

Lower Bounds on Average-Case and Non-Local Quantum Computation

by

Vahid Reza Asadi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science (Quantum Information)

Waterloo, Ontario, Canada, 2026

© Vahid Reza Asadi 2026

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Henry Yuen
Associate Professor, Computer Science Department,
Columbia University

Supervisors: Richard Cleve
Professor, Cheriton School of Computer Science,
University of Waterloo

Mohammad Hajiabadi
Assistant Professor, Cheriton School of Computer Science,
University of Waterloo

Internal Members: Eric Blais
Associate Professor, Cheriton School of Computer Science,
University of Waterloo

Rafael Oliveira
Assistant Professor, Cheriton School of Computer Science,
University of Waterloo

Internal-External Member: Ashwin Nayak
Professor, Dept. of Combinatorics and Optimization,
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The results presented in this thesis have appeared in the following published works, which I co-authored during my Ph.D. studies.

- Part I of the thesis is based on [AGGS22, AGG⁺24]. In particular, Chapter 4 presents results obtained in [AGGS22], and Chapter 5 presents results from [AGG⁺24].
- Part II is based on [ACM25, ACCM25]. Results from [ACM25] are presented in Chapter 9, and results from [ACCM25] appear in Chapter 10.

Abstract

This thesis studies computational and information-theoretic limitations in both classical and quantum models of computation. It is organized into two parts, each addressing a different aspect of computational hardness. Despite their differences, both parts share a common goal: understanding how structural and physical constraints shape what classical and quantum algorithms can achieve.

In Part I, we present lower bounds on the average-case complexity of certain tasks in classical and quantum settings. We develop a general framework for constructing efficient worst-case to average-case reductions. Applying this framework, we obtain such reductions for fundamental problems in a variety of computational models; namely, algorithms for matrix multiplication, streaming algorithms for the online matrix-vector multiplication problem, and static data structures for all linear problems, as well as the multivariate polynomial evaluation problem. We further extend this framework to the setting of quantum algorithms, along the way obtaining a tight bound on the average-case quantum query complexity of the matrix-vector multiplication problem. Our techniques rely crucially on tools from additive combinatorics. In particular, we show local correction lemmas that rely on new probabilistic and noise-robust versions of the quasi-polynomial Bogolyubov-Ruzsa lemma.

In Part II, we give quantum gate and entanglement lower bounds on certain non-local tasks. A non-local quantum computation (NLQC) replaces direct interaction between two quantum systems with a single simultaneous round of communication and shared entanglement. We study two classes of NLQC, f -routing and f -BB84, which are of relevance to classical information theoretic cryptography and quantum position-verification. We give the first non-trivial lower bounds on entanglement in both settings, under the assumption of perfect correctness. Within this setting, we give a lower bound on the Schmidt rank of any entangled state that completes these tasks for a given function $f(x, y)$ in terms of the rank of a matrix G_{xy} whose entries are zero when $f(x, y) = 0$ and strictly positive otherwise. This also leads to a lower bound in terms of the non-deterministic quantum communication complexity of f . Due to a relationship between f -routing and the conditional disclosure of secrets (CDS) primitive studied in information theoretic cryptography, we obtain a new technique for lower bounding the randomness complexity of CDS. Finally, we show that the number of quantum gates plus single-qubit measurements required to implement a function f is at least linear in the entanglement-assisted simultaneous-message-passing communication complexity of f . As a consequence, we derive a linear lower bound for the inner-product function.

Acknowledgements

First and foremost, I would like to thank my advisors, Professors Richard Cleve and Mohammad Hