 1$ for some z given by the subGaussian concentration of assumption (ii). Hence, either the interval $(-\infty, -4(2^b+1)]$ or the interval $[4(2^b+1), \infty)$ will have no quantization levels. We assume without loss of generality that the interval $[4(2^b+1), \infty)$ has no quantization levels. Hence, all the values in that interval will be mapped to values in $(-\infty, 4(2^b+1))$. We notice that for the described reward, the interval $(-\infty, 4(2^b+1))$ has non-zero probability. This contradicts assumption (i) (unbiasedness).

We next prove 2. Let G_t be the event that $|Q(r_t) - r_t| \leq 1$. We observe that by assumption (ii), since $\hat{r}_t - r_t$ is $(\frac{\sigma}{2})^2$-subGaussian, we have that $\mathbb{P}[G_t | r_t] \geq 1 - 2e^{-2} \geq 0.729$. Let us consider the intervals of the form $[2i-1, 2i+1] \forall i \in \mathbb{Z}$. As we proved above, it is sufficient to consider quantization schemes with deterministic quantization levels. Let $\ell_i(t)$ be the minimum length of a quantization level in the interval $[2i-1, 2i+1]$. We have that

$$\begin{aligned}
\mathbb{E}[B_t] &= \sum_{i\in\mathbb{Z}} \mathbb{P}[r_t = 2i]\mathbb{E}[B_t|r_t = 2i] \\
&\geq \sum_{i\in\mathbb{Z}} \mathbb{P}[r_t = 2i]\mathbb{P}[G_t|r_t = 2i]\mathbb{E}[B_t|r_t = 2i, G_t] \\
&\geq \sum_{i\in\mathbb{Z}} \mathbb{P}[r_t = 2i]\mathbb{P}[G_t|r_t = 2i]\ell_i \\
&\geq \sum_{|i|\leq 4, i\in\mathbb{Z}} \mathbb{P}[r_t = i]\mathbb{P}[G_t|r_t = 2i]\ell_i \\
&\geq 0.729 \min_{\{\ell_i\}} \sum_{|i|\leq 4, i\in\mathbb{Z}} \mathbb{P}[r_t = 2i]\ell_i. \qquad \text{(C.30)}
\end{aligned}$$

We also notice that as the code is prefix free, then if we restrict the code over a subset of quantization levels, it is still prefix free. It follows that the lengths ℓ_i need to satisfy the tree inequality [Cov99], namely, $\sum_{|i|\leq 4, i\in\mathcal{Z}} 2^{-\ell_i} \leq 1$. Hence, we have that the code that minimizes (C.30) is Huffman code [Cov99]. Performing Huffman code with the weights in (C.30) gives the following code lengths for $\ell_{-4}, ..., \ell_4$ respectively: $6, 5, 4, 3, 1, 3, 4, 4, 6$. Substituting in (C.30) gives $\mathbb{E}[B_t] \geq 0.729 \min_{\{\ell_i\}} \sum_{|i|\leq 4, i\in\mathbb{Z}} \mathbb{P}[r_t = i]\ell_i \geq 1.9$. $\square$

C.5 Proofs of Corollaries 1, 2

The expected regret bounds follow directly from Theorem 11. To bound the average number of bits used for the avg-pt, we only need to bound the decay rate of $\frac{1}{T}\sum_{t=1}^{T-1} \frac{R_t}{\sigma t}$.

Corollary 1:

From Theorem 11 and [ACF02], we have that for $QuBan$ with UCB, there is a constant C such that $R_T \leq C\sigma\sqrt{kT\log(T)}$. Then,

$$\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T-1} \frac{R_t}{\sigma t} &\leq C\frac{1}{T}\sum_{t=1}^{T} \frac{\sqrt{kt\log(t)}}{t} \\
&\leq \frac{C\sqrt{k\log(T)}}{T}\sum_{t=1}^{T} \frac{1}{\sqrt{t}}
\end{aligned}$$

$$\leq \frac{C\sqrt{k\log(T)}}{T}(1 + \int_{t=1}^{T} \frac{1}{\sqrt{t}})$$
$$\leq C\sqrt{k\log(T)T}. \tag{C.31}$$

Corollary 2:

From Theorem 11 and [ACF02], we have that for $QuBan$ with ϵ-greedy, there is a constant C such that $R_T \leq C\sigma k \log(1 + \frac{T}{k})$. Then,

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T-1}\frac{R_t}{\sigma t} &\leq \frac{Ck}{T}\sum_{t=1}^{T-1}\frac{\log(1+t)}{t}\\ &\leq \frac{Ck\log(1+T)}{T}\sum_{t=1}^{T-1}\frac{1}{t}\\ &\leq \frac{Ck\log(1+T)}{T}(1+\int_{1}^{T-1}\frac{1}{t})\\ &\leq \frac{Ck(\log(1+T))^2}{T}. \end{aligned} \tag{C.32}$$

C.6 Proof of Corollary 3

We observe that the LinUCB parameters, β_t, can be chosen such that $\max_{t\in\{1,\ldots,T\}} \sup_{a\in\mathcal{A}_t}\langle\theta_t - \theta_*, a\rangle \leq \sqrt{\beta_T}$. Hence, by Cauchy–Schwarz we have that

$$\begin{aligned}\sum_{t=1}^{T}|\langle\theta_t-\theta_*, A_t\rangle| &\leq \sqrt{T\sum_{t=1}^{T}|\langle\theta_t-\theta_*, A_t\rangle|^2}\\ &\leq \sqrt{T\sum_{t=1}^{T}\min\{\beta_T, \langle\theta_t-\theta_*, A_t\rangle^2\}}. \end{aligned} \tag{C.33}$$

The proof of the expected regret and average number of bits bounds then follows as in [LS20, Theorem 19.2] using Theorem 11.

APPENDIX D

Appendix for Chapter 5: The Common Information Dimension

D.1 Proofs of lemmas in Theorem 14

Lemma 6. Let $X = [X_1, X_2, ..., X_n]$ be a d_X-dimensional random vector, $d_X = \sum_{i=1}^n d_{X_i}$, with zero mean and covariance matrix $\mathbf{\Sigma}$. For any vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{d_X}$, we have that

$$\boldsymbol{a}^\top X = \boldsymbol{b}^\top X \text{ almost surely, if and only if } \boldsymbol{a}^\top \mathbf{\Sigma} = \boldsymbol{b}^\top \mathbf{\Sigma}.$$

Proof. If $\boldsymbol{a}^\top X = \boldsymbol{b}^\top X$ almost surely, the multiplying both sides by $X^\top$ and taking expectation gives $\boldsymbol{a}^\top \mathbf{\Sigma} = \boldsymbol{b}^\top \mathbf{\Sigma}$. It remains to show the other direction; namely, if $\boldsymbol{a}^\top \mathbf{\Sigma} = \boldsymbol{b}^\top \mathbf{\Sigma}$, then $\boldsymbol{a}^\top X = \boldsymbol{b}^\top X$ almost surely. We have that the second moment of $(\boldsymbol{a}^\top - \boldsymbol{b}^\top)X$ is given by

$$\begin{aligned} \mathbb{E}[((\boldsymbol{a}-\boldsymbol{b})^\top X)^2] &= \mathbb{E}[(\boldsymbol{a}-\boldsymbol{b})^\top X X^\top (\boldsymbol{a}-\boldsymbol{b})] \\ &= (\boldsymbol{a}-\boldsymbol{b})^\top \mathbf{\Sigma} (\boldsymbol{a}-\boldsymbol{b}) = 0. \end{aligned} \tag{D.1}$$

It follows that $(\boldsymbol{a}-\boldsymbol{b})^\top X = 0$ almost surely. □

Corollary 6. *There is a subset $I \subseteq \{1, ..., d_X\}$ such that $|I| = \text{rank}(\mathbf{\Sigma}_X)$, and $X_I \perp\!\!\!\perp Y|(V, W)$ if and only if $X \perp\!\!\!\perp Y|(V, W)$.*

Proof. By Lemma 6, there is a subset $I \subseteq \{1, ..., d_X\}$ such that $|I| = \text{rank}(\mathbf{\Sigma}_X)$, and $X = \boldsymbol{B} X_I$ for some $\boldsymbol{B} \in \mathbb{R}^{d_x \times |I|}$. Then we have that $X_I \perp\!\!\!\perp Y|(V, W)$ if and only if $X \perp\!\!\!\perp Y|(V, W)$. □

Lemma 7. Let $[X, Y]$ be a random vector with covariance matrix $\mathbf{\Sigma}$ and $\boldsymbol{N} = \begin{bmatrix} \boldsymbol{N}_X & -\boldsymbol{N}_Y \end{bmatrix}$ be a basis for the null space of $\mathbf{\Sigma}$, where $\boldsymbol{N} \in \mathbb{R}^{r(\boldsymbol{N}) \times (d_X + d_Y)}$, $\boldsymbol{N}_X \in \mathbb{R}^{r(\boldsymbol{N}) \times d_X}$, and $\boldsymbol{N}_Y \in \mathbb{R}^{r(\boldsymbol{N}) \times d_Y}$.

If X and Y are non-singular (i.e., $\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y$ are full-rank), we have that

$$\text{rank}(\boldsymbol{N}_X) = \text{rank}(\boldsymbol{N}_Y) = \text{rank}(\boldsymbol{N}). \tag{5.19}$$

Proof. Suppose towards a contradiction that $\boldsymbol{N}_X$ is not full-row-rank. Hence, there exists $\boldsymbol{b} \neq \mathbf{0}$ such that $\boldsymbol{b}^\top \boldsymbol{N}_X = \mathbf{0}$. As $\boldsymbol{N}$ is full-row-rank, we have that $\boldsymbol{b}^\top \boldsymbol{N}_Y \neq \mathbf{0}$. From equation (5.18) we have that $\boldsymbol{b}^\top \boldsymbol{N}_Y Y = \boldsymbol{b}^\top \boldsymbol{N}_X X$. Hence, we have that $\boldsymbol{b}^\top \boldsymbol{N}_Y Y = \mathbf{0}$, and $\boldsymbol{b}^\top \boldsymbol{N}_Y \neq \mathbf{0}$. As a result, by Lemma 6, we have $\boldsymbol{b}^\top \boldsymbol{N}_Y \mathbf{\Sigma}_Y = \mathbf{0}, \boldsymbol{b}^\top \boldsymbol{N}_Y \neq \mathbf{0}$, which contradicts our assumption that $\mathbf{\Sigma}_Y$ is full-rank. Therefore, it has to hold that $\text{rank}(\boldsymbol{N}_X) = \text{rank}(\boldsymbol{N}_Y) = \text{rank}(\boldsymbol{N})$. □

Lemma 8. Let $\boldsymbol{M}_X \in \mathbb{R}^{d_X \times d_X}, \boldsymbol{M}_Y \in \mathbb{R}^{d_Y \times d_Y}$ be full-rank matrices, and X, Y, V, W be random vectors of dimension d_X, d_Y, d_V, and d_W respectively. We have that

$$\boldsymbol{M}_X X \perp\!\!\!\perp \boldsymbol{M}_Y Y | (V, W) \text{ if and only if } X \perp\!\!\!\perp Y | (V, W).$$

Proof. Since $\boldsymbol{M}_X$ and $\boldsymbol{M}_Y$ are full-rank, the corresponding linear mappings are one-to-one. If X, Y are conditionally independent given (V, W), then for any sets $\mathcal{S}_X, \mathcal{S}_Y$, we have that

$$\begin{aligned}
&\mathbb{P}[\boldsymbol{M}_X X \in \mathcal{S}_X, \boldsymbol{M}_Y Y \in \mathcal{S}_Y | (V, W)] && \text{(D.2)}\\
&= \mathbb{P}[X \in \boldsymbol{M}_X^{-1} \mathcal{S}_X, Y \in \boldsymbol{M}_Y^{-1} \mathcal{S}_Y | (V, W)] \\
&= \mathbb{P}[X \in \boldsymbol{M}_X^{-1} \mathcal{S}_X | (V, W)] \mathbb{P}[Y \in \boldsymbol{M}_Y^{-1} \mathcal{S}_Y | (V, W)] \\
&= \mathbb{P}[\boldsymbol{M}_X X \in \mathcal{S}_X | (V, W)] \mathbb{P}[\boldsymbol{M}_Y Y \in \mathcal{S}_Y | (V, W)], && \text{(D.3)}
\end{aligned}$$

where $\boldsymbol{M}_X^{-1} \mathcal{S}_X = \{\boldsymbol{M}_X^{-1} x | x \in \mathcal{S}_X\}$ and $\boldsymbol{M}_Y^{-1} \mathcal{S}_Y$ is defined similarly. This shows that $\boldsymbol{M}_X X$ and $\boldsymbol{M}_Y Y$ are conditionally independent given (V, W). The proof of the other direction that $\boldsymbol{M}_X X \perp\!\!\!\perp \boldsymbol{M}_Y Y | (V, W)$ implies $X \perp\!\!\!\perp Y | (V, W)$ is similar. □

The following lemma is utilized in the proof of Lemma 9.

Lemma 19. Define $\boldsymbol{N}'_X$, $\boldsymbol{M}_Y$ as in (5.14) and (5.15). Then

$$\nexists\, \boldsymbol{a}, \boldsymbol{b} \text{ such that } \boldsymbol{a} \neq \mathbf{0} \text{ and } \boldsymbol{a}^\top \boldsymbol{N}'_X X = \boldsymbol{b}^\top \boldsymbol{M}_Y Y.$$

Proof. Suppose towards a contradiction that $\exists\ \boldsymbol{a}, \boldsymbol{b}$ such that $\boldsymbol{a} \neq \boldsymbol{0}$ and $\boldsymbol{a}^\top \boldsymbol{N}'_X X = \boldsymbol{b}^\top \boldsymbol{M}_Y Y$. Then, we can extend $\boldsymbol{N}$ by adding the following row $\begin{bmatrix}\boldsymbol{a}^\top \boldsymbol{N}'_X & -\boldsymbol{b}^\top \boldsymbol{M}_Y\end{bmatrix}$, which is linearly independent on rows of $\boldsymbol{N}$ because $\boldsymbol{N}'_X$ is the complementary row space of $\boldsymbol{N}_X$ and $\boldsymbol{N} = \begin{bmatrix}\boldsymbol{N}_X & -\boldsymbol{N}_Y\end{bmatrix}$. This contradicts the fact that $\boldsymbol{N}$ is a basis for the null space of $\boldsymbol{\Sigma}_{X,Y}$. □

Lemma 9. Let $\boldsymbol{N}_X, \boldsymbol{N}_Y, \boldsymbol{N}'_X, \boldsymbol{N}'_Y$ be as defined in (5.13) and (5.14). Conditioned on $\boldsymbol{N}_X X$, we have that $\begin{bmatrix}(\boldsymbol{N}'_X X)^\top & (\boldsymbol{N}'_Y Y)^\top\end{bmatrix}^\top$ has full-rank covariance matrix.

Proof. Suppose towards a contradiction that there is $\boldsymbol{a} \neq \boldsymbol{0}$ such that $\boldsymbol{a}^\top \boldsymbol{\Sigma}'_{|\boldsymbol{N}_X X} = \boldsymbol{0}$, where $\boldsymbol{\Sigma}'_{|\boldsymbol{N}_X X}$ is the covariance matrix of $\begin{bmatrix}\boldsymbol{N}'_X X \\ \boldsymbol{N}'_Y Y\end{bmatrix}$ conditioned on $\boldsymbol{N}_X X$. Since the mean of $\begin{bmatrix}\boldsymbol{N}'_X X \\ \boldsymbol{N}'_Y Y\end{bmatrix}$ conditioned on $\boldsymbol{N}_X X$ is a linear function of $\boldsymbol{N}_X X$ (say the conditional mean is $\boldsymbol{B}\boldsymbol{N}_X X$), then, by Lemma 6 we have that

$$\boldsymbol{a}^\top (\begin{bmatrix}\boldsymbol{N}'_X X \\ \boldsymbol{N}'_Y Y\end{bmatrix} - \boldsymbol{B}\boldsymbol{N}_X X) = \boldsymbol{0} \text{ almost surely.}$$

Let us partition $\boldsymbol{a}$ as $\boldsymbol{a} = \begin{bmatrix}\boldsymbol{a}_X \\ \boldsymbol{a}_Y\end{bmatrix}$. Using the fact that $\boldsymbol{N}_X X = \boldsymbol{N}_Y Y$ we get that

$$\boldsymbol{a}_X^\top \boldsymbol{N}'_X X = \boldsymbol{b}^\top \begin{bmatrix}\boldsymbol{N}_Y Y \\ \boldsymbol{N}'_Y Y\end{bmatrix} = \boldsymbol{b}^\top \boldsymbol{M}_Y Y,$$

where $\boldsymbol{b} = \begin{bmatrix}\boldsymbol{B}^\top \boldsymbol{a} \\ -\boldsymbol{a}_Y\end{bmatrix}$, which contradicts Lemma 19. □

D.2 Proof of Theorem 15

Theorem 15. *Let $X = [X_1, \cdots, X_n]$ be a jointly Gaussian random vector. The common information dimension between $X_1, \cdots, X_n$ with respect to the class of linear functions is*

$$d(X_1, \cdots, X_n) = \sum_{i=1}^{n} \mathit{rank}(\boldsymbol{\Sigma}_{-i}) - (n-1)\mathit{rank}(\boldsymbol{\Sigma}),$$

Table D.1: Table of notation for proof of Theorem 15

Notation	Definition	
$\boldsymbol{A}_i$	basis of the row space of $\boldsymbol{\Sigma}_{i\|1:i-1}$	(D.4)
$\boldsymbol{B}_i$	basis of the row space of $\boldsymbol{\Sigma}_{i+1:n\|1:i-1}$	(D.5)
U_i	$U_i = \boldsymbol{A}_i X_i$	(D.6)
Y_i	$Y_i = \boldsymbol{B}_i [X_{i+1}^\top \; \cdots \; X_n^\top]^\top$	(D.7)
$\tilde{\boldsymbol{N}}_i$	basis of the null space of $\boldsymbol{\Sigma}_{U_i, Y_i \| X_1, \cdots, X_{i-1}}$, $\tilde{\boldsymbol{N}}_i \in \mathbb{R}^{r(\tilde{\boldsymbol{N}}_i) \times (r(U_i) + r(Y_i))}$	(D.8)
$\boldsymbol{N}_i, \bar{\boldsymbol{N}}_i$	partition $\tilde{\boldsymbol{N}}_i$ as $[\boldsymbol{N}_i \; \bar{\boldsymbol{N}}_i]$ $\boldsymbol{N}_i \in \mathbb{R}^{r(\tilde{\boldsymbol{N}}_i) \times r(U_i)}$ $\bar{\boldsymbol{N}}_i \in \mathbb{R}^{r(\tilde{\boldsymbol{N}}_i) \times r(Y_i)}$	(D.9)
Z_i	$Z_i = \boldsymbol{N}_i U_i$	(D.10)
$\boldsymbol{C}_i$	basis of the row space of $\boldsymbol{\Sigma}_{X_i \| Z_1, \cdots, Z_n}$	(D.11)
T_i	$T_i = \boldsymbol{C}_i X_i$	(D.12)

where $\boldsymbol{\Sigma}$ is the covariance matrix of X, and $\boldsymbol{\Sigma}_{-i}$ is the covariance matrix of the random vector $[X_1, \cdots, X_{i-1}, X_{i+1}, \cdots, X_n]$. Moreover, a W that satisfies the minimum in (5.8) *is given by Algorithm 9.*

Proof. Consider the quantities defined in Table D.1; the proof logic proceeds as follows. We sequentially extract the common information between the X_i by constructing a sequence of variables Z_i, where Z_i contains the amount of information that X_i contains about $X_{i+1}, \cdots, X_n$ and that $X_1, \cdots, X_{i-1}$ does not contain. We recall from the proof of Theorem 14 that: (i) to remove the parts of X_i that can be determined by $X_1, \cdots, X_{i-1}$, we can consider $\boldsymbol{A}_i X_i$, where $\boldsymbol{A}_i$ is a basis for the row space of $\boldsymbol{\Sigma}_{X_i | X_1, \cdots, X_{i-1}}$ (Lemma 6); and (ii) to find the common information between two non-singular Gaussian random variables X, Y, we need to consider the null space of $\boldsymbol{\Sigma}_{XY}$ ((5.20) and (5.26)). We utilize this in the following to build Z:

• To get the common information between X_1 and $X_2, \cdots, X_n$; we first remove the singular part of X_1 by linearly transforming X_1 using a matrix $\boldsymbol{A}_1$ that captures the non-singular part of X_1. We denote the resulting vector variable by U_1.

• Similarly, we remove the singular part of $[X_2, \cdots, X_n]$ using a matrix $\boldsymbol{B}_1$ to produce Y_1.

• We then argue (using (ii)) that the parts of $X_2, \cdots, X_n$ that can be determined from X_1 (common information), denoted as Z_1, can be obtained from the null space of $\boldsymbol{\Sigma}_{U_1 Y_1}$.

• To obtain Z_2 which contains the amount of information that X_2 contains about $X_3, \cdots, X_n$ that X_1 does not contain: we first eliminate the parts of X_2 that can be obtained from X_1. We argue (as described in (i)) using Lemma 6, that the vectors in the null space of $\boldsymbol{\Sigma}_{X_2|X_1}$ are precisely the linear combinations of X_2 that can be obtained from X_1. Hence, we remove the parts of X_2 that can be obtained from X_1 using a matrix $\boldsymbol{A}_2$ that captures the non-singular part of the matrix $\boldsymbol{\Sigma}_{X_2|X_1}$, to obtain U_2.

• Similarly, we remove the parts of $[X_3, \cdots, X_n]$ that can be obtained from X_1 using a matrix $\boldsymbol{B}_2$ to produce Y_2.

• Similar to Z_1, the desired Z_2 can then be obtain from the null space of $\boldsymbol{\Sigma}_{U_2 Y_2}$.

More generally, to obtain Z_i which contains the amount of information that X_i contains about $X_{i+1}, \cdots, X_n$ that $X_1, \cdots, X_{i-1}$ does not contain: we first eliminate the parts of X_i that can be obtained from $X_1, \cdots, X_{i-1}$. This is done using a matrix $\boldsymbol{A}_i$ that captures the non-singular part of the matrix $\boldsymbol{\Sigma}_{X_i|X_1,\cdots,X_{i-1}}$, to obtain U_i. Similarly, we remove the parts of $[X_{i+1}, \cdots, X_n]$ that can be obtained from $X_1, \cdots, X_{i-1}$ using a matrix $\boldsymbol{B}_i$ to produce Y_i. Then, Z_i can be obtained from the null space of $\boldsymbol{\Sigma}_{U_i Y_i}$. The method of constructing (V, W) that break the dependency between $X_1, \cdots, X_n$, where $H(V) < \infty$ and W has minimum dimension is summarized in Algorithm 9. The main part of our proof shows that $Z = [Z_1, \cdots, Z_n]$ is sufficient to break the dependency between $X_1, \cdots, X_n$ up to finite randomness (Lemma 20), and that Z is also a necessary part of every W that breaks the dependency up to finite randomness and hence $d_W \geq d_Z$ (Lemma 21).

We next formally prove our result. We state and prove two properties of the quantities in Table D.1. The first property is that the conditional covariance matrix $\mathbf{\Sigma}_{T_1,\cdots,T_n|Z_1,\cdots,Z_n}$ is full-rank, where T_i, defined in (D.12), can be thought of as the part of X_i that cannot be expressed as a deterministic function of $Z_1, \cdots, Z_n$.

Lemma 20. The conditional covariance matrix of $T = [T_1, \cdots, T_n]$ conditioned on $Z = [Z_1, \cdots, Z_n]$, namely, $\mathbf{\Sigma}_{T|Z}$ is non-singular, where T_i is defined in (D.12) and Z_i is defined in (D.10).

Proof. Suppose towards a contradiction that there exists $\boldsymbol{a} \neq \mathbf{0}$ such that $\boldsymbol{a}^\top \mathbf{\Sigma}_{T|Z} = \mathbf{0}$. Let us partition $\boldsymbol{a}$ as $[\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n]$, where $\boldsymbol{a}_i \in \mathbb{R}^{d_{T_i}}$. Let j denote the largest index for which $\boldsymbol{a}_j \neq \mathbf{0}$. By Lemma 6, we have that

$$\boldsymbol{a}^\top T = \boldsymbol{a}'^\top Z \quad \text{almost surely,} \tag{D.13}$$

for some vector $\boldsymbol{a}'$. We recall that by definition of T_i, Z_i in (D.12) and (D.10), they are linear functions of X_i. Also, by definition of j as the largest index with $\boldsymbol{a}_j \neq \mathbf{0}$, we have that $T_{j+1}, \cdots, T_n$ do not appear in (D.13); only $Z_{j+1}, \cdots, Z_n$ appear. By expanding T_i, Z_i for $i < j$ as linear functions of X_i, we get that there exist vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3$ such that

$$\begin{aligned} &\boldsymbol{a}_j^\top T_j + \boldsymbol{c}_1^\top [Z_{j+1}^\top \ \cdots \ Z_n^\top]^\top \\ &= \boldsymbol{e}_1^\top X_j + \boldsymbol{e}_2^\top [X_{j+1}^\top \ \cdots \ X_n^\top]^\top \\ &= \boldsymbol{c}_2^\top X_{j-1} + \boldsymbol{c}_3^\top [X_1 \ \cdots \ X_{j-2}]^\top \text{ almost surely.} \end{aligned} \tag{D.14}$$

We want to show that X_{j-1} can be replaced by Z_{j-1} in the previous equation (after possibly changing $\boldsymbol{c}_2, \boldsymbol{c}_3$). In particular, we want to show that there exist vectors $\tilde{\boldsymbol{c}}_2, \tilde{\boldsymbol{c}}_3$ such that

$$\boldsymbol{a}_j^\top T_j + \boldsymbol{c}_1^\top [Z_{j+1}^\top \ \cdots \ Z_n^\top]^\top = \tilde{\boldsymbol{c}}_2^\top Z_{j-1} + \tilde{\boldsymbol{c}}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \text{ almost surely.} \tag{D.15}$$

We have three possible cases:
(i) $\boldsymbol{c}_2$ is in the left null space of $\mathbf{\Sigma}_{(j-1)|1:(j-2)}$: It follows from Lemma 6 that $\boldsymbol{c}_2^\top X_{j-1}$ is a linear function of $X_1, \cdots, X_{j-2}$. Hence, it can be replaced by a linear function of $X_1, \cdots, X_{j-2}$ in (D.14),

and (D.15) holds with $\tilde{\boldsymbol{c}}_2 = \mathbf{0}$.

(ii) $[\boldsymbol{e}_1 \ \boldsymbol{e}_2]$ is in the left null space of $\boldsymbol{\Sigma}_{j:n|1:(j-2)}$: It follows from Lemma 6 that $\boldsymbol{e}_1^\top X_j + \boldsymbol{e}_2^\top [X_{j+1}^\top \ \cdots \ X_n^\top]^\top$ is a linear function of $X_1, \cdots, X_{j-2}$. Hence, (D.15) holds again with $\tilde{\boldsymbol{c}}_2 = \mathbf{0}$.

(iii) $\boldsymbol{c}_2$ is in the row space of $\boldsymbol{\Sigma}_{(j-1)|1:(j-2)}$ and $[\boldsymbol{e}_1 \ \boldsymbol{e}_2]$ is in the row space of $\boldsymbol{\Sigma}_{j:n|1:(j-2)}$: by definition of $\boldsymbol{A}_{j-1}, \boldsymbol{B}_{j-1}$ in (D.4) and (D.5), it follows that $\boldsymbol{c}_2^\top = \bar{\boldsymbol{c}}_2^\top \boldsymbol{A}_{j-1}$ and $[\boldsymbol{e}_1^\top \ \boldsymbol{e}_2^\top]^\top = \bar{\boldsymbol{e}}^\top \boldsymbol{B}_{j-1}$ for some $\bar{\boldsymbol{c}}, \bar{\boldsymbol{e}}$. Thus

$$\begin{aligned}\bar{\boldsymbol{e}}^\top Y_{j-1} &= \boldsymbol{e}_1^\top X_j + \boldsymbol{e}_2^\top [X_{j+1}^\top \ \cdots \ X_n^\top]^\top \\ &= \bar{\boldsymbol{c}}_2^\top U_{j-1} + \boldsymbol{c}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \text{ almost surely.}\end{aligned} \tag{D.16}$$

By Lemma 6 it follows that $[-\bar{\boldsymbol{c}}_2^\top \ \bar{\boldsymbol{e}}^\top]$ is in the left null space of $\boldsymbol{\Sigma}_{U_{j-1},Y_{j-1}|X_1,\cdots,X_{j-2}}$. By definition of $\boldsymbol{N}_{j-1}$ in (D.9) it follows that $\bar{\boldsymbol{c}}_2$ is in the row space of $\boldsymbol{N}_{j-1}$. Hence, we have that there is $\bar{\boldsymbol{c}}_3$ such that

$$\begin{aligned}&\boldsymbol{e}_1^\top X_j + \boldsymbol{e}_2^\top [X_{j+1}^\top \ \cdots \ X_n^\top]^\top \\ &= \bar{\boldsymbol{c}}_3^\top \boldsymbol{N}_{j-1} U_{j-1} + \boldsymbol{c}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \\ &= \bar{\boldsymbol{c}}_3^\top Z_{j-1} + \boldsymbol{c}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \text{ almost surely,}\end{aligned} \tag{D.17}$$

where the last equality follows by the definition of $Z_{j-1} = \boldsymbol{N}_{j-1} U_{j-1}$ in (D.10). Substituting in (D.14) it follows that there are vectors $\tilde{\boldsymbol{c}}_2, \tilde{\boldsymbol{c}}_3$ such that

$$\boldsymbol{a}_j^\top T_j + \boldsymbol{c}_1^\top [Z_{j+1}^\top \ \cdots \ Z_n^\top]^\top = \tilde{\boldsymbol{c}}_2^\top Z_{j-1} + \tilde{\boldsymbol{c}}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \text{ almost surely.} \tag{D.18}$$

Equivalently, there are vectors $\tilde{\boldsymbol{c}}_2, \tilde{\boldsymbol{c}}_3$ such that

$$\boldsymbol{a}_j^\top T_j + \boldsymbol{c}_1^\top [Z_{j+1}^\top \ \cdots \ Z_n^\top]^\top - \tilde{\boldsymbol{c}}_2^\top Z_{j-1} = \tilde{\boldsymbol{c}}_3^\top [X_1^\top \ \cdots \ X_{j-2}^\top]^\top \text{ almost surely.} \tag{D.19}$$

Applying the same result $j-2$ times, it follows that $X_1 \cdots, X_{j-2}$ can be replaced by $Z_1 \cdots, Z_{j-2}$. Equivalently, there is a vector $\boldsymbol{c}$ such that

$$\boldsymbol{a}_j^\top T_j = \boldsymbol{c}^\top Z \quad \text{almost surely.} \tag{D.20}$$

This contradicts the fact that $T_j = \boldsymbol{C}_j X_j$ where $\boldsymbol{C}_j$ is a basis of the row space of $\boldsymbol{\Sigma}_{X_i|Z_1,\cdots,Z_n}$ (recall that $\boldsymbol{a}_j \neq \boldsymbol{0}$). This concludes the proof that the conditional covariance matrix of T conditioned on Z is non-singular. $\square$

By definition of $T_i = \boldsymbol{C}_i X_i$ in (D.12), where $\boldsymbol{C}_i$ is a basis for the row space of $\boldsymbol{\Sigma}_{X_i|Z_1,\cdots,Z_n}$, it follows that conditioned on $Z_1, \cdots, Z_n$, X_i can be obtained from T_i. Hence, conditioned on $Z_1, \cdots, Z_n$, breaking the dependency between $X_1, \cdots, X_n$ reduces to breaking the dependency between $T_1, \cdots, T_n$. From Lemma 20, we have that conditioned on Z, T is jointly Gaussian with full-rank covariance matrix. Hence, by the result in [LE17], there is a V_Z with $H(V_Z) < \infty$ such that $T_1 \perp\!\!\!\perp \cdots \perp\!\!\!\perp T_n|(Z, V_Z)$, and hence, $X_1 \perp\!\!\!\perp \cdots \perp\!\!\!\perp X_n|(Z, V_Z)$. Since the covariance matrix of T conditioned on Z does not depend on the value of Z and is only a function of the covariance matrix of Z, then V_Z can be chosen the same for all Z. Hence, we can refer to V_Z as V, which gives that $X_1 \perp\!\!\!\perp \cdots \perp\!\!\!\perp X_n|(Z, V)$. This shows that

$$d(X_1, \cdots, X_n) \leq \sum_{i=1}^{n} d_{Z_i}. \tag{D.21}$$

We next prove a second property that the quantities defined in Table D.1 satisfy, that $\boldsymbol{\Sigma}_Z$ is a full-rank covariance matrix.

Lemma 21. The covariance matrix $\boldsymbol{\Sigma}_Z$ of $Z = [Z_1, \cdots, Z_n]$, where Z_i is defined in (D.10) is full-rank.

Proof. Suppose towards a contradiction that $\boldsymbol{\Sigma}_Z$ is not full-rank and pick a vector $\boldsymbol{a} \neq \boldsymbol{0}$ in the null space of $\boldsymbol{\Sigma}_Z$. Partition $\boldsymbol{a}$ as $[\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n]$, where $\boldsymbol{a}_i \in \mathbb{R}^{d_{Z_i}}$. Let j be the largest index with $\boldsymbol{a}_j \neq \boldsymbol{0}$. By definition of Z_i in (D.10) as linear function of X_i, we get that there is a vector $\boldsymbol{b}$ such that

$$\boldsymbol{a}_j^\top Z_j = \boldsymbol{b}^\top [X_1^\top \ \cdots \ X_{j-1}^\top]^\top \quad \text{almost surely.} \tag{D.22}$$

Substituting for Z_j using (D.10) we get that

$$(\boldsymbol{a}_j^\top \boldsymbol{N}_j) U_j = \boldsymbol{b}^\top [X_1^\top \ \cdots \ X_{j-1}^\top]^\top \quad \text{almost surely.} \tag{D.23}$$

We notice by definition of U_i, Y_i in (D.5) and (D.6) that $\mathbf{\Sigma}_{U_i|X_1,\cdots,X_{i-1}}$ and $\mathbf{\Sigma}_{Y_i|X_1,\cdots,X_{i-1}}$ are both full-rank. By Lemma 7, we also get that $\boldsymbol{N}_j$ is full-rank. Hence, the vector $\boldsymbol{a}_j^\top \boldsymbol{N}_j \neq \mathbf{0}$. By (D.23) and Lemma 6 the vector $\boldsymbol{a}_j^\top \boldsymbol{N}_j \neq \mathbf{0}$ is in the null space of $\mathbf{\Sigma}_{U_i|X_1,\cdots,X_{i-1}}$ which contradicts the fact that the matrix is full rank. We conclude that $\mathbf{\Sigma}_Z$ is full rank. $\square$

We notice that by definition of $\boldsymbol{N}_i$ in (D.9) and Lemma 6, we have that $\boldsymbol{N}_i U_i + \hat{\boldsymbol{N}}_i Y_i = \boldsymbol{A}[X_1^\top \ \cdots \ X_{i-1}^\top]^\top$ a.s. for some matrix $\boldsymbol{A}$, hence, $Z_i = \boldsymbol{N}_i U_i = \boldsymbol{A}' X_{-i}$ a.s. for some matrix $\boldsymbol{A}'$ (as both $Y_i, [X_1, \cdots, X_{i-1}]$ do not include X_i). By Lemma 10, as $Z_i = \boldsymbol{A}' X_{-i}$ a.s., then, if (V, W) breaks the dependency between $X_1, \cdots, X_n$, we must have that Z_i is a deterministic function of (V, W) for all i. And hence, similar to (5.25), we get that

$$Z = \boldsymbol{B} W \quad \text{almost surely,} \tag{D.24}$$

for some matrix $\boldsymbol{B}$. Next, by multiplying both sides by $Z^\top$ and taking the expectation we get that

$$\mathbf{\Sigma}_Z = \boldsymbol{B} \mathbb{E}[W Z^\top]. \tag{D.25}$$

Hence, we have that

$$\sum_{i=1}^{n} d_{Z_i} \overset{(i)}{=} \text{rank}(\mathbf{\Sigma}_Z) \leq \text{rank}(\boldsymbol{B}) \leq d_W, \tag{D.26}$$

where (i) follows from Lemma 21. As V, W are arbitrary in $\mathcal{W}$, we have that

$$d(X_1, \cdots, X_n) \geq \sum_{i=1}^{n} d_{Z_i}. \tag{D.27}$$

Combining (D.21) and (D.27), it follows that

$$d(X_1, \cdots, X_n) = \sum_{i=1}^{n} d_{Z_i}. \tag{D.28}$$

We notice that by definition of Z_i in (D.10) we have that

$$d_{Z_i} = \text{rank}(\mathbf{\Sigma}_{i|1:i-1}) + \text{rank}(\mathbf{\Sigma}_{i+1:n|1:i-1}) - \text{rank}(\mathbf{\Sigma}_{i:n|1:i-1}). \tag{D.29}$$

To further simplify this expression, we state and prove the following property about the rank of the conditional covariance matrix $\mathbf{\Sigma}_{X|Y}$ for two random vectors X, Y.

Proposition 4. *Let (X,Y) be a pair of jointly Gaussian random variables. We have that*

$$\mathit{rank}(\boldsymbol{\Sigma}_{X|Y}) = \mathit{rank}(\boldsymbol{\Sigma}) - \mathit{rank}(\boldsymbol{\Sigma}_Y), \tag{D.30}$$

where $\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_Y, \boldsymbol{\Sigma}_{X|Y}$ are the covariance matrices of $[X, Y], Y$ and X conditioned on Y respectively.

Proof. For a vector $\boldsymbol{a} \neq \mathbf{0}$ to be in the null space of $\boldsymbol{\Sigma}_{X|Y}$, we have by Lemma 6 that

$$\boldsymbol{a}^\top X = \boldsymbol{b}^\top Y \text{ almost surely} \tag{D.31}$$

for some vector $\boldsymbol{b}$. Without loss of generality we assume that $\boldsymbol{\Sigma}_Y$ is full rank, otherwise by Lemma 6 we can replace Y with $Y' = \boldsymbol{B}Y$, where $\boldsymbol{B}$ is a matrix containing basis for the row space of $\boldsymbol{\Sigma}_Y$. Hence, we may assume without loss of generality that

$$d_Y = \text{rank}(\boldsymbol{\Sigma}_Y) \tag{D.32}$$

Let $\boldsymbol{N}$ denote the basis of the covariance matrix $\boldsymbol{\Sigma}$,

$$\boldsymbol{N} = [\boldsymbol{N}_X^{r(\boldsymbol{N})\times d_X} \; \boldsymbol{N}_Y^{r(\boldsymbol{N})\times d_X}]. \tag{D.33}$$

Similarly to Lemma 7, we get that $\boldsymbol{\Sigma}_Y$ full rank implies that

$$\boldsymbol{N}_X \text{ is full row rank} \tag{D.34}$$

We also notice by (D.31) that the rank of $\boldsymbol{N}_X$ is the dimension of the null space of $\boldsymbol{\Sigma}_{X|Y}$. This is because from (D.31), $[\boldsymbol{a}\ \boldsymbol{b}]$ is in the null space of $\boldsymbol{\Sigma}$ (by Lemma 6), hence, $\boldsymbol{a}$ is in the row space of $\boldsymbol{N}_X$ by definition of $\boldsymbol{N}_X$ in (D.33); conversely, every vector in the row space of $\boldsymbol{N}_X$ satisfies (D.31) for some $\boldsymbol{b}$ (also by definition of $\boldsymbol{N}_X$). Hence, by noticing that the dimension of the null space of $\boldsymbol{\Sigma}_{X|Y}$ is $d - \text{rank}(\boldsymbol{\Sigma}_{X|Y})$ we get that

$$d_X - \text{rank}(\boldsymbol{\Sigma}_{X|Y}) = \text{rank}(\boldsymbol{N}_X) \overset{\text{(D.34)}}{=} \text{rank}(\boldsymbol{N}) = d_X + d_Y - \text{rank}(\boldsymbol{\Sigma}).$$

It follows that

$$\text{rank}(\boldsymbol{\Sigma}_{X|Y}) = \text{rank}(\boldsymbol{\Sigma}) - d_Y \overset{\text{(D.32)}}{=} \text{rank}(\boldsymbol{\Sigma}) - \text{rank}(\boldsymbol{\Sigma}_Y).$$

□

By substituting (D.29) in (D.28) we get that

$$\begin{aligned}
d(X_1, \cdots, X_n) = & \sum_{i=1}^{n} d_{Z_i} = \sum_{i=1}^{n} \left(\text{rank}(\mathbf{\Sigma}_{i|1:i-1}) + \text{rank}(\mathbf{\Sigma}_{i+1:n|1:i-1}) - \text{rank}(\mathbf{\Sigma}_{i:n|1:i-1})\right) \\
\overset{(i)}{=} & \sum_{i=1}^{n} (\text{rank}(\mathbf{\Sigma}_{1:i}) - \text{rank}(\mathbf{\Sigma}_{1:i-1}) + \text{rank}(\mathbf{\Sigma}_{-i}) - \text{rank}(\mathbf{\Sigma}_{1:i-1}) - \text{rank}(\mathbf{\Sigma}) + \text{rank}(\mathbf{\Sigma}_{1:i-1})) \\
\overset{(ii)}{=} & \text{rank}(\mathbf{\Sigma}) + \left(\sum_{i=1}^{n} \text{rank}(\mathbf{\Sigma}_{-i})\right) - (n)\,\text{rank}(\mathbf{\Sigma}) \\
= & \left(\sum_{i=1}^{n} \text{rank}(\mathbf{\Sigma}_{-i})\right) - (n-1)\text{rank}(\mathbf{\Sigma}),
\end{aligned} \tag{D.35}$$

where (i) follows by Proposition 4, and (ii) follows from the fact that the first two terms inside the summation form a telescopic summation. □

Complexity of Computing W. It is easy to see that every step (finding row space, inverse, matrix multiplication and summations) in the for loop in Algorithm 9 can be performed in $O((\sum_{i=1}^{n} d_{X_i})^3)$. It follows that the overall time complexity of computing W is $O((\sum_{i=1}^{n} d_{X_i})^4)$.

Corollary 8. *The time complexity of computing W in Algorithm 9 is $O((\sum_{i=1}^{n} d_{X_i})^4)$.*

Lemma 5. Let $[X_1, \cdots, X_n]$ be a jointly Gaussian random vector. Then, the Rényi common information dimension between $X_1, \cdots, X_n$ with respect to the class of $\mathcal{F}$ of linear functions is given by

$$d^R(X_1, \cdots, X_n) = d(X_1, \cdots, X_n).$$

Proof. Let W be such that $W \in \mathcal{W}$ and $d_W = d(X_1, \cdots, X_n)$. As $W \in \mathcal{W}$, we have that W is a linear function of $[X_1, \cdots, X_n]$, hence, it is jointly Gaussian. Then, we have that $d^R(W) \leq d_W$ [Ren59]. This shows that

$$d^R(X_1, \cdots, X_n) \leq d(X_1, \cdots, X_n). \tag{D.36}$$

To show that $d^R(X_1, \cdots, X_n) \geq d(X_1, \cdots, X_n)$, we first observe that for every $W \in \mathcal{W}$, we have that $d^R(W) \in \mathbb{N}$, hence, the infimum in the definition of $d^R(X_1, \cdots, X_n)$ can be replaced by a minimum. Then there exists W such that $W \in \mathcal{W}$ and $d^R(W) = d^R(X_1, \cdots, X_n)$. As $W \in \mathcal{W}$, we have that W is jointly Gaussian. By Lemma 6, we have that there is $I \subseteq \{1, ..., d_W\}$ such that W_I has a non-singular covariance matrix and $W = \boldsymbol{B} W_I$ for some matrix $\boldsymbol{B}$. By construction of W_I we have that $d^R(W_I) \leq d^R(W)$. Hence, without loss of generality, we assume that $\boldsymbol{\Sigma}_W$ is non-singular, otherwise, we can replace W by W_I. We also note that since $\boldsymbol{\Sigma}_W^{1/2} W$ has independent entries, we have that $d^R(W) = d^R(\boldsymbol{\Sigma}_W^{1/2} W) = d_W$ [Ren59]. This shows that

$$d(X_1, \cdots, X_n) \leq d^R(X_1, \cdots, X_n). \tag{D.37}$$

Combining (D.36) and (D.37) concludes the proof. □

D.3 Proof of Theorem 16

Theorem 16. *Let $X = [X_1, \cdots, X_n]$ be a jointly Gaussian random vector. The GKCID between $X_1, \cdots, X_n$ with respect to the class of linear functions is given by*

$$d^{GK}(X_1, \cdots, X_n) = r(\tilde{\boldsymbol{\Sigma}}) - \mathit{rank}(\tilde{\boldsymbol{\Sigma}}), \tag{5.11}$$

where $r(\tilde{\boldsymbol{\Sigma}})$ is the number of rows of $\tilde{\boldsymbol{\Sigma}}$, with

$$\tilde{\boldsymbol{\Sigma}} = \begin{bmatrix} \boldsymbol{\Sigma}_{X_1'X_2'} & 0 & \cdots & 0 \\ 0 & \boldsymbol{\Sigma}_{X_2'X_3'} & \cdots & 0 \\ & \cdots & & \\ 0 & 0 & \cdots & \boldsymbol{\Sigma}_{X_{n-1}'X_n'} \\ \mathbf{0}_1\,\mathbf{1}_2 & \mathbf{1}_2\,\mathbf{0}_3 & \cdots & \mathbf{0}_{n-1}\,\mathbf{0}_n \\ \mathbf{0}_1\,\mathbf{0}_2 & \mathbf{0}_2\,\mathbf{1}_3 & \cdots & \mathbf{0}_{n-1}\,\mathbf{0}_n \\ \mathbf{0}_1\,\mathbf{0}_2 & \mathbf{0}_2\,\mathbf{0}_3 & \cdots & \mathbf{1}_{n-1}\,\mathbf{0}_n \end{bmatrix}, \tag{5.12}$$

$X_i' = \boldsymbol{F}_i X_i, \forall i \in [n]$, $\boldsymbol{F}_i$ is a basis of the row space of $\boldsymbol{\Sigma}_{X_i}$, $\mathbf{0}_i \in \mathbb{R}^{1 \times d_{X_i'}}$ and $\mathbf{1}_i \in \mathbb{R}^{1 \times d_{X_i'}}$ are all zeros (and ones respectively) row vectors with the same dimension as X_i'.

Proof. Let $\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}$ be the right null space of the matrix $\tilde{\boldsymbol{\Sigma}}$. We will show that $\text{rank}(\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}) = d^{GK}(X_1, \cdots, X_n)$. As in the proof of Lemma 5, we observe that for every W in the optimization set, we have that $d^R(W) \in \mathbb{N}$. We also note that $d^{GK}(X_1, \cdots, X_n) \leq d_{X_1}$. Hence, the supremum in the definition of d^{GK} is attainable; namely, there exists $W, f_1, \cdots, f_n$ with $W = f_i(X_i), \forall i \in [n]$ and $d^R(W) = d^{GK}(X_1, \cdots, X_n)$. As f_i is linear, we have that W is jointly Gaussian. By Lemma 6, we have that there is $I \subseteq \{1, ..., d_W\}$ such that W_I has a non-singular covariance matrix and $W = \boldsymbol{G}W_I$ for some matrix $\boldsymbol{G}$. By construction of W_I we have that $d^R(W_I) \leq d^R(W)$. Hence, without loss of generality, we assume that $\boldsymbol{\Sigma}_W$ is non-singular, otherwise, we can replace W by W_I. We also note that since $\boldsymbol{\Sigma}_W^{1/2} W$ has independent entries, we have that $d^R(W) = d^R(\boldsymbol{\Sigma}_W^{1/2} W) = d_W$ [Ren59]. This shows that there is W with $d_W = d^{GK}(X_1, \cdots, X_n)$ and matrices $\boldsymbol{K}_1, \cdots, \boldsymbol{K}_n$ such that $W = \boldsymbol{K}_1 X_1 = \cdots = \boldsymbol{K}_n X_n$ almost surely and W is non-singular, hence, $\boldsymbol{K}_i$ is full row rank for each i. Hence, the matrix

$$\boldsymbol{N}_W = \begin{bmatrix} \boldsymbol{K}_1 & -\boldsymbol{K}_2 & \boldsymbol{K}_2 & -\boldsymbol{K}_3 & \boldsymbol{K}_3 & \cdots & \boldsymbol{K}_{n-1} & -\boldsymbol{K}_n \end{bmatrix} \tag{D.38}$$

is in the null space of $\tilde{\boldsymbol{\Sigma}}$ by Lemma 6. As $\boldsymbol{K}_1$ is full row rank, we get that $\boldsymbol{N}_W$ is full row rank, hence,

$$\text{rank}(\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}) \geq \text{rank}(\boldsymbol{N}_W) = r(\boldsymbol{N}_W) = d^{GK}(X_1, \cdots, X_n). \tag{D.39}$$

It remains to show that $\text{rank}(\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}) \leq d^{GK}(X_1, \cdots, X_n)$. Let us partition $\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}$ (recall that N is the basis for the left null space of $\tilde{\boldsymbol{\Sigma}}$) as

$$\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}} = \begin{bmatrix} \boldsymbol{N}_1^{r(\boldsymbol{N}) \times d_{X_1'}} & \boldsymbol{N}_2^{r(\boldsymbol{N}) \times d_{X_2'}} & \cdots & \boldsymbol{N}_n^{r(\boldsymbol{N}) \times d_{X_n'}} \end{bmatrix}. \tag{D.40}$$

By Lemma 6 we get that the following is a common function

$$W = \boldsymbol{N}_1 \boldsymbol{F}_1 X_1 = -\boldsymbol{N}_2 \boldsymbol{F}_2 X_2 = \cdots = (-1)^{n+1} \boldsymbol{N}_n \boldsymbol{F}_n X_n. \tag{D.41}$$

Hence, by definition of d^{GK} we get that

$$d^{GK}(X_1, \cdots, X_n) \geq d^R(W) \tag{D.42}$$

where the common function of variable X_i achieving this is

$$W = (-1)^{i+1}\boldsymbol{N}_i\boldsymbol{F}_iX_i. \tag{D.43}$$

We observe that if $\boldsymbol{a}^\top\boldsymbol{N}_1 = \boldsymbol{0}$ for some vector $\boldsymbol{a}$, then by (D.41) we have that $\boldsymbol{a}^\top\boldsymbol{N}_iX_i' = \boldsymbol{0}, \forall i \in [n]$. As X_i' has non-singular covariance matrix by construction, we get that $\boldsymbol{a}^\top\boldsymbol{N}_i = \boldsymbol{0}, \forall i \in [n]$, hence, $\boldsymbol{a}^\top\boldsymbol{N} = \boldsymbol{0}$. As $\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}$ is full row rank, we conclude that $\boldsymbol{a} = \boldsymbol{0}$. This shows that $\boldsymbol{a}^T\boldsymbol{N}_1 = \boldsymbol{0}$ if and only if $\boldsymbol{a} = \boldsymbol{0}$, hence, $\boldsymbol{N}_1$ is full row rank.

We notice that since $X_1' = \boldsymbol{F}_1X_1$ has non-singular covariance matrix and $\boldsymbol{N}_1$ is full row rank, then W has a non-singular covariance matrix. Hence, we have that $d_W = d^R(W)$, hence from (D.42) we get that

$$d^{GK}(X_1, \cdots, X_n) \geq d_W = r(\boldsymbol{N}_1) = \text{rank}(\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}) \tag{D.44}$$

From (D.39) and (D.44) we get that

$$d^{GK}(X_1, \cdots, X_n) = \text{rank}(\boldsymbol{N}_{\tilde{\boldsymbol{\Sigma}}}) = r(\tilde{\boldsymbol{\Sigma}}) - \text{rank}(\tilde{\boldsymbol{\Sigma}}). \tag{D.45}$$

□

D.4 Proofs of Lemma 11 and Lemma 12

In this section, we give the proofs of the following two properties of covariance matrices and the common information, which are used in both Theorems 17 and 18.

Lemma 11. Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be jointly Gaussian variables with covariance matrix $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_X & \boldsymbol{\Sigma}_{XY}^\top \\ \boldsymbol{\Sigma}_{XY} & \boldsymbol{\Sigma}_Y \end{bmatrix}$, and $d = \min\{d_X, d_Y\}$. Then the singular values of $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$, denoted as $\{\sigma_i\}_{i=1}^d$, satisfy

$$0 \leq \sigma_i \leq 1, \forall i \in [d]. \tag{5.29}$$

Proof. Note that singular values are non-negative. In the following, we show $\sigma_i, \rho_i \leq 1$. Let $\boldsymbol{U\Lambda V}^\top$ be the singular value decomposition of $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$, where $\boldsymbol{\Lambda} = \text{diag}(\rho_1, \cdots, \rho_d)$.

We denote $X' = \boldsymbol{U}^\top \boldsymbol{\Sigma}_X^{-1/2} X$, and $Y' = \boldsymbol{V}^\top \boldsymbol{\Sigma}_Y^{-1/2} Y$, $r_X = \text{rank}(\boldsymbol{\Sigma}_X)$ and $r_Y = \text{rank}(\boldsymbol{\Sigma}_Y)$. Then the covariance matrix $\boldsymbol{\Sigma}'$ of X', Y' are given by

$$\boldsymbol{\Sigma}_{X'} = \begin{bmatrix} \boldsymbol{I}_{r_X} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \boldsymbol{\Sigma}_{Y'} = \begin{bmatrix} \boldsymbol{I}_{r_Y} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \boldsymbol{\Sigma}_{X'Y'} = \boldsymbol{\Lambda}.$$

For $i \le \min\{r_X, r_Y\}$, we have that $\sigma_i = 0$. By Cauchy-Schwarz inequality, if $i \le \min\{r_X, r_Y\}$, we have that $\sigma_i^2 = \text{cov}(X_i', Y_i')^2 \le \text{var}(X_i')\text{var}(Y_i') = 1$. □

Lemma 12. Assume $X \in \mathbb{R}^{d_X}$, $Y \in \mathbb{R}^{d_Y}$ are jointly Gaussian variables with covariance matrix $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_X & \boldsymbol{\Sigma}_{XY}^\top \\ \boldsymbol{\Sigma}_{XY} & \boldsymbol{\Sigma}_Y \end{bmatrix}$, and $\{\sigma_i\}$ are the singular values of $\boldsymbol{\Sigma}_X^{-1/2} \boldsymbol{\Sigma}_{XY} \boldsymbol{\Sigma}_Y^{-1/2}$. Then the common information dimension between X and Y, with respect to linear functions, satisfies

$$d(X, Y) = \sum_{i=1}^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}. \tag{5.30}$$

Proof. Similar to the proof of Lemma 11, let $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top$ be the singular value decomposition of $\boldsymbol{\Sigma}_X^{-1/2} \boldsymbol{\Sigma}_{XY} \boldsymbol{\Sigma}_Y^{-1/2}$, where $\boldsymbol{\Lambda} = \text{diag}(\rho_1, \cdots, \rho_d)$. We let $X' = \boldsymbol{U}^\top \boldsymbol{\Sigma}_X^{-1/2} X$, and $Y' = \boldsymbol{V}^\top \boldsymbol{\Sigma}_Y^{-1/2} Y$, $r_X = \text{rank}(\boldsymbol{\Sigma}_X)$ and $r_Y = \text{rank}(\boldsymbol{\Sigma}_Y)$. Then the covariance matrix $\boldsymbol{\Sigma}'$ of X', Y' are given by

$$\boldsymbol{\Sigma}_{X'} = \begin{bmatrix} \boldsymbol{I}_{r_X} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \boldsymbol{\Sigma}_{Y'} = \begin{bmatrix} \boldsymbol{I}_{r_Y} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}, \boldsymbol{\Sigma}_{X'Y'} = \boldsymbol{\Lambda}.$$

We notice that the rank of the covariance matrix $\text{rank}(\boldsymbol{\Sigma}') = r_X + r_Y - \sum_i^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}$. By Theorem 14, we have that $d(X', Y') = \text{rank}(\boldsymbol{\Sigma}_{X'}) + \text{rank}(\boldsymbol{\Sigma}_{Y'}) - \text{rank}(\boldsymbol{\Sigma}') = \sum_i^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}$. Since one-to-one transformations preserve the ranks of covariance matrices, $d(X, Y) = d(X', Y') = \sum_i^{\min\{d_X, d_Y\}} \mathbb{1}\{\sigma_i = 1\}$. □

D.5 Proof of Theorem 17

Theorem 17. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly singular Gaussian variables, and $\{(X_\epsilon, Y_\epsilon)\}_{\epsilon > 0}$ be a sequence as defined in* (5.27). *Then the common information $C(X_\epsilon, Y_\epsilon)$ satisfies*

$$\lim_{\epsilon \downarrow 0} \frac{C(X_\epsilon, Y_\epsilon)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = d(X, Y). \tag{5.31}$$

Proof. Let $d = \min\{d_X, d_Y\}$. We use $\{\sigma_i\}_{i=1}^d$ and $\{\rho_i(\epsilon)\}_{i=1}^d$ to denote the singular values of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$ and $\mathbf{\Sigma}_{X_\epsilon}^{-1/2}\mathbf{\Sigma}_{X_\epsilon Y_\epsilon}\mathbf{\Sigma}_{Y_\epsilon}^{-1/2}$, respectively, in a decreasing order. As definition (5.27) requires that $|\rho_i(\epsilon) - \sigma_i| = \epsilon$, and from Lemma 11 we have that $0 \leq \rho_i(\epsilon) \leq 1$, then $\rho_i(\epsilon) = 1 - \epsilon$ whenever $\sigma_i = 1$. Let $k = \sum_i^d \mathbb{1}\{\sigma_i = 1\}$ and $l = \arg\max\{\sigma_i | \forall i : \sigma_i \neq 1\}$. Since we assume $\{\sigma_i\}$ are in decreasing order, $l = k + 1$. Recall that

$$C(X_\epsilon, Y_\epsilon) = \frac{1}{2}\sum_{i=1}^{d} \log \frac{1 + \rho_i(\epsilon)}{1 - \rho_i(\epsilon)} \tag{D.46}$$

Since $\frac{1+\rho_i(\epsilon)}{1-\rho_i(\epsilon)} \geq 0$ and is an increasing function of $\rho_i(\epsilon)$, then for $\epsilon \leq (1 - \sigma_l)/2$ we have that

$$\frac{k}{2}\log\frac{2-\epsilon}{\epsilon} \leq C(X_\epsilon, Y_\epsilon) \leq \frac{k}{2}\log\frac{2-\epsilon}{\epsilon} + \frac{d-k}{2}\log\frac{1+\sigma_l+\epsilon}{1-\sigma_l-\epsilon} \tag{D.47}$$

It follows that

$$\lim_{\epsilon \downarrow 0} \frac{C(X_\epsilon, Y_\epsilon)}{\frac{1}{2}\log(1/\epsilon)} = \lim_{\epsilon \downarrow 0} \frac{\frac{k}{2}\log\frac{2-\epsilon}{\epsilon}}{\frac{1}{2}\log(1/\epsilon)} = \lim_{\epsilon \downarrow 0} \frac{k \log\frac{1}{\epsilon}}{\log(1/\epsilon)} = k \tag{D.48}$$

As Lemma 12 proves that $k = d(X, Y)$, this concludes the proof of Theorem 17. □

D.6 Proof of Theorem 18

Theorem 18. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly Gaussian random variables, then*

$$\lim_{\epsilon \downarrow 0} \frac{C_\epsilon(X, Y)}{\frac{1}{2}\log(\frac{1}{\epsilon})} = d(X, Y). \tag{5.32}$$

Proof. We recall that $C_\epsilon(X, Y)$ is defined as the optimal value of the optimization problem

$$\begin{aligned} \min_{\hat{\mathbf{\Sigma}}} \quad & C(\hat{X}, \hat{Y}) \\ \text{s.t.} \quad & \|\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}\|_F \leq \epsilon. \end{aligned} \tag{D.49}$$

Due to the difficulty in finding a closed-form solution of the optimization problem, we instead derive an upper and a lower bound on $C_\epsilon(X, Y)$ that have the same asymptotic behaviors.

Upper bound. We derive the upper bound by constructing a feasible solution. We first choose the marginal covariance matrices to be $\mathbf{\Sigma}_{\hat{X}} = \mathbf{\Sigma}_X, \mathbf{\Sigma}_{\hat{Y}} = \mathbf{\Sigma}_Y$. In the following, we provide our choice for the cross-covariance matrix $\mathbf{\Sigma}_{\hat{X},\hat{Y}}$.

Let $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top$, $\hat{\boldsymbol{U}}\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{V}}^\top$ denote the singular value decompositions of $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$ and $\boldsymbol{\Sigma}_X^{-1/2}\boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\boldsymbol{\Sigma}_Y^{-1/2}$ respectively, where $\boldsymbol{\Lambda}$, $\hat{\boldsymbol{\Lambda}}$ are diagonal matrices containing the singular values, and $\boldsymbol{U}, \hat{\boldsymbol{U}}, \boldsymbol{V}, \hat{\boldsymbol{V}}$ are orthonormal matrices. We choose $\hat{\boldsymbol{U}} = \boldsymbol{U}$ and $\hat{\boldsymbol{V}} = \boldsymbol{V}$. Then $\|\boldsymbol{U}(\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}})\boldsymbol{V}^\top\|_F = \|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F$, because the Frobenius norm is unitary invariant. If we let $s = \|\boldsymbol{\Sigma}_X^{1/2}\|_F\|\boldsymbol{\Sigma}_Y^{1/2}\|_F$, by Cauchy–Schwarz inequality of the Frobenius norm, we have that

$$\begin{aligned} \|\boldsymbol{\Sigma}_{\hat{X}\hat{Y}} - \boldsymbol{\Sigma}_{XY}\|_F &= \|\boldsymbol{\Sigma}_X^{1/2}(\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}})\boldsymbol{\Sigma}_Y^{1/2}\|_F \\ &\leq s\|(\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}})\|_F. \end{aligned} \tag{D.50}$$

Hence, choosing $\hat{\boldsymbol{\Lambda}}$ satisfying $\|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F \leq \frac{1}{s}\epsilon$ provides a feasible solution. Among all such values of $\hat{\boldsymbol{\Lambda}}$, we choose the one minimizing the common information $C(\hat{X}, \hat{Y})$. In particular, we choose $\hat{\boldsymbol{\Lambda}}$ to be the solution of

$$S_{\frac{\epsilon}{s}}(X, Y) := \min_{\|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F \leq \frac{1}{s}\epsilon} C(\hat{X}, \hat{Y}) \tag{D.51}$$

As a result, we get that

$$C_\epsilon(X, Y) \leq S_{\frac{\epsilon}{s}}(X, Y). \tag{D.52}$$

In the following, we provide a lower bound on $C_\epsilon(X, Y)$ using $S_\epsilon(X, Y)$. We analyze the asymptotic behavior of $S_\epsilon(X, Y)$ at the end of the proof.

Lower bound. Next, we derive a lower bound on the optimal objective. Note that the marginal distributions of X and Y might also be singular (i.e., rank($\boldsymbol{\Sigma}_X$) $< d_X$ and rank($\boldsymbol{\Sigma}_Y$) $< d_Y$). In this case, there exist one-to-one transformations $X' = \boldsymbol{P}_X X$ and $Y' = \boldsymbol{P}_Y Y$ (with rows from the Identity Matrix) that select a subset of elements in X and Y such that $\boldsymbol{\Sigma}_{X'}$ and $\boldsymbol{\Sigma}_{Y'}$ are full-rank. If we also apply the same transformation $\hat{X}' = \boldsymbol{P}_X \hat{X}$ and $\hat{Y}' = \boldsymbol{P}_Y \hat{Y}$, then $C(\hat{X}', \hat{Y}') = C(\hat{X}, \hat{Y})$ because common information is preserved under one-to-one linear transformations [SC15]. Then the objective of (D.49) remains the same and the constraints are relaxed, therefore solving the problem (D.49) on X', Y' would give a lower bound on $C_\epsilon(X, Y)$. In the remaining part of the proof, we assume without loss of generality that $\boldsymbol{\Sigma}_X$ and $\boldsymbol{\Sigma}_Y$ are full-rank.

The proof relies on relaxing the constraints in (D.49) to result in a smaller optimal value. The first relaxation we make is the following

$$\begin{aligned} \min_{\hat{\boldsymbol{\Sigma}}} \quad & C(\hat{X}, \hat{Y}) && \text{(D.53)} \\ \text{s.t.} \quad & \|\boldsymbol{\Sigma}_X - \boldsymbol{\Sigma}_{\hat{X}}\|_F \leq \epsilon \\ & \|\boldsymbol{\Sigma}_Y - \boldsymbol{\Sigma}_{\hat{Y}}\|_F \leq \epsilon \\ & \|\boldsymbol{\Sigma}_{XY} - \boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \epsilon. && \text{(D.54)} \end{aligned}$$

In the following, we further relax the last constraint (D.54) to a bound on the difference between the singular values using the following lemma, which will be proved in the end of this section.

Lemma 22. If covariance matrices satisfy $\|\boldsymbol{\Sigma}_X - \boldsymbol{\Sigma}_{\hat{X}}\|_F \leq \epsilon$,$\|\boldsymbol{\Sigma}_Y - \boldsymbol{\Sigma}_{\hat{Y}}\|_F \leq \epsilon$, and $\|\boldsymbol{\Sigma}_{XY} - \boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \epsilon$, then

$$\|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F \leq c\epsilon, \text{for some constant } c.$$

As a result, the following optimization problem has a larger feasible set, hence, lower optimal value

$$S_{c\epsilon}(X, Y) := \min_{\|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F \leq c\epsilon} C(\hat{X}, \hat{Y}) \tag{D.55}$$

Combining with (D.52), we have shown that

$$S_{c\epsilon}(X, Y) \leq C_\epsilon(X, Y) \leq S_{\frac{\epsilon}{s}}(X, Y). \tag{D.56}$$

Asymptotic behavior of S_ϵ. In the following, we analyze the asymptotic behavior of S_ϵ. Let $d = \min\{d_X, d_Y\}$, and $\boldsymbol{\Lambda} = \operatorname{diag}\{\sigma_1, \sigma_2, \cdots, \sigma_d\}$, $\hat{\boldsymbol{\Lambda}} = \operatorname{diag}\{\rho_1, \rho_2, \cdots, \rho_d\}$ with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_d$ and $\rho_1 \geq \rho_2 \geq \cdots \geq \rho_d$. By Lemma 11, we have that $0 \leq \sigma_i, \rho_i \leq 1$. Then $S_\epsilon(X, Y)$ can be rewritten as

$$\begin{aligned} \min_{\rho} \quad & \frac{1}{2} \sum_{i=1}^{d} \log \frac{1 + \rho_i}{1 - \rho_i} \\ \text{s.t.} \quad & \sum_{i=1}^{d} (\sigma_i - \rho_i)^2 \leq \epsilon^2;\ 0 \leq \rho_i \leq 1, \forall i \in [d]. \end{aligned} \tag{D.57}$$

Next, let $k = \sum_i^d \mathbb{1}\{\sigma_i = 1\}$, we will show that $\lim_{\epsilon\downarrow 0} \frac{S_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)} = k$ using upper and lower bounds. Define $\tilde{\rho} \in \mathbb{R}^d$ as

$$\tilde{\rho}_i = \begin{cases} \sigma_i, & \text{if } \sigma_i \neq 1 \\ 1 - \frac{\epsilon}{\sqrt{k}}, & \text{if } \sigma_i = 1. \end{cases} \tag{D.58}$$

Since $\tilde{\rho}$ is a feasible solution of problem (D.57), we have that $S_\epsilon(X,Y) \leq \frac{1}{2}\sum_{i=1}^d \log\frac{1+\tilde{\rho}_i}{1-\tilde{\rho}_i}$. Then we have

$$\begin{aligned} \lim_{\epsilon\downarrow 0} S_\epsilon(X,Y) \leq & \lim_{\epsilon\downarrow 0} \frac{\sum_{i=1}^d \log\frac{1}{1-\tilde{\rho}_i}}{\log(1/\epsilon)} + \frac{\sum_{i=1}^d \log(1+\tilde{\rho}_i)}{\log(1/\epsilon)} \\ = & \lim_{\epsilon\downarrow 0} \frac{\sum_{i=1}^d \log\frac{1}{1-\tilde{\rho}_i}}{\log(1/\epsilon)} \\ = & \lim_{\epsilon\downarrow 0} \frac{\sum_{i:\sigma_i=1} \log\frac{\sqrt{k}}{\epsilon} + \sum_{i:\sigma_i\neq 1} \log\frac{1}{1-\sigma_i}}{\log(1/\epsilon)} \\ = & \lim_{\epsilon\downarrow 0} \frac{k\log\frac{\sqrt{k}}{\epsilon}}{\log(1/\epsilon)} = k \end{aligned} \tag{D.59}$$

To derive the lower bound, we consider the following variant optimization problem

$$\begin{aligned} \min_\rho \quad & \frac{1}{2}\sum_{i:\sigma_i=1} \log\frac{1}{1-\rho_i} \\ \text{s.t.} \quad & \sum_{i:\sigma_i=1} (\sigma_i - \rho_i)^2 \leq \epsilon^2;\ 0 \leq \rho_i \leq 1, \forall i \in [d] \end{aligned} \tag{D.60}$$

where the feasible set is relaxed, hence, gives a smaller optimal value, and the objective function is the sum of a subset of the terms in (D.57). Since $\log\frac{1+\rho_i}{1-\rho_i} \geq 0$ and $\log(1+\rho_i) \geq 0, \forall\rho_i$, the optimal value of problem (D.60) is a lower bound on $S_\epsilon(X,Y)$.

In the following, we solve problem (D.60) and show that the optimal solution $\rho_i^* = \tilde{\rho}_i$. First, we observe that by Jensen's inequality, for all ρ_i in the feasible set of problem (D.60),

$$-\sum_{i:\sigma_i=1} \log(1-\rho_i)^2 \geq -k\log\left(\frac{\sum_{i:\sigma_i=1}(1-\rho_i)^2}{k}\right) \overset{(a)}{\geq} -k\log(1-\tilde{\rho}_i)^2, \tag{D.61}$$

where the inequality (a) follows from the fact that ρ_i is in the feasible set of Problem (D.60). This shows that $\rho_i^* = \tilde{\rho}_i$ is an optimal solution for Problem (D.60). This shows that $\frac{1}{2}\sum_{i:\sigma_i=1}\log\frac{1}{1-\tilde{\rho}_i} \leq$

$S_\epsilon(X,Y)$. Using the similar calculations as in (D.59) , we can also obtain that

$$\lim_{\epsilon\downarrow 0}\frac{S_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)} \geq k \tag{D.62}$$

Combining (D.59), (D.62) and (D.56), we have shown that

$$\lim_{\epsilon\downarrow 0}\frac{C_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)} = k. \tag{D.63}$$

Applying Lemma 12, we can now conclude that

$$\lim_{\epsilon\downarrow 0}\frac{C_\epsilon(X,Y)}{\frac{1}{2}\log(1/\epsilon)} = d(X,Y) \tag{D.64}$$

□

Lemma 22. If covariance matrices satisfy $\|\mathbf{\Sigma}_X - \mathbf{\Sigma}_{\hat{X}}\|_F \leq \epsilon$,$\|\mathbf{\Sigma}_Y - \mathbf{\Sigma}_{\hat{Y}}\|_F \leq \epsilon$, and $\|\mathbf{\Sigma}_{XY} - \mathbf{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \epsilon$, then

$$\|\mathbf{\Lambda} - \hat{\mathbf{\Lambda}}\|_F \leq c\epsilon, \text{for some constant } c.$$

Proof. In this proof, we use c to denote a general-purpose constant that depends on $\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y$. With abuse of notation, c may refer to different values in different equations. Recall that $\mathbf{\Lambda}$ and $\hat{\mathbf{\Lambda}}$ are the diagonal matrices containing the singular values of $\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}$ and $\mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}$ respectively. We notice that

$$\begin{aligned}
\|\mathbf{\Lambda} - \hat{\mathbf{\Lambda}}\|_F &\leq \|\mathbf{\Sigma}_X^{-1/2}\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2}\mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}\|_F && \text{(D.65)}\\
&\leq \|(\mathbf{\Sigma}_X^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2})\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2}\|_F + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}(\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2})\|_F && \text{(D.66)}\\
&\leq \|\mathbf{\Sigma}_X^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{XY}\|_F\|\mathbf{\Sigma}_Y^{-1/2}\|_F + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{XY}\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{X}\hat{Y}}\mathbf{\Sigma}_{\hat{Y}}^{-1/2}\|_F && \text{(D.67)}\\
&\leq \|\mathbf{\Sigma}_X^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{XY}\|_F\|\mathbf{\Sigma}_Y^{-1/2}\|_F + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|(\mathbf{\Sigma}_{XY} - \mathbf{\Sigma}_{\hat{X}\hat{Y}})\mathbf{\Sigma}_Y^{-1/2}\|_F \\
&\quad + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{\hat{X}\hat{Y}}(\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{Y}}^{-1/2})\|_F && \text{(D.68)}\\
&\leq \|\mathbf{\Sigma}_X^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{XY}\|_F\|\mathbf{\Sigma}_Y^{-1/2}\|_F + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{XY} - \mathbf{\Sigma}_{\hat{X}\hat{Y}}\|_F\|\mathbf{\Sigma}_Y^{-1/2}\|_F \\
&\quad + \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\mathbf{\Sigma}_{\hat{X}\hat{Y}}\|_F\|\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{Y}}^{-1/2}\|_F && \text{(D.69)}
\end{aligned}$$

$$\leq c\|\boldsymbol{\Sigma}_X^{-1/2} - \boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F + c\epsilon\|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F + \|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F\|\boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F\|\boldsymbol{\Sigma}_Y^{-1/2} - \boldsymbol{\Sigma}_{\hat{Y}}^{-1/2}\|_F, \tag{D.70}$$

where c is a constant that may depend on $\boldsymbol{\Sigma}_X, \boldsymbol{\Sigma}_Y, \boldsymbol{\Sigma}_{XY}$, (D.65) uses Von Neumann's trace inequality [Von37], (D.66) is obtained by subtracting and adding $\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\boldsymbol{\Sigma}_{XY}\boldsymbol{\Sigma}_Y^{-1/2}$ then applying the triangle inequality, similarly, (D.68) is obtained by subtracting and adding $\boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\boldsymbol{\Sigma}_Y^{-1/2}$, (D.67) and (D.69) use the Cauchy–Schwarz inequality, and (D.70) uses the bound $\|\boldsymbol{\Sigma}_{XY} - \boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \epsilon$.

In the following, we will bound each term in (D.70). We first show that $\|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F$ is bounded. We have that

$$\|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F \leq \sqrt{d_X}\sigma_{d_X}(\boldsymbol{\Sigma}_{\hat{X}})^{-1/2}, \tag{D.71}$$

where $\sigma_{d_X}(\boldsymbol{\Sigma}_{\hat{X}})$ denote the smallest singular value of the matrix $\boldsymbol{\Sigma}_{\hat{X}}$. From the bound $\|\boldsymbol{\Sigma}_{XY} - \boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \epsilon$, we have that

$$\sigma_{d_X}(\boldsymbol{\Sigma}_{\hat{X}}) \geq \sigma_{d_X}(\boldsymbol{\Sigma}_X) - \epsilon \geq \sigma_{d_X}(\boldsymbol{\Sigma}_X)/2 \tag{D.72}$$

for $\epsilon \leq \sigma_{d_X}(\boldsymbol{\Sigma}_X)/2$. Hence, we have that

$$\|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F \leq c, \tag{D.73}$$

where c is a constant that may depend on $\boldsymbol{\Sigma}_X, \boldsymbol{\Sigma}_Y, \boldsymbol{\Sigma}_{XY}$. Similarly,

$$\|\boldsymbol{\Sigma}_{\hat{X}}^{-1}\|_F \leq c. \tag{D.74}$$

By the triangle inequality, we also have that

$$\|\boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq \|\boldsymbol{\Sigma}_{XY}\|_F + \|\boldsymbol{\Sigma}_{XY} - \boldsymbol{\Sigma}_{\hat{X}\hat{Y}}\|_F \leq c. \tag{D.75}$$

Substituting in (D.70) we get that

$$\|\boldsymbol{\Lambda} - \hat{\boldsymbol{\Lambda}}\|_F \leq c(\epsilon + \|\boldsymbol{\Sigma}_X^{-1/2} - \boldsymbol{\Sigma}_{\hat{X}}^{-1/2}\|_F + \|\boldsymbol{\Sigma}_Y^{-1/2} - \boldsymbol{\Sigma}_{\hat{Y}}^{-1/2}\|_F). \tag{D.76}$$

It remains to bound $\|\boldsymbol{\Sigma}_{\hat{X}}^{-1/2} - \boldsymbol{\Sigma}_X^{-1/2}\|_F$, and similarly $\|\boldsymbol{\Sigma}_{\hat{Y}}^{-1/2} - \boldsymbol{\Sigma}_Y^{-1/2}\|_F$. We have that

$$\boldsymbol{\Sigma}_X^{-1/2} - \boldsymbol{\Sigma}_{\hat{X}}^{-1/2} = \boldsymbol{\Sigma}_X^{-1/2}(\boldsymbol{\Sigma}_X^{1/2} - \boldsymbol{\Sigma}_{\hat{X}}^{1/2})\boldsymbol{\Sigma}_{\hat{X}}^{-1/2}$$

$$= \mathbf{\Sigma}_X^{-1/2}(\mathbf{\Sigma}_X - \mathbf{\Sigma}_{\hat{X}})(\mathbf{\Sigma}_X^{1/2} + \mathbf{\Sigma}_{\hat{X}}^{1/2})^{-1}\mathbf{\Sigma}_{\hat{X}}^{-1/2}. \tag{D.77}$$

Hence, we have that

$$\begin{aligned}\|\mathbf{\Sigma}_X^{-1/2} - \mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F &\leq \|\mathbf{\Sigma}_X^{-1/2}\|_F \|\mathbf{\Sigma}_X - \mathbf{\Sigma}_{\hat{X}}\|_F \|(\mathbf{\Sigma}_X^{1/2} + \mathbf{\Sigma}_{\hat{X}}^{1/2})^{-1}\|_F \|\mathbf{\Sigma}_{\hat{X}}^{-1/2}\|_F \\ &\leq c\epsilon / \sqrt{\sigma_{d_X}(\mathbf{\Sigma}_X)}. \end{aligned} \tag{D.78}$$

Similarly, we have that $\|\mathbf{\Sigma}_Y^{-1/2} - \mathbf{\Sigma}_{\hat{Y}}^{-1/2}\|_F \leq c\epsilon$. Substituting in (D.76) we get that

$$\|\mathbf{\Lambda} - \hat{\mathbf{\Lambda}}\|_F \leq c\epsilon. \tag{D.79}$$

□

D.7 Proof of Theorem 19

Theorem 19. *Let $X \in \mathbb{R}^{d_X}$ and $Y \in \mathbb{R}^{d_Y}$ be a pair of jointly singular Gaussian random vectors. Then the common information between the quantized $\langle X \rangle_m$ and $\langle Y \rangle_m$ satisfies*

$$\lim_{m\to\infty} \frac{C(\langle X \rangle_m, \langle Y \rangle_m)}{\log m} = d(X, Y), \tag{5.34}$$

where $d(X, Y)$ is the common information dimension of X and Y with respect to the class of linear functions.

Proof. Since it is hard in general to directly solve Wyner's common information even for discrete variables, we prove the result by matching upper and lower bounds. First, we show that $\lim_{m\to\infty} \frac{C(\langle X \rangle_m, \langle Y \rangle_m)}{\log m} \geq d(X, Y)$ using the property that mutual information is a lower bound of Wyner's common information [Wyn75]. Following the definition of mutual information and the

Rényi dimension in (5.6), we can derive that

$$\begin{aligned}
\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} &\geq \lim_{m\to\infty} \frac{I(\langle X\rangle_m; \langle Y\rangle_m)}{\log m} \\
&= \lim_{m\to\infty} \frac{H(\langle X\rangle_m)}{\log m} + \lim_{m\to\infty} \frac{H(\langle Y\rangle_m)}{\log m} - \lim_{m\to\infty} \frac{H(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \\
&= d^R(X) + d^R(Y) - d^R(X, Y) \\
&\overset{\text{[Ren59]}}{=} \text{rank}(\boldsymbol{\Sigma}_X) + \text{rank}(\boldsymbol{\Sigma}_Y) - \text{rank}(\boldsymbol{\Sigma}) \\
&= d(X, Y),
\end{aligned} \tag{D.80}$$

where the last equality is proved in our Theorem 14.

Next, we show that $\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \leq d(X, Y)$. Recall (from the proofs of Lemma 11 and 12) that there exist invertible linear transformations $\boldsymbol{T}_X$ and $\boldsymbol{T}_Y$ such that $X = \boldsymbol{T}_X X'$ and $Y = \boldsymbol{T}_Y Y'$, where $X' \in \mathbb{R}^{d_X}$ and $Y' \in \mathbb{R}^{d_Y}$ are another pair of Gaussian random vectors with independent elements (i.e. their covariance matrices $\boldsymbol{\Sigma}_{X'}$, $\boldsymbol{\Sigma}_{Y'}$, and $\boldsymbol{\Sigma}_{X'Y'}$ are all diagonal). Note that $C(X, Y) = C(X', Y')$ and $d(X, Y) = d(X', Y')$ since $\boldsymbol{T}_X$ and $\boldsymbol{T}_Y$ are invertible transformations. However, $C(\langle X\rangle_m, \langle Y\rangle_m) \neq C(\langle X'\rangle_m, \langle Y'\rangle_m)$, because $\langle \boldsymbol{T}_X \langle X'\rangle_{m'}\rangle_m \neq \langle \boldsymbol{T}_X X'\rangle_m = \langle X\rangle_m$ can happen (similarly for Y' and Y).

In the remaining of the proof, we will first show in Proposition 5 that $\lim_{m\to\infty} \frac{C(\langle X'\rangle_m, \langle Y'\rangle_m)}{\log m} \leq d(X', Y')$ because of the special structure of the covariance matrices $\boldsymbol{\Sigma}_{X'}$ and $\boldsymbol{\Sigma}_{Y'}$. Then we will show that $\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \leq \lim_{m\to\infty} \frac{C(\langle X'\rangle_m, \langle Y'\rangle_m)}{\log m}$.

Proposition 5. *Let $\{(X_i, Y_i)\}_n$ be independent pairs of Gaussian random variables with $n < \infty$, then*

$$\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \leq d(X, Y) \tag{D.81}$$

Proof.

$$\begin{aligned}\lim_{m\to\infty}\frac{C(\langle X\rangle_m,\langle Y\rangle_m)}{\log m} &\overset{(i)}{=} \lim_{m\to\infty}\frac{\sum_{i=1}^n C(\langle X_i\rangle_m,\langle Y_i\rangle_m)}{\log m}\\ &= \lim_{m\to\infty}\frac{\sum_{i:X_i=Y_i} C(\langle X_i\rangle_m,\langle Y_i\rangle_m)+\sum_{i:X_i\neq Y_i} C(\langle X_i\rangle_m,\langle Y_i\rangle_m)}{\log m}\\ &\overset{(ii)}{\leq} \lim_{m\to\infty}\frac{\sum_{i:X_i=Y_i} H(\langle X_i\rangle_m)+\sum_{i:X_i\neq Y_i} c}{\log m}\\ &\overset{(iii)}{=} d(X,Y),\end{aligned} \tag{D.82}$$

where the c is an absolute constant. The equality (i) is because $\{(X_i,Y_i)\}_n$ are independent pairs; the inequality (ii) is due to the property that $C(\langle X_i\rangle_m,\langle Y_i\rangle_m)\leq\min\{H(\langle X_i\rangle_m),H(\langle Y_i\rangle_m)\}$ and (5.5); and the last equality (iii) follows the definition of Rényi dimension (5.6) and Theorem 14 $\square$

Let $m'=\alpha m$. To connect $C(\langle X\rangle_m,\langle Y\rangle_m)$ and $C(\langle X'\rangle_{\alpha m},\langle Y'\rangle_{\alpha m})$, we focus on two subsets $\mathcal{A}_m=\{\boldsymbol{x}\in\mathbb{R}^{d_X}:\langle \boldsymbol{T}_X\langle\boldsymbol{x}\rangle_{\alpha m}\rangle_m=\langle\boldsymbol{T}_X\boldsymbol{x}\rangle_m\}$ and $\mathcal{B}_m=\{\boldsymbol{y}\in\mathbb{R}^{d_Y}:\langle \boldsymbol{T}_Y\langle\boldsymbol{y}\rangle_{\alpha m}\rangle_m=\langle\boldsymbol{T}_Y\boldsymbol{y}\rangle_m\}$. We also define an indicator variable Z_m as follows:

$$Z_m=\begin{cases}1, & \text{if } X'\in\mathcal{A}_m \text{ and } Y'\in\mathcal{B}_m,\\ 0, & \text{otherwise.}\end{cases} \tag{D.83}$$

We consider the conditional common information [LW16] $C(\langle X\rangle_m,\langle Y\rangle_m|Z_m)$ that is defined as

$$C(\langle X\rangle_m,\langle Y\rangle_m|Z_m):=\min_{\langle X\rangle_m-(W_{Zm},Z_m)-\langle Y\rangle_m} I(\langle X\rangle_m\ \langle Y\rangle_m;W_{Zm}|Z_m). \tag{D.84}$$

We can use it to upper bound the common information as follows. Recall from (5.2), the common information is defined as

$$C(\langle X\rangle_m,\langle Y\rangle_m):=\min_{\langle X\rangle_m-W_m-\langle Y\rangle_m} I(\langle X\rangle_m\ \langle Y\rangle_m;W_m)$$

Let W^*_{Zm} be the optimal solution of $C(\langle X\rangle_m,\langle Y\rangle_m|Z_m)$. (W^*_{Zm},Z_m) is a feasible solution of $C(\langle X\rangle_m,\langle Y\rangle_m)$ that satisfies $\langle X\rangle_m-(W^*_{Zm},Z_m)-\langle Y\rangle_m$ and

$$\begin{aligned}I(\langle X\rangle_m\ \langle Y\rangle_m;W^*_{Zm},Z_m)&=I(\langle X\rangle_m\ \langle Y\rangle_m;Z_m)+I(\langle X\rangle_m\ \langle Y\rangle_m;W^*_{Zm}\mid Z_m)\\ &\leq H(Z_m)+I(\langle X\rangle_m\ \langle Y\rangle_m;W^*_{Zm}\mid Z_m).\end{aligned} \tag{D.85}$$

Therefore, we have

$$\begin{aligned} C(\langle X\rangle_m, \langle Y\rangle_m) \leq & C(\langle X\rangle_m, \langle Y\rangle_m | Z_m) + H(Z_m) \\ = & C(\langle X\rangle_m, \langle Y\rangle_m | Z_m = 1)\mathbb{P}(Z_m = 1) + C(\langle X\rangle_m, \langle Y\rangle_m | Z_m = 0)\mathbb{P}(Z_m = 0) \\ & + H(Z_m), \end{aligned} \tag{D.86}$$

where the equality expands the definition of conditional common information.

Since we are interested in the asymptotic behavior of $C(\langle X\rangle_m, \langle Y\rangle_m)$, we divide both sides of (D.86) by $\log m$ and take the limit of $m \to \infty$. In the following, we examine the limit of each term on the right-hand side. First, we have that

$$\begin{aligned} \lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m | Z_m = 1)\mathbb{P}(Z_m = 1)}{\log m} & \overset{(i)}{\leq} \lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m | Z_m = 1)}{\log m} \\ & \overset{(ii)}{=} \lim_{m\to\infty} \frac{C(\langle \boldsymbol{T}_X \langle X'\rangle_{\alpha m}\rangle_m, \langle \boldsymbol{T}_Y \langle Y'\rangle_{\alpha m}\rangle_m | Z_m = 1)}{\log m} \\ & \overset{(iii)}{\leq} \lim_{m\to\infty} \frac{C(\langle \boldsymbol{T}_X \langle X'\rangle_{\alpha m}\rangle_m, \langle \boldsymbol{T}_Y \langle Y'\rangle_{\alpha m}\rangle_m)}{\log m} \\ & \overset{(iv)}{\leq} \lim_{m\to\infty} \frac{C(\langle X'\rangle_{\alpha m}, \langle Y'\rangle_{\alpha m})}{\log m} \\ & \overset{\text{Proposition 5}}{\leq} d(X,Y), \end{aligned} \tag{D.87}$$

where the inequality (i) follows that $\mathbb{P}(Z_m = 1) \leq 1$; the equality (ii) is due to the fact that $Z_m = 1$ implies that $X' \in \mathcal{A}_m$ and $Y' \in \mathcal{B}_m$; the inequality (iii) is because if $\langle \boldsymbol{T}_X \langle X'\rangle_{\alpha m}\rangle_m - W - \langle \boldsymbol{T}_Y \langle Y'\rangle_{\alpha m}\rangle_m$ forms a Markov chain for some random variable W, then conditioned on the event that $Z_m = 1$, $\langle \boldsymbol{T}_X \langle X'\rangle_{\alpha m}\rangle_m - W - \langle \boldsymbol{T}_Y \langle Y'\rangle_{\alpha m}\rangle_m$ also forms a Markov chain (one proof can be found in Lemma 12 in [YT20]); the inequality (iv) follows the data processing inequality of common information and the last inequality applies Proposition 5. Using one of the basic properties of Wyner's common information and the definition of the Rényi dimension (5.6), we also have that

$$\begin{aligned} \lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m | Z_m = 0)}{\log m} & \leq \lim_{m\to\infty} \frac{\min\{H(\langle X\rangle_m), H(\langle Y\rangle_m)\}}{\log m} \\ & = \min\{d^R(X), d^R(Y)\}. \end{aligned} \tag{D.88}$$

And we upper bound the limit of the probability $\mathbb{P}(Z_m = 0)$ in Lemma 23 below, which we will prove after this Theorem.

Lemma 23. For random variables X, Y, X', Y' and set $\mathcal{A}_m, \mathcal{B}_m$ defined as above, we denote $t = \max\{\max\{|(\boldsymbol{T}_X)_{i,j}|, \forall i, j \in [d_X]\}, \max\{|(\boldsymbol{T}_Y)_{i,j}|, \forall i, j \in [d_Y]\}\}$, and $d = \max\{d_X, d_Y\}$. We have that

$$\lim_{m\to\infty} \mathbb{P}(Z_m = 0) \le \frac{4td^2}{\alpha}, \tag{D.89}$$

In addition, since Z_m is a binary variable, $H(Z_m) \le 1$ and $\lim_{m\to\infty} \frac{H(Z_m)}{\log m} = 0$. Combining with (D.86),(D.87),(D.88) and Lemma 23, we have that for all $\alpha > 0$

$$\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \le d(X, Y) + \min\{d^R(X), d^R(Y)\} \cdot \frac{4td^2}{\alpha} \tag{D.90}$$

As this holds for all $\alpha > 0$ taking the limit of α to infinity, we get $\lim_{\alpha\to\infty} \frac{4td^2}{\alpha} = 0$, and thus obtain that

$$\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} \le d(X, Y). \tag{D.91}$$

Therefore, combining with (D.80), we can conclude that

$$\lim_{m\to\infty} \frac{C(\langle X\rangle_m, \langle Y\rangle_m)}{\log m} = d(X, Y). \tag{D.92}$$

□

Lemma 23. For random variables X, Y, X', Y' and set $\mathcal{A}_m, \mathcal{B}_m$ defined as above, we denote $t = \max\{\max\{|(\boldsymbol{T}_X)_{i,j}|, \forall i, j \in [d_X]\}, \max\{|(\boldsymbol{T}_Y)_{i,j}|, \forall i, j \in [d_Y]\}\}$, and $d = \max\{d_X, d_Y\}$. We have that

$$\lim_{m\to\infty} \mathbb{P}(Z_m = 0) \le \frac{4td^2}{\alpha}, \tag{D.89}$$

Proof. Since the quantization is element-wise and $\langle X'_i\rangle_{\alpha m} = \frac{\lfloor \alpha m X'_i\rfloor}{\alpha m}$, we know that $|\langle X'_i\rangle_{\alpha m} - X'_i| \le \frac{1}{\alpha m}$. Now for all $i \in [d_X]$, we can bound the imprecision of transformation caused by quantization as follows:

$$\begin{aligned}|(\boldsymbol{T}_X < X' >_{\alpha m})_i - X_i| &= |(\boldsymbol{T}_X)_i(< X' >_{\alpha m} - X')| \\ &\leq \sum_{j=1}^{d_X} |(\boldsymbol{T}_X)_{ij}|| < X'_j >_{\alpha m} - X'_j| \\ &\leq \frac{td_X}{\alpha m} \leq \frac{td}{\alpha m}.\end{aligned} \tag{D.93}$$

Similarly, for all $i \in [d_Y]$, we can bound $|(\boldsymbol{T}_Y < Y' >_{\alpha m})_i - Y_i| \leq \frac{td}{\alpha m}$.

Let $\{\frac{k}{m}\}_{k\in\mathbb{Z}}$ denote the sequence of quantization boundaries of $\langle X_i\rangle_m$ and $\langle Y_i\rangle_m$. We define the set $\mathcal{C}_m = \{x \in \mathbb{R} : \exists k \in \mathbb{Z},\ \text{s.t. } x \in [\frac{k}{m} + \frac{td}{\alpha m}, \frac{k+1}{m} - \frac{td}{\alpha m}]\}$. We observe that if $X_i \in \mathcal{C}_m$ for some $k \in \mathbb{Z}$, then

$$(\boldsymbol{T}_X < X' >_{\alpha m})_i - \frac{k}{m} \geq X_i - \frac{k}{m} - |(\boldsymbol{T}_X < X' >_{\alpha m})_i - X_i| \geq 0, \tag{D.94}$$

where the second inequality is due to the assumption that $X_i \in [\frac{k}{m} + \frac{td}{\alpha m}, \frac{k+1}{m} - \frac{td}{\alpha m}]$ and (D.93). Similarly $\frac{k+1}{m} - (\boldsymbol{T}_X < X' >_{\alpha m})_i \geq 0$, which implies that $\langle(\boldsymbol{T}_X\langle X'\rangle_{\alpha m})_i\rangle_m = \langle(\boldsymbol{T}_X X')_i\rangle_m$. Therefore, we have that $\{\forall i \in [d_X], X_i \in \mathcal{C}_m \text{ and } \forall i \in [d_Y], Y_i \in \mathcal{C}_m\} \subset \{X' \in \mathcal{A}_m \text{ and } Y' \in \mathcal{B}_m\} = \{Z_m = 1\}$, and

$$\begin{aligned}\mathbb{P}(Z_m = 0) &\leq \mathbb{P}(\{\cup_{i\in[d_X]} X_i \in \mathcal{C}_m^c\} \cup \{\cup_{i\in[d_Y]} Y_i \in \mathcal{C}_m^c\}) \\ &\leq \sum_{i=1}^{d_X} \mathbb{P}(X_i \in \mathcal{C}_m^c) + \sum_{i=1}^{d_Y} \mathbb{P}(Y_i \in \mathcal{C}_m^c)\end{aligned} \tag{D.95}$$

Taking X_1 as an example, we next show that $\lim_{m\to\infty} \mathbb{P}(X_1 \in \mathcal{C}_m^c) \leq \frac{2td}{\alpha}$. For $k \in \mathbb{Z}$, we use $\bar{f}_k(X_1)$ and $\underline{f_k}(X_1)$ denote the maximum and minimum values of the probability density $f(X_1)$ in the range $[\frac{k}{m}, \frac{k+1}{m}]$. We can write each term in the summation as

$$\begin{aligned}\mathbb{P}(X_1 \in \mathcal{C}^c) &= \sum_{k\in\mathbb{Z}} \mathbb{P}(X_1 \in \mathcal{C}^c | X_1 \in [\frac{k}{m}, \frac{k+1}{m}])\mathbb{P}(X_1 \in [\frac{k}{m}, \frac{k+1}{m}]) \\ &= \sum_{k\in\mathbb{Z}} \frac{\mathbb{P}(X_i \in [\frac{k}{m}, \frac{k}{m} + \frac{td}{\alpha m}] \cup [\frac{k+1}{m} - \frac{td}{\alpha m}, \frac{k+1}{m}])}{\mathbb{P}(X_i \in [\frac{k}{m}, \frac{k+1}{m}])}\mathbb{P}(X_i \in [\frac{k}{m}, \frac{k+1}{m}]) \\ &\leq \sum_{k\in\mathbb{Z}} \frac{2tdm \cdot \bar{f}_k(X_1)}{\alpha m \cdot \underline{f_k}(X_1)}\mathbb{P}(X_i \in [\frac{k}{m}, \frac{k+1}{m}]).\end{aligned} \tag{D.96}$$

Since $f(X_1)$ is a continuous function, $\lim_{m\to\infty} \frac{\bar{f}_k(X_1)}{\underline{f_k}(X_1)} = 1$ for all $k \in \mathbb{Z}$. Then we can bound the limit as $\lim_{m\to\infty} \mathbb{P}(X_i \in \mathcal{C}^c) \le \frac{2td}{\alpha} \sum_{k\in\mathbb{Z}} \mathbb{P}(X_i \in [\frac{k}{m}, \frac{k+1}{m}]) = \frac{2td}{\alpha}$. Similar computation extends to all elements of X and Y. Plugging this result into (D.95), we can conclude that

$$\lim_{m\to\infty} \mathbb{P}(Z_m = 0) \le (d_X + d_Y)\frac{2td}{\alpha} = \frac{4td^2}{\alpha}. \tag{D.97}$$

□

APPENDIX E

Appendix for Chapter 6: Distributed Feature Compression for Classification

E.1 On-the-Line Optimality

In this appendix, we prove the optimality of Algorithm 10 under the considered restriction (horizontal and vertical lines defining $d_{k,i}$ meet along the line $x_1 = x_2$) and assuming that the data points are linearly separable and scaled such that the line $x_1 = x_2$ separates the data.

We prove by induction that $\forall k \in [1 : 2N]$: at the k-th iteration of the algorithm, it finds the optimal quantization boundaries considering only the points $\mathcal{T}_{s_k} = \{(\mathbf{x}, y(\mathbf{x})) \in \mathcal{T} | x_1, x_2 \leq s_k\}$, $\forall b \in [1 : 2^R - 1]$, where s_k is the k-th smallest element in the set of possible boundaries $\mathbf{s}$.

- At the first iteration, there is only one possible position for all the boundaries, s_1. Hence, at the first iteration the algorithm finds the optimal quantization boundaries considering the points $\mathcal{T}_{s_k}$, $\forall b \in [1 : 2^R - 1]$.

- Assuming that at iteration k the algorithm finds the optimal quantization boundaries considering the points $\mathcal{T}_{s_k}$, we show that it finds the optimal quantization boundaries considering the points $\mathcal{T}_{s_{k+1}}$ at iteration $k + 1$, $\forall b \in [1 : 2^R - 1]$.

  At iteration $k + 1$, the only possible positions for the first boundary before s_{k+1} are the k positions $s_1, s_2, ..., s_k$. Therefore, to find the optimal boundaries at iteration $k + 1$, we can condition on the position of the first boundary before s_{k+1}, and then optimize over this position. Conditioned on the position of the first boundary before s_{k+1} being at s_ℓ, the updated loss function can be expressed as

$$\begin{aligned}&\mathcal{L}_{B\cup s_\ell}(\mathcal{E},\mathcal{T}_{s_{k+1}})\\&=\mathcal{L}_B(\mathcal{E},\mathcal{T}_{s_\ell})+\min_{c\in\{1,2\}}|\{j|s_\ell\prec\mathbf{x}^{(j)}\preceq s_{k+1},\ y^{(j)}=c\}|,\end{aligned}$$

where $\mathcal{L}_B(\mathcal{E},\mathcal{T}_{s_\ell})$ is the loss $\mathcal{L}(\mathcal{E},\mathcal{D},\mathcal{T}_{s_\ell})$ when using the boundaries in the set B, and B is the set of boundaries in the region of the space defined by $\mathbf{x}\in\mathbb{R}^2:\mathbf{x}\preceq s_\ell$, $|B\cup s_\ell|=b$. This is, as we discussed in Chapter 6, due to the fact that the misclassified points contributing to the quantizer loss can only lie in the 2-dimensional intervals crossed by the line $x_1=x_2$. Which is because any other interval lies on one side of the line that separates the points, hence, contains points from only one class. Minimizing over all the possible values for s_ℓ, B we get

$$\begin{aligned}E(s_{k+1},b)=&\min_{B,\ell:|B|\le b-1,\ell<k+1}\{\mathcal{L}_B(\mathcal{E},\mathcal{T}_{s_\ell})\\&+\min_{c\in\{1,2\}}|\{j|s_\ell\prec\mathbf{x}^{(j)}\preceq s_{k+1},\ y^{(j)}=c\}|\}.\end{aligned}$$

We can observe that only the first term in the previous minimization depends on B, hence, we can optimize over B first, which gives

$$\begin{aligned}E(s_{k+1},b)=\min_{\ell<k+1}\Big\{&E(s_\ell,b-1)\\&+\min_{c\in\{1,2\}}|\{j|s_\ell\prec\mathbf{x}^{(j)}\preceq s_{k+1},\ y^{(j)}=c\}|\Big\},\end{aligned}$$

which is the update rule used in the algorithm. Hence, the boundaries corresponding to $E(s_{\ell^*},b-1)$ along with the boundary at s_{ℓ^*} are the optimal boundaries at iteration $k+1$, where

$$\begin{aligned}\ell^*=\arg\min_{\ell<k+1}\Big\{&E(s_\ell,b-1)\\&+\min_{c\in\{1,2\}}|\{j|s_\ell\prec\mathbf{x}^{(j)}\preceq s_{k+1},\ y^{(j)}=c\}|\Big\}.\end{aligned}$$

E.2 Discussion on Breaking Ties in `GBI`: The Purity Criterion

In this section we illustrate what we call the purity criterion which is used to break ties in `GBI`. If it happens that two or more possible boundaries lead to the same quantizer loss (something that happened surprisingly often in our experiments), it makes a significant performance difference to add the boundary that looks ahead to allow future boundaries to further decrease the loss. The intuition behind this is the following: for a tie, we have a fixed number of misclassified points; what matters for the algorithm performance is that the misclassified points are in bins that can be more easily partitioned to bins that contains no misclassification in a next iteration. This was more likely to happen in our experiments (and small examples) if a bin that has misclassified points contained a number of majority class points that was just slightly higher than that of the misclassified classes. Formally, let H be an $\mathbb{R}^n$-bin and let $\mathcal{T}$ be the set of training points $\{(\mathbf{x}^{(i)}, y(\mathbf{x}^{(i)})\}$. Define, $B(H,\mathcal{T}) := 0$, if if all points in H are of the same class. Otherwise, $B(H,\mathcal{T}) := \max_{c\in\mathcal{C}} |\{(\mathbf{x}, y(\mathbf{x}))\in\mathcal{T}\cap H \text{ s.t. } y(\mathbf{x})=c\}|$.

$B(H,\mathcal{T})$ counts the number of points of the **majority** class in H when there is at least two or more classes represented in H, and is zero otherwise. For a particular set of boundaries that partition $\mathbb{R}^n$ into the $\mathbb{R}^n$-bins $\{H_k\}_{k=1}^M$, we want to **minimize** the **purity criterion** defined as

$$U(\{H_k\}_{k=1}^M, \mathcal{T}) := \frac{\sum_{k=1}^M B(H_k,\mathcal{T})^2}{N},$$

where the term $B(H_k,\mathcal{T})^2$ penalizes bins with more majority points. $U(\{H_k\}_{k=1}^M, \mathcal{T})$ is minimized when the correctly classified points represent a weak majority in the bins that have misclassification. This allows for the bins that have misclassification to be easily partitioned to bins that have no misclassification in a following iteration. For illustration, consider the example shown in Fig. E.1. In this example all the potential boundaries ①, ②, ③ ,④ result in the same value for the quantizer loss. However, it is clear that, unlike boundaries ③,④, if we pick boundaries ①, ②, this allows for the separation of the "o" points (red) from the "x" points (blue) in the next iteration. The purity criterion chooses boundary ② as shown by the values in Fig. E.1. This choice separates the maximum number of "x" points from the bin that have misclassification. Note that, a power greater

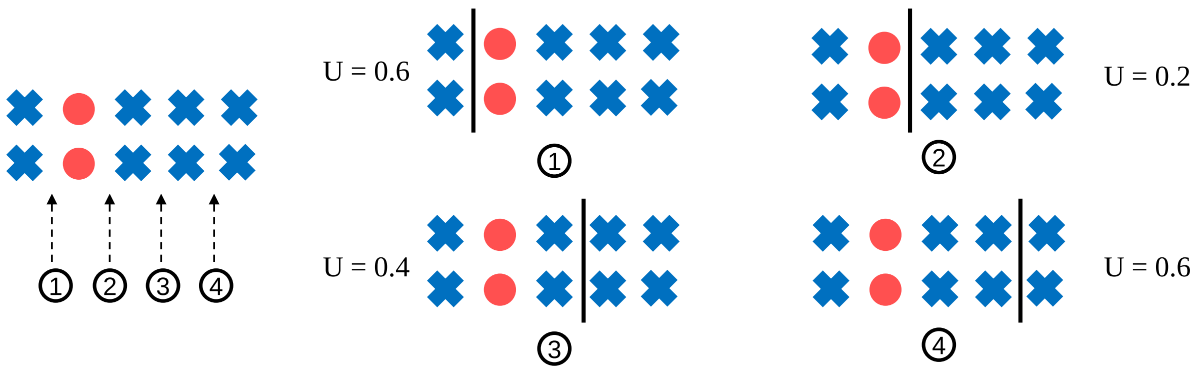


Figure E.1: Illustration of the purity criterion.

than 1, hence a function with slope that increases when we move away from zero, is needed in the purity criterion to highly penalize the bins that have misclassification having high concentration of majority class points, which prevents isolating the misclassified points in the following iterations. If we use a power of 1, the purity criterion is reduced to the quantizer loss. In our case we use a power of 2.

E.3 Parameters of Trained Models for Experimental Evaluation

In this appendix, we describe the structure of the encoders/decoders neural networks and parameters used in the experimental results for each dataset.

E.3.1 sEMG Dataset:

The structures of the neural networks used by the encoders $f_k(\cdot;\theta_k)$ and decoder $g(\cdot;\phi)$ are given in Table E.1.

- **General parameters.** The distributed quantization system was trained using Adam optimizer with learning rate 10^{-3} for 300 epochs. The parameters of the pretrained classifier C were frozen (not updated) during the learning phase.

- **NN-REG.** For our approach in Section 6.7.1, we chose the regularization parameter β through

Table E.1: Structure of the encoder neural networks $f_k(\cdot;\theta_k)$ and the decoder neural network $g(\cdot;\phi)$ used for the sEMG dataset.

Encoder neural network $f_k(\cdot;\theta_k),\ \forall k \in [1:4]$

Index	Layer Type	Output size
1	FC-Relu	90
2	FC-Relu	170
3	FC-Batchnorm-Tanh	# of bits per encoder

Decoder neural network $g(\cdot;\phi)$

Index	Layer Type	Output size
1	FC-Relu	170
2	FC-Relu	90
3	FC	8

5-fold cross validation out of possible parameter values $\{0, 0.1, \cdots, 2\}$. The chosen regularization parameter is $\beta = 1.4$.

- **NN-GBI.** For our approach in Section 6.7.2, the output of the last layer in the encoder was chosen to be 1 when applying **Phase 1** (refer to Section 6.7.2). When applying `GBI`, a batch size of 300 data points was used in each iteration of the `GBI` algorithm. After designing the quantizer with `GBI`, the network is trained for an extra 20 epochs with the quantizer in **Phase 3**.

E.3.2 CIFAR10 Dataset:

The structures of the neural networks used by the encoders $f_k(\cdot;\theta_k)$ and decoder $g(\cdot;\phi)$ are given in Table E.2 and Table E.3, respectively.

- **General parameters.** The distributed quantization system was trained using Adam optimizer with learning rate 10^{-3} for 200 epochs. The parameters of the pretrained classifier $\mathcal{C}$ were frozen (not updated) during the learning phase.

- **NN-REG.** The chosen regularization parameter is $\beta = 0.25$.

- **NN-GBI.** For our approach in Section 6.7.2, the output of the last layer in the encoder was chosen to be 5 when applying **Phase 1** (refer to Section 6.7.2). When applying `GBI`, a batch size of 300 data points was used in each iteration of the `GBI` algorithm. After designing the quantizer with `GBI`, the network is trained for an extra 50 epochs with the quantizer in **Phase 3**.

Table E.2: Structure of encoder neural networks $f_k(\cdot; \theta_k)$ used for CIFAR-10 dataset with R bits.

Index	Layer Type	Output size	Input channels	Output channels	Kernel/Stride/Pad
1	Conv-Relu	-	3	64	3/1/1
2	Conv-Relu	-	64	64	3/1/1
3	Maxpool	-	-	-	2/2/1
4	Conv-Relu	-	64	128	3/1/1
5	Conv-Relu	-	128	128	3/1/1
6	Maxpool	-	-	-	2/2/1
7	FC-Tanh	R	-	-	-

E.4 Proof of Theorem 20: NP-Hardness and Hardness of Approximation

NP-Hardness of (P1') and (P2')

In this subsection, we prove that (**P1'**), (**P2'**) are NP-hard. We start with (**P2'**). Since (**P2'**) is a special case of (**P1'**); it follows that (**P1'**) is NP-hard. We prove hardness of (**P2'**) by reduction from the Chromatic Number problem. In particular, we show that any instance of the Chromatic

Table E.3: Structure of decoder neural network $g(\cdot;\phi)$ used for CIFAR-10 dataset. We use the shorthand CT-R-B = ConvTranspose-Relu-Batchnorm.

Index	Layer Type	Output size	Input channels	Output channels	Kernel/Stride/Pad
1	FC-Relu	80	-	-	-
2	CT-R-B	-	5	5	4/2/1
3	CT-R-B	-	5	5	4/2/1
4	CT-R-B	-	5	5	4/2/1
5	Conv	-	5	3	5/1/2

Number problem can be reduced to an instance of problem (**P2'**) in polynomial time. The decision version of the Chromatic Number problem is on Karp's list of NP-complete problems [Kar72].

Let us consider an undirected graph G. We denote the set of vertices and edges of G by V, E respectively, and assume that the vertices are labeled by numbers $1, 2, ..., |V|$. Since the graph is undirected, we assume without loss of generality that the vertices pair corresponding to each edge is ordered such that if $(q_1, q_2) \in E$, then $q_1 > q_2$. We will see that this assumption, ensures that we can construct a linearly separable dataset as required in problem (**P2'**).

We construct two matrices $\{F_i\}_{i=0}^1, F_i \in \mathbb{R}^{|E|\times|V|}$ that represent the set of edges E, by Algorithm 13. We can think of the matrices $\{F_i\}_{i=0}^1$ as a decomposed version of the incidence matrix, where

Algorithm 13 Incidence matrices

```
0: Initialize the entries of $F_i \in \mathbb{R}^{|E|\times|V|}, i \in \{0, 1\}$ with all zeros. $k = 1$.
$\forall (q_1, q_2) \in E$ , do the following two steps:
    1: Put $[F_0]_{kq_1} = 1, [F_1]_{kq_2} = 1$.
    2: k=k+1.
```

for each edge, one endpoint is represented in F_0 and the other endpoint is represented in F_1. As an illustrative example, we consider the graph in Fig. E.2. The corresponding matrices are given by

$$F_0 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}, F_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{E.1}$$

If two vertices are colored with the same color, we update the matrices $\{F_i\}_{i=0}^{1}$ by replacing the columns that correspond to the vertices colored with the same color, with their sum. For example, if vertices $1, 2$ are assigned the same color, we update each matrix F_i by replacing the first two columns with their sum, which results in

$$F_0^{'} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, F_1^{'} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \tag{E.2}$$

We notice that **a coloring is valid** (no two vertices connected with an edge are assigned the same color) if and only if the updated matrices satisfy that $\forall k \in [1 : |E|]$, $\forall q \in [1 : V']$, $\forall i \in \{0, 1\}$, we have

$$[F_i^{'}]_{kq} \neq 0 \implies [F_{1-i}^{'}]_{kq} = 0, \tag{E.3}$$

where V' is the number of columns of the matrix $F_0^{'}$ or $F_1^{'}$. In the above example, the property in

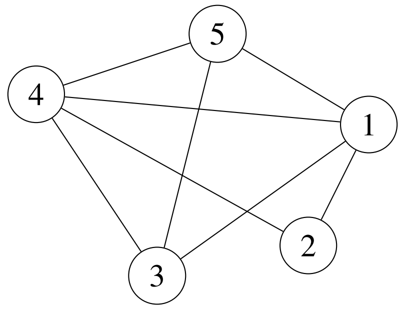


Figure E.2: Graph with edges represented by the matrices in (E.1).

(E.3) is not satisfied since, $[F_i^{'}]_{80} = 1, [F_{1-i}^{'}]_{81} = 1$. This is because the vertices $1, 2$, which are assigned the same color, are connected with an edge.

Hence, the Chromatic Number of the graph is the minimum number of columns of matrices $\{F_i^{'}\}_{i=0}^{1}$ that satisfy the property in (E.3) and are constructed according to the following rules:

- Any set of columns in the matrix F_i can be replaced by their sum, $i = 0, 1$.
- If the set of columns indexed by $\mathcal{I}$ in F_i are replaced with their sum, then the set of columns indexed by $\mathcal{I}$ in F_{1-i} are replaced with their sum, $i = 0, 1$, i.e., exactly the same operations done on F_0 are done on F_1 and vice versa.

The next step is to consider an instance from the problem (**P2'**), and show that it is equivalent to the problem of finding the minimum number of columns of the matrices $\{F_i^{'}\}_{i=0}^{1}$. To that end, we consider a dataset, with two classes, namely $\mathcal{Y} = \{0, 1\}$, and two features x_1, x_2. The dataset is constructed based on the matrices $\{F_i\}_{i=0}^{1}$ by Algorithm 14.

Algorithm 14 Reduction from vertex coloring problem

```
0: Start with $k = 1$.
$\forall (q_1, q_2) \in E$ , do the following two steps:
    1: Put a training point that belong to class 0 at $x_1 = q_1, x_2 = \frac{q_1+q_2}{2}$,
       and a training point that belong to class 1 at $x_1 = q_2, x_2 = \frac{q_1+q_2}{2}$.
    2: k=k+1.
```

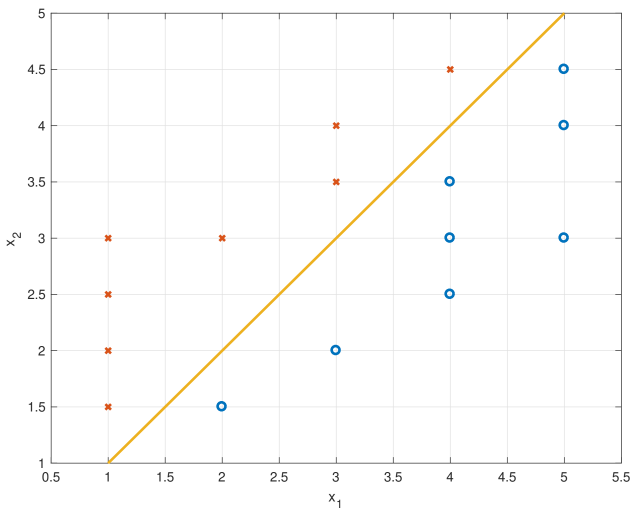


Figure E.3: Training data corresponding to $\{F_i\}$ in (E.1).

Note that under the assumption $q_1 > q_2$, the point $(x_1, x_2) = (q_1, \frac{q_1+q_2}{2})$ lies on the right side of the line $x_1 = x_2$, while the point $(x_1, x_2) = (q_2, \frac{q_1+q_2}{2})$ lies on the left side of the line $x_1 = x_2$. Hence, all the points that belong to class 0 lie on the right side of the line $x_1 = x_2$, and all the points that belong to class 1 lie on the left side of the line $x_1 = x_2$. That is, the constructed dataset is linearly separable by the line $x_1 = x_2$.

For the constructed dataset, we want to answer the following question: for $R_2 \to \infty$, what is the minimum R_1 for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{2}$?

As an example, consider the matrices $\{F_i\}_{i=0}^{1}$ in (E.1), the constructed dataset is given in Fig. E.3.

Assume that we want to find the minimum R_1 for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{2}$. If $2^{R_1} \geq 5$, then we do not need to do quantization and can send x_1 as it is. In this case we have $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) = 0$. Now, assume that $2^{R_1} = 4$, then we have only 4 values to send to represent x_1, i.e., $\mathcal{E}_1(x_1) \in [1:4]$. Hence, the quantizer has to map two different values of x_1 to the same quantized value, i.e., $\exists x_1^{(1)}, x_1^{(2)} : x_1^{(1)} \neq x_1^{(2)}, \mathcal{E}_1(x_1^{(1)}) = \mathcal{E}_1(x_1^{(2)})$. The matrices $\{F_i'\}$ are constructed from $\{F_i\}$ such that if the encoder maps a set of x_1 values to the same encoded value, we replace the corresponding columns with their sum. For instance, assume that the encoder maps the values

$x_1 = 1, x_1 = 2$ to the same quantized value. Based on this, we update each matrix F_i by replacing the first two columns with their sum, which results in the matrices in (E.2). Since this encoder maps the two points $(1, 1.5), (2, 1.5)$, which belong to different classes, to the same encoded value, we have $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) = \sum_{k=1}^{l_2} \sum_{q=1}^{l_1-1} \min_j\{[F_j^{'}]_{kq}\} = 1 > \frac{1}{2}$. So, this encoder does not satisfy $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{2}$. The reason for this is that the updated matrices do not satisfy: $\forall k \in [1 : |E|] \forall q \in [1 : |V'|] \forall i \in \{0, 1\}\ [F_i^{'}]_{kq} \neq 0 \implies [F_{1-i}^{'}]_{kq} = 0$. Note that the matrices constructed by Algorithm 13 satisfy the following properties

- Every row in F_i has exactly one non-zero entry.
- If $[F_i]_{kq} \neq 0$, then $[F_{1-i}]_{kq} = 0$.
- All the non-zero entries have value 1.

It is easy to observe that for all matrices satisfying the three properties mentioned above, $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{2}$ if and only if the encoder $\mathcal{E}$ satisfies:

$$\begin{aligned}
&\forall q_1 \forall q_2 : q_1, q_2 \in \text{dom}(\mathcal{E}_1), q_1 \neq q_2, \mathcal{E}_1(q_1) = \mathcal{E}_1(q_2),\\
&\forall k \in [1 : |E|], \forall i \in \{0, 1\}, \text{ we have that,}\\
&[F_i]_{kq_1} + [F_i]_{kq_2} \neq 0 \quad \implies [F_{1-i}]_{kq_1} + [F_{1-i}]_{kq_2} = 0.
\end{aligned}$$

Hence, the $\min\{2^{R_1} | \mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{2}\}$ is the minimum number of columns of matrices $\{F_i^{'}\}_{i=0}^{1}$ that satisfy the property in (E.3) and are constructed according to the following rules:

- Any set of columns in the matrix F_i can be replaced by their sum, $i = 0, 1$.
- If the set of columns indexed by $\mathcal{I}$ in F_i are replaced with their sum, then the set of columns indexed by $\mathcal{I}$ in F_{1-i} are replaced with their sum, $i = \{0, 1\}$, i.e., exactly the same operations done on F_0 are done on F_1 and vice versa.

This shows that $\mathcal{X}(G) = \min\{2^{R_1} | \mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}_G) < \frac{1}{2}\}$, where $\mathcal{X}(G)$ is the chromatic number of the graph G, and $\mathcal{T}_G$ is the dataset constructed by Algorithm 14.

Note that the maximum number of x_1 values that are encoded to the same value is $|V|$, i.e., $\mathcal{E}_1(\mathbf{z}_1)$ is the union of at most $|V|$ $\mathbb{R}$-bins. Hence, r is chosen to be $r = |V|$. This concludes the proof that problem (**P2'**) is NP-hard.

NP-hardness of (P3'):

In this subsection, we prove that (**P3'**) is NP-hard. For reference, we restate the statement of (**P3'**) below:

(P3') : For $n = 2$ features, $|\mathcal{Y}| = 2$ classes, find the minimum $R_1 = R_2$ bits for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < p_e$, assuming that $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is a single $\mathcal{X}_{\Omega_k}$-bin.

To prove that (**P3'**) is NP-hard, we show that the maximum Balanced Complete Bipartite Subgraph (BCBS) problem [ADL94] can be reduced to (**P3'**) in polynomial-time. The maximum BCBS problem is defined below.

Definition 2. (maximum BCBS) Given a balanced bipartite graph $G = (V_1, V_2, E)$: find the maximum size, in terms of number of vertices, of a balanced bi-clique in G[1].

The maximum BCBS problem is known to be NP-hard as proved in [ADL94].

As an example, we first consider an instance of (**P3'**) and show that it is equivalent to an instance of the BCBS problem. Then, we show how to reduce any instance of the BCBS problem to an instance of (**P3'**) in polynomial time.

Consider the training data shown in Fig. E.4 for which we want to answer the question in (**P3'**). If two values of x_i are encoded to the same message we say that they are **combined**. To minimize R_1, R_2, we want to combine as many values of x_1, x_2 as possible while satisfying $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{N}$.

We notice the following:

[1]By a balanced bipartite graph G, we mean that $|V_1| = |V_2|$.

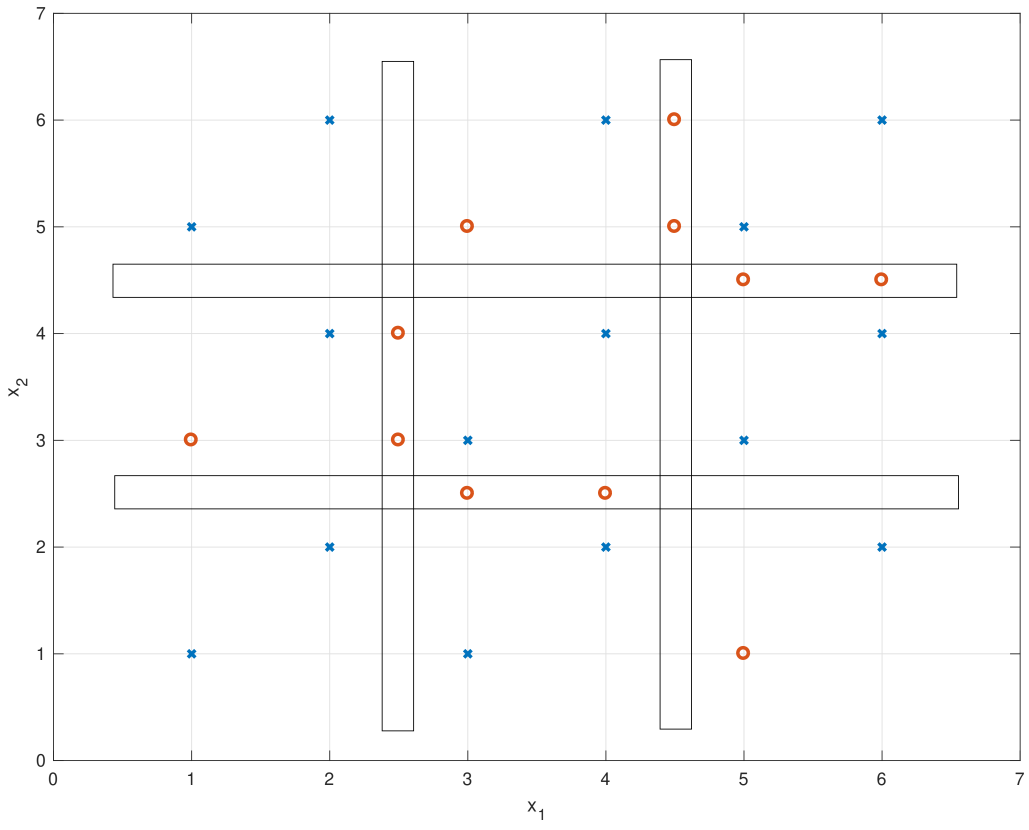


Figure E.4: Training data used to prove (**P3'**).

1. The only way to satisfy $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{N}$ is to make $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) = 0$. Hence, we can combine values of x_1, x_2 as long as all the bins only have training points that belong to the same class.

2. Since (**P3'**) assumes $\mathcal{E}_k^{-1}(\mathbf{z}_k)$ is a single $\mathcal{X}_{\Omega_k}$-bin, then quantization of x_1 (resp., x_2) in $\mathcal{T}_G$ is equivalent to combining adjacent columns (resp., rows) in the dataset. Combining non-adjacent columns (resp., rows) is not allowed as it will not result in a single bin as assumed by (**P3'**). Recall that we say that two values of x_1 (resp., x_2), i.e., columns (resp., rows) are **combined**, if they are assigned to the same $\mathbf{z}_1$ (resp., $\mathbf{z}_2$). For example, we can combine the values $x_1 = 1, x_1 = 2, x_1 = 2.5$, but we cannot combine $x_1 = 1, x_1 = 2.5$ without combining $x_1 = 2$ with them, because $x_1 = 1, x_1 = 2.5$ are not adjacent.

3. From Fig. E.4, we notice that the values of $x_i \notin \mathbb{Z}$, cannot be combined with any adjacent value of $x_i, \forall i \in \{1, 2\}$, because combining them results in bins having points that belong to different classes. These values also prevent us from combining any two values of x_i that are on different sides from a value $x_i \notin \mathbb{Z}$, since the combined values has to be adjacent. For this reason, we call the training points with feature values $x_i \notin \mathbb{Z}$ (surrounded by rectangles in the figure), **guard points**. Having this, the only possible values to combine are $x_i = a, x_i = a+1$ for some odd number a. Hence, the problem reduces to deciding whether $x_i = a$ should be combined with $x_i = a + 1$ or not $\forall a \in \{1, 3, 5\}$.

We can also notice from the figure that if we, for example, combine the values $x_1 = 1, x_1 = 2$, then we cannot combine the values $x_2 = 3, x_2 = 4$, because this results in the two training points $(1, 3), (2, 4)$, which belong to different classes, being assigned to the same bin. Now it is clear that this instance of the problem of finding the minimum number of bins for which $\mathcal{L}(\mathcal{E}, \mathcal{D}, \mathcal{T}) < \frac{1}{N}$ is equivalent to finding the maximum size of the balanced complete bipartite subgraph for the graph $\mathcal{G}$ constructed in the following way:

- There is a vertex in the upper side of the bipartite graph corresponding to each value $x_1 = a, \forall a \in \{1, 3, 5\}$, and a vertex in the lower side corresponding to each value $x_2 = b, \forall b \in \{1, 3, 5\}$.
- There is an edge between the vertex corresponding to $x_1 = a$ in the upper side and the vertex corresponding $x_2 = b$ in the lower side if and only if the two training data points $(a, b), (a + 1, b + 1)$ belong to the same class.

Fig. E.5 shows the constructed graph corresponding to the dataset in Fig. E.4. Combining the values $x_1 = a, x_1 = a + 1$ corresponds to having the vertex corresponding to $x_1 = a$ belonging to the maximum balanced complete bipartite subgraph. For instance, as we mentioned above combining the values $x_1 = 1, x_1 = 2$, implies that we cannot combine the values $x_2 = 3, x_2 = 4$; similarly in the graph in Fig. E.5, having the vertex corresponding to $x_1 = 1$ in the set of vertices of a balanced

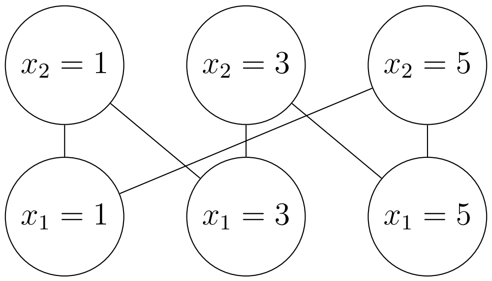


Figure E.5: Equivalent BCBS problem graph for the training data in Fig. E.4.

complete bipartite subgraph implies that the vertex corresponding to $x_2 = 3$ does not belong to that set.

Next, we describe an algorithm to reduce any instance from the BCBS problem to (**P3'**). We assume without loss of generality that every vertex is connected to at least one other vertex. We discuss at the end of this subsection how to deal with vertices that are not connected to any other vertex. We also assume that the vertices of the upper and lower side of the bipartite graph are labeled by numbers from $1, ..., M_i$, where M_1, M_2 are the number of vertices in the upper and lower side of the bipartite graph respectively. The pseudo-code is provided in Algorithm 15.

Applying Algorithm 15 on the graph in Fig. E.5, results in the training data plotted in Fig. E.4.

As explained in the example above, by construction of the guard points in Step 4 in Algorithm 15, they cannot be combined with any adjacent points. Moreover they prevent combining any points except for $x_i = a, x_i = a + 1$ for some odd a. In addition, as also explained in the example above, by construction of the points in steps 2, 3, vertices i, j can both belong to a biclique if and only if both $x_1 = 2i - 1, x_1 = 2i$ and $x_2 = 2j - 1, x_2 = 2j$ can be combined. This implies that any biclique of size W implies a quantization scheme with rate $2^{R_1} = 2^{R_2} = 3|V| - 1 - W$, where $|V|$ is the number of vertices in the graph (note that the guard points add $|V| - 1$ values to each axis) and vice versa. Hence, the two problems are equivalent.

If there is a vertex that is not connected to any other vertex in the graph, it is easy to add points (similar to the guard points) that cannot be combined with any adjacent points to represent such vertex. This concludes the proof of NP-hardness of (**P3'**).

Algorithm 15 Reduction from the BCBS problem to (**P3'**)

```
1: for all pairs of vertices pair of vertices in different sides $(i, j) \in [1 : M_1] \times [1 : M_2]$ do
2:     If there is an edge connecting $i, j$: put two training points that belong to class $0$ one at $(2i - 1, 2j - 1)$ and one at $(2i, 2j)$.
3:     If no edge between $i, j$: put a point from class $1$ at $(2i - 1, 2j - 1)$ and a point from class $0$ at $(2i, 2j)$.
4: Put the guard points as follows:
5: for all vertices $i \in [2 : M_1]$ in the upper part of the graph do
6:     Find any vertex $j \in [1 : M_2]$ from the lower side which is connected with vertex $i^2$.
7:     Put two points that belong to class $1$, one at $(2i - 1.5, 2j - 1)$ and another at $(2i - 1.5, 2j)$.
8: Similarly:
9: for all vertices $j \in [2 : M_2]$ in the lower part of the graph do
10:     Find any vertex $i \in [1 : M_1]$ from the upper side which is connected with vertex $j$.
11:     Put two point that belong to class $1$, one at $(2i - 1, 2j - 1.5)$ and another at $(2i, 2j - 1.5)$.
```

NP-Hardness of (P4')

To prove (**P4'**), we notice the following. Consider a training dataset with $n = 3$ features, where the training data points have arbitrary values for features x_1, x_2, while feature x_3 takes values $x_3 = -1$ for all points that belong to class 0 and $x_3 = 1$ for all points that belong to class 1. Such training data points are linearly separable by the hyperplane $x_3 = 0$. However, if $R_3 = 0$, we cannot send any information based on feature 3, hence, we have to only consider the projection of the training data on features x_1, x_2 ignoring feature x_3. This shows that problem (**P3'**) is a special case of problem (**P4'**). For instance, we can have a training dataset that have values for features x_1, x_2 generated based on the method used to prove (**P3'**), while feature x_3 takes $x_3 = -1$ for all points that belong to class 0 and $x_3 = 1$ for all points that belong to class 1. It follows that problem (**P4'**) is NP-hard.

Hardness of Approximation

From the proof of (**P1'**),(**P2'**), we can see that a polynomial-time algorithm that approximates 2^{R_1} for problem (**P1'**) or (**P2'**) within $O(N^{1-\epsilon})$ for some $\epsilon > 0$ can be used to approximate the chromatic number within $O(N^{1-\epsilon})$ in polynomial time, since the chromatic number problem is polynomial-time reducible to problem (**P1'**) or (**P2'**) with $\mathcal{X}(G) = \min 2^{R_1}$, and number of training points equal to the number of vertices in the graph. However, it was shown in [Zuc06] that approximating the chromatic number within $O(N^{1-\epsilon})$ is NP-hard $\forall \epsilon > 0$.

Similarly, from the proof of (**P3'**),(**P4'**), we can see that a polynomial-time algorithm that approximates 2^R for problem (**P3'**) or (**P4'**) within a factor of $O(N^{\frac{1}{2}-\epsilon})$ for some $\epsilon > 0$ can be used to approximate the BCBS within a factor of $O(N^{1-\epsilon})$ in polynomial time. The fraction $\frac{1}{2}$ in the exponent is because the BCBS problem is polynomial-time reducible to problem (**P3'**) or (**P4'**) with number of training points in the order of the square of the number of vertices in the graph. It was shown in [Man17] that approximating the BCBS within $O(N^{1-\epsilon})$ is NP-hard $\forall \epsilon > 0$ assuming the *Small Set Expansion Hypothesis* (SSEH) and that NP $\nsubseteq$ BPP.

REFERENCES

[AAA17] Arpit Agarwal, Shivani Agarwal, Sepehr Assadi, and Sanjeev Khanna. “Learning with limited rounds of adaptivity: Coin tossing, multi-armed bandits, and ranking from pairwise comparisons.” In *Conference on Learning Theory*, pp. 39–75. PMLR, 2017.

[AB09] Jean-Yves Audibert, Sébastien Bubeck, et al. “Minimax Policies for Adversarial and Stochastic Bandits.” In *COLT*, volume 7, pp. 1–122, 2009.

[ABC16] Alekh Agarwal, Sarah Bird, Markus Cozowicz, Luong Hoang, John Langford, Stephen Lee, Jiaji Li, Dan Melamed, Gal Oshri, Oswaldo Ribas, et al. “Making contextual decisions with low technical debt.” *arXiv preprint arXiv:1606.03966*, 2016.

[AC93] Rudolf Ahlswede and Imre Csiszár. “Common randomness in information theory and cryptography. I. Secret sharing.” *IEEE Transactions on Information Theory*, **39**(4):1121–1132, 1993.

[AC98] Rudolf Ahlswede and Imre Csiszár. “Common randomness in information theory and cryptography. II. CR capacity.” *IEEE Transactions on Information Theory*, **44**(1):225–240, 1998.

[ACE08] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, Nitin Motgi, Seung-Taek Park, Raghu Ramakrishnan, Scott Roy, and Joe Zachariah. “Online models for content optimization.” *Advances in Neural Information Processing Systems*, **21**, 2008.

[ACF02] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. “Finite-time analysis of the multi-armed bandit problem.” *Machine learning*, **47**(2):235–256, 2002.

[ACT18] Jayadev Acharya, Clément L Canonne, and Himanshu Tyagi. “Distributed Simulation and Distributed Inference.” *arXiv preprint arXiv:1804.06952*, 2018.

[ADL94] Noga Alon, Richard A Duke, Hanno Lefmann, Vojtech Rodl, and Raphael Yuster. “The algorithmic aspects of the regularity lemma.” *Journal of Algorithms*, **16**(1):80–109, 1994.

[AFD17] Alex Alemi, Ian Fischer, Josh Dillon, and Kevin Murphy. “Deep Variational Information Bottleneck.” In *ICLR*, 2017.

[AG12] Shipra Agrawal and Navin Goyal. “Analysis of thompson sampling for the multi-armed bandit problem.” In *Conference on learning theory*, pp. 39–1. JMLR Workshop and Conference Proceedings, 2012.

[AG13a] Shipra Agrawal and Navin Goyal. “Further optimal regret bounds for thompson sampling.” In *Artificial intelligence and statistics*, pp. 99–107. PMLR, 2013.

[AG13b] Shipra Agrawal and Navin Goyal. "Thompson sampling for contextual bandits with linear payoffs." In *International conference on machine learning*, pp. 127–135. PMLR, 2013.

[AGL17] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. "QSGD: Communication-efficient SGD via gradient quantization and encoding." *Advances in Neural Information Processing Systems*, **30**, 2017.

[Agr95] Rajeev Agrawal. "Sample mean based index policies with O (log n) regret for the multi-armed bandit problem." *Advances in Applied Probability*, pp. 1054–1078, 1995.

[AK08] Baruch Awerbuch and Robert Kleinberg. "Online linear optimization and adaptive routing." *Journal of Computer and System Sciences*, **74**(1):97–114, 2008.

[AL99] Naoki Abe and Philip M Long. "Associative reinforcement learning using linear probabilistic concepts." In *ICML*, pp. 3–11. Citeseer, 1999.

[AL17] Marc Abeille and Alessandro Lazaric. "Linear thompson sampling revisited." In *Artificial Intelligence and Statistics*, pp. 176–184. PMLR, 2017.

[AMT11] Animashree Anandkumar, Nithin Michael, Ao Kevin Tang, and Ananthram Swami. "Distributed algorithms for learning and cognitive medium access with logarithmic regret." *IEEE Journal on Selected Areas in Communications*, **29**(4):731–745, 2011.

[Ans63] FJ Anscombe. "Sequential medical trials." *Journal of the American Statistical Association*, **58**(302):365–383, 1963.

[APS11] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. "Improved algorithms for linear stochastic bandits." *Advances in neural information processing systems*, **24**, 2011.

[APS12] Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. "Online-to-confidence-set conversions and application to sparse stochastic bandits." In *Artificial Intelligence and Statistics*, pp. 1–9. PMLR, 2012.

[AVW87] Venkatachalam Anantharam, Pravin Varaiya, and Jean Walrand. "Asymptotically efficient allocation rules for the multiarmed bandit problem with multiple plays-Part I: IID rewards." *IEEE Transactions on Automatic Control*, **32**(11):968–976, 1987.

[BB20] Hamsa Bastani and Mohsen Bayati. "Online decision making with high-dimensional covariates." *Operations Research*, **68**(1):276–294, 2020.

[BBG12] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski. "A contextual-bandit algorithm for mobile context-aware recommender system." In *Neural Information Processing: 19th International Conference, ICONIP 2012, Doha, Qatar, November 12-15, 2012, Proceedings, Part III 19*, pp. 324–331. Springer, 2012.

[BCK12] Sébastien Bubeck, Nicolo Cesa-Bianchi, and Sham M Kakade. “Towards minimax policies for online linear optimization with bandit feedback.” In *Conference on Learning Theory*, pp. 41–1. JMLR Workshop and Conference Proceedings, 2012.

[Bel03] Anthony J Bell. “The co-information lattice.” In *Proceedings of the Fifth International Workshop on Independent Component Analysis and Blind Signal Separation: ICA*, volume 2003, 2003.

[BES13] Swapna Buccapatnam, Atilla Eryilmaz, and Ness B Shroff. “Multi-armed bandits in the presence of side observations in social networks.” In *52nd IEEE Conference on Decision and Control*, pp. 7309–7314. IEEE, 2013.

[BES14] Swapna Buccapatnam, Atilla Eryilmaz, and Ness B Shroff. “Stochastic bandits with side observations on networks.” In *The 2014 ACM international conference on Measurement and modeling of computer systems*, pp. 289–300, 2014.

[Bis06] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[BKS20] Ilija Bogunovic, Andreas Krause, and Jonathan Scarlett. “Corruption-tolerant gaussian process bandit optimization.” In *International Conference on Artificial Intelligence and Statistics*, pp. 1071–1081. PMLR, 2020.

[BLK21] Ilija Bogunovic, Arpan Losalka, Andreas Krause, and Jonathan Scarlett. “Stochastic linear bandits robust to adversarial attacks.” In *International Conference on Artificial Intelligence and Statistics*, pp. 991–999. PMLR, 2021.

[BLM13] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[BLS16] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. “End-to-end optimization of nonlinear transform codes for perceptual quality.” In *2016 Picture Coding Symposium (PCS)*, pp. 1–5. IEEE, 2016.

[BNO18] Alankrita Bhatt, Bobak Nazer, Or Ordentlich, and Yury Polyanskiy. “Information-distilling quantizers.” *arXiv preprint arXiv:1812.03031*, 2018.

[BZV96] Toby Berger, Zhen Zhang, and Harish Viswanathan. “The CEO problem [multiterminal source coding].” *IEEE Transactions on Information Theory*, **42**(3):887–902, 1996.

[CB15] Aaron Courville and Ian Goodfellow andYoshua Bengio. *Deep learning*. MIT press, 2015.

[CDJ21] Yifang Chen, Simon Du, and Kevin Jamieson. “Improved corruption robust algorithms for episodic reinforcement learning.” In *International Conference on Machine Learning*, pp. 1561–1570. PMLR, 2021.

[CG13] Koby Crammer and Claudio Gentile. “Multiclass classification with bandit feedback using adaptive regularization.” *Machine learning*, **90**(3):347–383, 2013.

[CGL21] William Cai, Josh Grossman, Zhiyuan Jerry Lin, Hao Sheng, Johnny Tian-Zheng Wei, Joseph Jay Williams, and Sharad Goel. “Bandit algorithms to personalize educational chatbots.” *Machine Learning*, **110**(9):2389–2418, 2021.

[cla] “Clarifai.” `http://clarifai.com/`.

[CM12] Alexandra Carpentier and Rémi Munos. “Bandit theory meets compressed sensing for high dimensional stochastic linear bandit.” In *Artificial Intelligence and Statistics*, pp. 190–198. PMLR, 2012.

[CN00] Imre Csiszár and Prakash Narayan. “Common randomness and secret key generation with a helper.” *IEEE Transactions on Information Theory*, **46**(2):344–366, 2000.

[Cov99] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[CPC10] Paul Warner Cuff, Haim H Permuter, and Thomas M Cover. “Coordination capacity.” *IEEE Transactions on Information Theory*, **56**(9):4181–4206, 2010.

[CRT06] Emmanuel J Candes, Justin K Romberg, and Terence Tao. “Stable signal recovery from incomplete and inaccurate measurements.” *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, **59**(8):1207–1223, 2006.

[Csa01] Balázs Csanád Csáji. “Approximation with artificial neural networks.” *Faculty of Sciences, Etvs Lornd University, Hungary*, **24**:48, 2001.

[CTW18] Kristy Choi, Kedar Tatwawadi, Tsachy Weissman, and Stefano Ermon. “NECST: Neural Joint Source-Channel Coding.” *arXiv preprint arXiv:1811.07557*, 2018.

[CV07] Jean-Francois Chamberland and Venugopal V Veeravalli. “Wireless sensors in distributed detection applications.” *IEEE Signal Processing Magazine*, **24**(3):16–25, 2007.

[CWF22] Yi Chen, Yining Wang, Ethan X Fang, Zhaoran Wang, and Runze Li. “Nearly dimension-independent sparse linear bandit over small action spaces via best subset selection.” *Journal of the American Statistical Association*, (just-accepted):1–31, 2022.

[DGL17] Ilias Diakonikolas, Elena Grigorescu, Jerry Li, Abhiram Natarajan, Krzysztof Onak, and Ludwig Schmidt. “Communication-efficient distributed learning of discrete distributions.” In *Advances in Neural Information Processing Systems*, pp. 6391–6401, 2017.

[DHK08] Varsha Dani, Thomas P Hayes, and Sham M Kakade. “Stochastic linear optimization under bandit feedback.” 2008.

[DKW19] Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. "Is a good representation sufficient for sample efficient reinforcement learning?" *arXiv preprint arXiv:1910.03016*, 2019.

[DLL19] Kaize Ding, Jundong Li, and Huan Liu. "Interactive anomaly detection on attributed networks." In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 357–365, 2019.

[Don06] David L Donoho. "Compressed sensing." *IEEE Transactions on information theory*, **52**(4):1289–1306, 2006.

[DP16] Rémy Degenne and Vianney Perchet. "Anytime optimal algorithms in stochastic multi-armed bandits." In *International Conference on Machine Learning*, pp. 1587–1595. PMLR, 2016.

[DSM10] Vishal Doshi, Devavrat Shah, Muriel Médard, and Michelle Effros. "Functional compression through graph coloring." *IEEE Transactions on Information Theory*, **56**(8):3901–3917, 2010.

[DWD22] Yan Dai, Ruosong Wang, and Simon S Du. "Variance-Aware Sparse Linear Bandits." *arXiv preprint arXiv:2205.13450*, 2022.

[EFD19] Yahya H Ezzeldin, Christina Fragouli, and Suhas Diggavi. "Quantizing Signals for Linear Classification." In *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 912–916. IEEE, 2019.

[EKM19] Hossein Esfandiari, Amin Karbasi, Abbas Mehrabian, and Vahab Mirrokni. "Regret bounds for batched bandits." *arXiv preprint arXiv:1910.04959*, 2019.

[EKM21] Hossein Esfandiari, Amin Karbasi, Abbas Mehrabian, and Vahab Mirrokni. "Regret bounds for batched bandits." In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7340–7348, 2021.

[EMM02] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. "PAC bounds for multi-armed bandit and Markov decision processes." In *International Conference on Computational Learning Theory*, pp. 255–270. Springer, 2002.

[EMM06] Eyal Even-Dar, Shie Mannor, Yishay Mansour, and Sridhar Mahadevan. "Action Elimination and Stopping Conditions for the Multi-Armed Bandit and Reinforcement Learning Problems." *Journal of machine learning research*, **7**(6), 2006.

[FF20] Farshad Firouzi and Bahar Farahani. "Architecting IoT Cloud." In *Intelligent Internet of Things*, pp. 173–241. Springer, 2020.

[FGM20] Dylan J Foster, Claudio Gentile, Mehryar Mohri, and Julian Zimmert. "Adapting to misspecification in contextual bandits." *Advances in Neural Information Processing Systems*, **33**:11478–11489, 2020.

[FR20] Dylan Foster and Alexander Rakhlin. "Beyond ucb: Optimal and efficient contextual bandits with regression oracles." In *International Conference on Machine Learning*, pp. 3199–3210. PMLR, 2020.

[FS97] Yoav Freund and Robert E Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting." *Journal of computer and system sciences*, **55**(1):119–139, 1997.

[GCG17] Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. "Misspecified linear bandits." In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[Ger11] Sébastien Gerchinovitz. "Sparsity regret bounds for individual sequences in online linear regression." In *Proceedings of the 24th Annual Conference on Learning Theory*, pp. 377–396. JMLR Workshop and Conference Proceedings, 2011.

[GHR19] Zijun Gao, Yanjun Han, Zhimei Ren, and Zhengqing Zhou. "Batched multi-armed bandits problem." *Advances in Neural Information Processing Systems*, **32**, 2019.

[GK73] Peter Gács and János Körner. "Common information is far less than mutual information." *Problems of Control and Information Theory*, **2**(2):149–162, 1973.

[GLS12] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[GLW19] Weihao Gao, Yu-Han Liu, Chong Wang, and Sewoong Oh:. "Rate Distortion For Model Compression: From Theory To Practice." In *ICML*, pp. 2102–2111, 2019.

[goo08] "Google Cloud AI." `http://cloud.google.com/products/ai/`, 2008. [Online; accessed 19-July-2008].

[Gro97] International Stroke Trial Collaborative Group et al. "The International Stroke Trial (IST): a randomised trial of aspirin, subcutaneous heparin, both, or neither among 19 435 patients with acute ischaemic stroke." *The Lancet*, **349**(9065):1569–1581, 1997.

[GS93] Robert M Gray and Thomas G Stockham. "Dithered quantizers." *IEEE Transactions on Information Theory*, **39**(3):805–812, 1993.

[GW74] R. M. Gray and A. D. Wyner. "Source coding for a simple network." *The Bell System Technical Journal*, **53**(9):1681–1721, 1974.

[HA98] Te Sun Han and S. Amari. "Statistical inference under multiterminal data compression." *IEEE Transactions on Information Theory*, **44**(6):2300–2324, Oct 1998.

[HEF21] Osama A Hanna, Yahya H Ezzeldin, Christina Fragouli, and Suhas Diggavi. "Quantization of distributed data for learning." *IEEE Journal on Selected Areas in Information Theory*, **2**(3):987–1001, 2021.

[HES20] Osama A Hanna, Yahya H Ezzeldin, Tara Sadjadpour, Christina Fragouli, and Suhas Diggavi. "On distributed quantization for classification." *IEEE Journal on Selected Areas in Information Theory*, **1**(1):237–249, 2020.

[HGF24] Osama Hanna, Antonious M Girgis, Christina Fragouli, and Suhas Diggavi. "Differentially private stochastic linear bandits:(almost) for free." *IEEE Journal on Selected Areas in Information Theory*, 2024.

[HK16] Elad Hazan and Zohar Karnin. "Volumetric spanners: an efficient exploration basis for learning." *Journal of Machine Learning Research*, 2016.

[HKY23] Osama A Hanna, Merve Karakas, Lin F Yang, and Christina Fragouli. "Multi-arm bandits over action erasure channels." In *2023 IEEE International Symposium on Information Theory (ISIT)*, pp. 1312–1317. IEEE, 2023.

[HKY24] Osama A Hanna, Merve Karakas, Lin Yang, and Christina Fragouli. "Multi-Agent Bandit Learning through Heterogeneous Action Erasure Channels." In *International Conference on Artificial Intelligence and Statistics*, pp. 3898–3906. PMLR, 2024.

[HLD21] Botao Hao, Tor Lattimore, and Wei Deng. "Information directed sampling for sparse linear bandits." *Advances in Neural Information Processing Systems*, **34**:16738–16750, 2021.

[HLD23] Osama A Hanna, Xinlin Li, Suhas Diggavi, and Christina Fragouli. "Common information dimension." In *2023 IEEE International Symposium on Information Theory (ISIT)*, pp. 406–411. IEEE, 2023.

[HLD24] Osama A Hanna, Xinlin Li, Suhas Diggavi, and Christina Fragouli. "On the Relation Between the Common Information Dimension and Wyner Common Information." In *2024 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2024.

[HLF22] Osama A Hanna, Xinlin Li, Christina Fragouli, and Suhas Diggavi. "Can we break the dependency in distributed detection?" In *2022 IEEE International Symposium on Information Theory (ISIT)*, pp. 2720–2725. IEEE, 2022.

[HLW20] Botao Hao, Tor Lattimore, and Mengdi Wang. "High-dimensional sparse linear bandits." *Advances in Neural Information Processing Systems*, **33**:10753–10763, 2020.

[HMP17] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework." *ICLR*, **2**(5):6, 2017.

[HOW18] Yanjun Han, Ayfer Özgür, and Tsachy Weissman. "Geometric Lower Bounds for Distributed Parameter Estimation under Communication Constraints." In *The 31st Annual Conference on Learning Theory*, 2018.

[HRZ09] Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. "Multi-class adaboost." *Statistics and its Interface*, **2**(3):349–360, 2009.

[HYF22a] Osama Hanna, Lin Yang, and Christina Fragouli. "Learning from Distributed Users in Contextual Linear Bandits Without Sharing the Context." *Advances in Neural Information Processing Systems*, **35**:11049–11062, 2022.

[HYF22b] Osama A Hanna, Lin Yang, and Christina Fragouli. "Solving multi-arm bandit using a few bits of communication." In *International Conference on Artificial Intelligence and Statistics*, pp. 11215–11236. PMLR, 2022.

[HYF22c] Osama A Hanna, Lin F Yang, and Christina Fragouli. "Compression for multi-arm bandits." *IEEE Journal on Selected Areas in Information Theory*, **3**(4):773–788, 2022.

[HYF23] Osama A Hanna, Lin Yang, and Christina Fragouli. "Contexts can be cheap: Solving stochastic contextual bandits with linear bandit algorithms." In *The Thirty Sixth Annual Conference on Learning Theory*, pp. 1791–1821. PMLR, 2023.

[HYF24] Osama Hanna, Lin Yang, and Christina Fragouli. "Efficient batched algorithm for contextual linear bandits with large action space via soft elimination." *Advances in Neural Information Processing Systems*, **36**, 2024.

[HZZ20] Yanjun Han, Zhengqing Zhou, Zhengyuan Zhou, Jose Blanchet, Peter W Glynn, and Yinyu Ye. "Sequential batch learning in finite-action linear contextual bandits." *arXiv preprint arXiv:2004.06321*, 2020.

[HZZ22] Jiafan He, Dongruo Zhou, Tong Zhang, and Quanquan Gu. "Nearly Optimal Algorithms for Linear Contextual Bandits with Adversarial Corruptions." *arXiv preprint arXiv:2205.06811*, 2022.

[IHS19] Shinji Ito, Daisuke Hatano, Hanna Sumita, Kei Takemura, Takuro Fukunaga, Naonori Kakimura, and Ken-Ichi Kawarabayashi. "Oracle-efficient algorithms for online linear optimization with bandit feedback." *Advances in Neural Information Processing Systems*, **32**, 2019.

[JBN17] Kwang-Sung Jun, Aniruddha Bhargava, Robert Nowak, and Rebecca Willett. "Scalable generalized linear bandits: Online computation and hashing." *Advances in Neural Information Processing Systems*, **30**, 2017.

[JHL21] Tiancheng Jin, Longbo Huang, and Haipeng Luo. "The best of both worlds: stochastic and adversarial episodic mdps with unknown transition." *Advances in Neural Information Processing Systems*, **34**:20491–20502, 2021.

[JKC18] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. "Quantization and training of neural networks for efficient integer-arithmetic-only inference." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2704–2713, 2018.

[JXX21] Tianyuan Jin, Pan Xu, Xiaokui Xiao, and Quanquan Gu. "Double explore-then-commit: Asymptotic optimality and beyond." In *Conference on Learning Theory*, pp. 2584–2633. PMLR, 2021.

[JZJ22] Kyoungseok Jang, Chicheng Zhang, and Kwang-Sung Jun. "PopArt: Efficient Sparse Regression and Experimental Design for Optimal Sparse Linear Bandits." *arXiv preprint arXiv:2210.15345*, 2022.

[Kar72] Richard M Karp. "Reducibility among combinatorial problems." In *Complexity of computer computations*, pp. 85–103. Springer, 1972.

[KAS23] Michael Kleinman, Alessandro Achille, Stefano Soatto, and Jonathan Kao. "Gacs-Korner Common Information Variational Autoencoder." In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[KHK96] Teuvo Kohonen, Jussi Hynninen, Jari Kangas, Jorma Laaksonen, and Kari Torkkola. "LVQ PAK: The learning vector quantization program package." Technical report, Technical report, Laboratory of computer and Information Science . . . , 1996.

[KLE14] Gowtham Ramani Kumar, Cheuk Ting Li, and Abbas El Gamal. "Exact common information." In *2014 IEEE International Symposium on Information Theory*, pp. 161–165. IEEE, 2014.

[KP09] Dinesh Krithivasan and S Sandeep Pradhan. "Lattices for distributed source coding: Jointly Gaussian sources and reconstruction of a linear function." *IEEE Transactions on Information Theory*, **55**(12):5628–5651, 2009.

[KR95] Michael N Katehakis and Herbert Robbins. "Sequential choice from several populations." *Proceedings of the National Academy of Sciences of the United States of America*, **92**(19):8584, 1995.

[KW60] Jack Kiefer and Jacob Wolfowitz. "The equivalence of two extremum problems." *Canadian Journal of Mathematics*, **12**:363–366, 1960.

[KW13] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*, 2013.

[Lai87] Tze Leung Lai. "Adaptive treatment allocation and the multi-armed bandit problem." *The Annals of Statistics*, pp. 1091–1114, 1987.

[Lan19] Peter Chal Landgren. *Distributed multi-agent multi-armed bandits*. PhD thesis, Princeton University, 2019.

[LCL10] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. "A contextual-bandit approach to personalized news article recommendation." In *Proceedings of the 19th international conference on World wide web*, pp. 661–670, 2010.

[LCS15] Tor Lattimore, Koby Crammer, and Csaba Szepesvári. “Linear multi-resource allocation with semi-bandit feedback.” *Advances in Neural Information Processing Systems*, **28**, 2015.

[LE17] Cheuk Ting Li and Abbas El Gamal. “Distributed simulation of continuous random variables.” *IEEE Transactions on Information Theory*, **63**(10):6329–6343, 2017.

[LHF23] Xinlin Li, Osama A Hanna, Christina Fragouli, Suhas Diggavi, Gunjan Verma, and Joydeep Bhattacharyya. “Feature Compression for Multimodal Multi-Object Tracking.” In *MILCOM 2023-2023 IEEE Military Communications Conference (MILCOM)*, pp. 139–143. IEEE, 2023.

[LKK18] Sergey Lobov, Nadia Krilova, Innokentiy Kastalskiy, Victor Kazantsev, and Valeri Makarov. “Latent factors limiting the performance of sEMG-interfaces.” *Sensors*, **18**(4):1122, 2018.

[LLG90] Maurizio Longo, Tom D Lookabaugh, and Robert M Gray. “Quantization for decentralized hypothesis testing under communication constraints.” *IEEE Trans. on Inf. Theory*, **36**(2):241–255, 1990.

[LLS19] Yingkai Li, Edmund Y Lou, and Liren Shan. “Stochastic linear optimization with adversarial corruption.” *arXiv preprint arXiv:1909.02109*, 2019.

[LLW21] Chung-Wei Lee, Haipeng Luo, Chen-Yu Wei, Mengxiao Zhang, and Xiaojin Zhang. “Achieving near instance-optimality and minimax-optimality in stochastic and adversarial linear bandits simultaneously.” In *International Conference on Machine Learning*, pp. 6142–6151. PMLR, 2021.

[LN06] Mikhail A Lebedev and Miguel AL Nicolelis. “Brain–machine interfaces: past, present and future.” *TRENDS in Neurosciences*, **29**(9), 2006.

[LS20] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

[LSS21] Thodoris Lykouris, Max Simchowitz, Alex Slivkins, and Wen Sun. “Corruption-robust exploration in episodic reinforcement learning.” In *Conference on Learning Theory*, pp. 3242–3245. PMLR, 2021.

[LSW20] Tor Lattimore, Csaba Szepesvari, and Gellert Weisz. “Learning with good feature representations in bandits and in rl with a generative model.” In *International Conference on Machine Learning*, pp. 5662–5670. PMLR, 2020.

[Luo05] Zhi-Quan Luo. “Universal decentralized estimation in a bandwidth constrained sensor network.” *IEEE Transactions on Information Theory*, **51**(6):2210–2219, 2005.

[LW16] Amos Lapidoth and Michèle Wigger. "Conditional and relevant common information." In *2016 IEEE International Conference on the Science of Electrical Engineering (ICSEE)*, pp. 1–5. IEEE, 2016.

[LWC21] Yingkai Li, Yining Wang, Xi Chen, and Yuan Zhou. "Tight regret bounds for infinite-armed linear contextual bandits." In *International Conference on Artificial Intelligence and Statistics*, pp. 370–378. PMLR, 2021.

[LWZ19] Yingkai Li, Yining Wang, and Yuan Zhou. "Nearly minimax-optimal regret for linearly parameterized bandits." In *Conference on Learning Theory*, pp. 2173–2174. PMLR, 2019.

[LXC10] Wei Liu, Ge Xu, and Biao Chen. "The common information of N dependent random variables." In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 836–843. IEEE, 2010.

[LYZ18] Yingming Li, Ming Yang, and Zhongfei Zhang. "A survey of multi-view representation learning." *IEEE transactions on knowledge and data engineering*, **31**(10):1863–1883, 2018.

[Mac67] James MacQueen. "Some methods for classification and analysis of multivariate observations." In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pp. 281–297. Oakland, CA, USA, 1967.

[Man17] Pasin Manurangsi. "Inapproximability of maximum edge biclique, maximum balanced biclique and minimum k-cut from the small set expansion hypothesis." In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[Mau93] Ueli M Maurer. "Secret key agreement by public discussion from common information." *IEEE transactions on information theory*, **39**(3):733–742, 1993.

[Mey00] Carl D Meyer. *Matrix analysis and applied linear algebra*, volume 71. Siam, 2000.

[MGP15] Jérémie Mary, Romaric Gaudel, and Philippe Preux. "Bandits and recommender systems." In *International Workshop on Machine Learning, Optimization and Big Data*, pp. 325–336. Springer, 2015.

[MGV11] Vinith Misra, Vivek K Goyal, and Lav R Varshney. "Distributed scalar quantization for computing: High-resolution analysis and extensions." *IEEE Transactions on Information Theory*, **57**(8):5298–5325, 2011.

[MRS19] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. "Disentangling Disentanglement in Variational Autoencoders." In *International Conference on Machine Learning*, pp. 4402–4412, 2019.

[MT04] Shie Mannor and John N Tsitsiklis. "The sample complexity of exploration in the multi-armed bandit problem." *Journal of Machine Learning Research*, **5**(Jun):623–648, 2004.

[MT20] Prathamesh Mayekar and Himanshu Tyagi. "RATQ: A universal fixed-length quantizer for stochastic optimization." In *International Conference on Artificial Intelligence and Statistics*, pp. 1399–1409. PMLR, 2020.

[Mun14] James Munkres. *Topology*. Pearson Education, 2014.

[Ooh98] Yasutada Oohama. "The rate-distortion function for the quadratic Gaussian CEO problem." *IEEE Transactions on Information Theory*, **44**(3):1057–1070, 1998.

[OR11] Luís ML Oliveira and Joel JPC Rodrigues. "Wireless Sensor Networks: A Survey on Environmental Monitoring." *JCM*, **6**(2):143–151, 2011.

[OVK17] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. "Neural discrete representation learning." In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

[PAS18] Ciara Pike-Burke, Shipra Agrawal, Csaba Szepesvari, and Steffen Grunewalder. "Bandits with delayed, aggregated anonymous feedback." In *International Conference on Machine Learning*, pp. 4105–4113. PMLR, 2018.

[Poo88] H V. Poor. "High-rate vector quantization for detection." *IEEE Transactions on Information Theory*, **34**:960–972, 1988.

[PPA20] Aldo Pacchiano, My Phan, Yasin Abbasi Yadkori, Anup Rao, Julian Zimmert, Tor Lattimore, and Csaba Szepesvari. "Model selection in contextual stochastic bandit problems." *Advances in Neural Information Processing Systems*, **33**:10328–10337, 2020.

[PRC16] Vianney Perchet, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg. "Batched bandit problems." *The Annals of Statistics*, pp. 660–681, 2016.

[PTR04] Vinod Prabhakaran, David Tse, and Kannan Ramachandran. "Rate region of the quadratic Gaussian CEO problem." In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, p. 119. IEEE, 2004.

[RDN22] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. "Hierarchical text-conditional image generation with clip latents." *arXiv preprint arXiv:2204.06125*, 2022.

[Ren59] Alfréd Rényi. "On the dimension and entropy of probability distributions." *Acta Mathematica Academiae Scientiarum Hungarica*, **10**(1):193–215, 1959.

[RMR96] Ajit Rao, David Miller, Kenneth Rose, and Allen Gersho. "A generalized VQ method for combined compression and estimation." In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 4, pp. 2032–2035. IEEE, 1996.

[Rob52] Herbert Robbins. "Some aspects of the sequential design of experiments." *Bulletin of the American Mathematical Society*, **58**(5):527–535, 1952.

[ROV19] Ali Razavi, Aäron van den Oord, and Oriol Vinyals. "Generating Diverse High-Fidelity Images with VQ-VAE-2." *arXiv preprint arXiv:1906.00446*, 2019.

[RV14] Daniel Russo and Benjamin Van Roy. "Learning to optimize via posterior sampling." *Mathematics of Operations Research*, **39**(4):1221–1243, 2014.

[RYZ21] Yufei Ruan, Jiaqi Yang, and Yuan Zhou. "Linear bandits with limited adaptivity and learning distributional optimal design." In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 74–87, 2021.

[Sah74] Sartaj Sahni. "Computationally related problems." *SIAM Journal on computing*, **3**(4):262–279, 1974.

[SC15] Sanket Satpathy and Paul Cuff. "Gaussian secure source coding and wyner's common information." In *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 116–120. IEEE, 2015.

[SDV21] Thomas M. Sutter, Imant Daunhawer, and Julia E Vogt. "Generalized Multimodal ELBO." In *International Conference on Learning Representations*, 2021.

[SEF14] Marco Di Summa, Friedrich Eisenbrand, Yuri Faenza, and Carsten Moldenhauer. "On largest volume simplices and sub-determinants." In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pp. 315–323. SIAM, 2014.

[SFD14] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns." In *Fifteenth Annual Conference of the International Speech Communication Association*. Citeseer, 2014.

[SFS18] Linqi Song, Christina Fragouli, and Devavrat Shah. "Recommender systems over wireless: Challenges and opportunities." In *2018 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2018.

[SG22] Erixhen Sula and Michael Gastpar. "The Gray-Wyner Network and Wyner's Common Information for Gaussian Sources." *IEEE Transactions on Information Theory*, **68**(2):1369–1384, 2022.

[SM50] Jack Sherman and Winifred J Morrison. “Adjustment of an inverse matrix corresponding to a change in one element of a given matrix.” *The Annals of Mathematical Statistics*, **21**(1):124–127, 1950.

[SMG13] John Z Sun, Vinith Misra, and Vivek K Goyal. “Distributed functional scalar quantization simplified.” *IEEE Transactions on Signal Processing*, **61**(14):3495–3508, 2013.

[SRJ17] Shahin Shahrampour, Alexander Rakhlin, and Ali Jadbabaie. “Multi-armed bandits in multi-agent networks.” In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2786–2790. IEEE, 2017.

[STS19] Matan Shohat, Georgee Tsintsadze, Nir Shlezinger, and Yonina C Eldar. “Deep Quantization for MIMO Channel Estimation.” In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3912–3916. IEEE, 2019.

[SY96] Atsushi Sato and Keiji Yamada. “Generalized learning vector quantization.” In *Advances in neural information processing systems*, pp. 423–429, 1996.

[SZ15] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition.” In *ICLR*, 2015.

[TDW14] Zhuowen Tu, Piotr Dollar, and Yingnian Wu. “Layered Logic Classifiers: Exploring theAnd’andOr’Relations.” *arXiv preprint arXiv:1405.6804*, 2014.

[Tho33] William R Thompson. “On the likelihood that one unknown probability exceeds another in view of the evidence of two samples.” *Biometrika*, **25**(3/4):285–294, 1933.

[TM17] Ambuj Tewari and Susan A Murphy. “From ads to interventions: Contextual bandits in mobile health.” *Mobile Health: Sensors, Analytic Methods, and Applications*, pp. 495–517, 2017.

[Tod16] Michael J Todd. *Minimum-volume ellipsoids: Theory and algorithms*. SIAM, 2016.

[TPB00] Naftali Tishby, Fernando C Pereira, and William Bialek. “The information bottleneck method.” *arXiv preprint physics/0004057*, 2000.

[TSC17] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. “Lossy image compression with compressive autoencoders.” *arXiv preprint arXiv:1703.00395*, 2017.

[TVV04] Kalyan T Talluri, Garrett Van Ryzin, and Garrett Van Ryzin. *The theory and practice of revenue management*, volume 1. Springer, 2004.

[VAR14] Kumar B Viswanatha, Emrah Akyol, and Kenneth Rose. “The lossy common information of correlated sources.” *IEEE Transactions on Information Theory*, **60**(6):3238–3253, 2014.

[VB97] Harish Viswanathan and Toby Berger. "The quadratic Gaussian CEO problem." *IEEE Transactions on Information Theory*, **43**(5):1549–1559, 1997.

[VGO20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." *Nature Methods*, **17**:261–272, 2020.

[VKK16] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel Recurrent Neural Networks." In *International Conference on Machine Learning*, pp. 1747–1756, 2016.

[VMK14] Michal Valko, Rémi Munos, Branislav Kveton, and Tomáš Kocák. "Spectral bandits for smooth graph functions." In *International Conference on Machine Learning*, pp. 46–54. PMLR, 2014.

[Von37] John Von Neumann. *Some matrix-inequalities and metrization of matric space*. 1937.

[VSS20] Daniel Vial, Sanjay Shakkottai, and R Srikant. "One-bit feedback is sufficient for upper confidence bound policies." *arXiv preprint arXiv:2012.02876*, 2020.

[Wag11] Aaron B Wagner. "On distributed compression of linear functions." *IEEE Transactions on Information Theory*, **57**(1):79–94, 2011.

[Wai19] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

[Wat60] Satosi Watanabe. "Information theoretical analysis of multivariate correlation." *IBM Journal of research and development*, **4**(1):66–82, 1960.

[WDZ22] Chen-Yu Wei, Christoph Dann, and Julian Zimmert. "A model selection approach for corruption robust reinforcement learning." In *International Conference on Algorithmic Learning Theory*, pp. 1043–1096. PMLR, 2022.

[Wit75] Hans S Witsenhausen. "On sequences of pairs of dependent random variables." *SIAM Journal on Applied Mathematics*, **28**(1):100–113, 1975.

[Wit76] Hans S Witsenhausen. "Values and bounds for the common information of two discrete random variables." *SIAM Journal on Applied Mathematics*, **31**(2):313–333, 1976.

[WLC18] Shuang Wu, Guoqi Li, Feng Chen, and Luping Shi. "Training and Inference with Integers in Deep Neural Networks." In *International Conference on Learning Representations*, 2018.

[WNW13] Robert Wood, Radhika Nagpal, and Gu-Yeon Wei. “Flight of the robobees.” *Scientific American*, **308**(3):60–65, 2013.

[WV10] Yihong Wu and Sergio Verdú. “Rényi information dimension: Fundamental limits of almost lossless analog compression.” *IEEE Transactions on Information Theory*, **56**(8):3721–3748, 2010.

[Wyn75] Aaron Wyner. “The common information of two dependent random variables.” *IEEE Transactions on Information Theory*, **21**(2):163–179, 1975.

[XLC15] Ge Xu, Wei Liu, and Biao Chen. “A lossy source coding interpretation of Wyner’s common information.” *IEEE Transactions on Information Theory*, **62**(2):754–768, 2015.

[YLC17] Lei Yu, Houqiang Li, and Chang Wen Chen. “Generalized Common Informations: Measuring Commonness by the Conditional Maximal Correlation.”, 2017.

[YRS22] Shuo Yang, Tongzheng Ren, Sanjay Shakkottai, Eric Price, Inderjit S Dhillon, and Sujay Sanghavi. “Linear bandit algorithms with sublinear time complexity.” In *International Conference on Machine Learning*, pp. 25241–25260. PMLR, 2022.

[YT20] Lei Yu and Vincent YF Tan. “On exact and ∞-Rényi common informations.” *IEEE Transactions on Information Theory*, **66**(6):3366–3406, 2020.

[YT22] Lei Yu, Vincent YF Tan, et al. “Common Information, Noise Stability, and Their Extensions.” *Foundations and Trends® in Communications and Information Theory*, **19**(2):107–389, 2022.

[ZFL22] Yinglun Zhu, Dylan J Foster, John Langford, and Paul Mineiro. “Contextual bandits with large action spaces: Made practical.” In *International Conference on Machine Learning*, pp. 27428–27453. PMLR, 2022.

[ZJZ21] Zihan Zhang, Xiangyang Ji, and Yuan Zhou. “Almost Optimal Batch-Regret Tradeoff for Batch Linear Contextual Bandits.” *arXiv preprint arXiv:2110.08057*, 2021.

[ZLK20] Andrea Zanette, Alessandro Lazaric, Mykel Kochenderfer, and Emma Brunskill. “Learning near optimal policies with low inherent bellman error.” In *International Conference on Machine Learning*, pp. 10978–10989. PMLR, 2020.

[ZSE19] Shengjia Zhao, Jiaming Song, and Stefano Ermon. “Infovae: Balancing learning and inference in variational autoencoders.” In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5885–5892, 2019.

[Zuc06] David Zuckerman. “Linear degree extractors and the inapproximability of max clique and chromatic number.” In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pp. 681–690. ACM, 2006.

[ZYG17] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. “Incremental Network Quantization: Towards Lossless CNNs with Low-precision Weights.” In *International Conference on Learning Representations*, 2017.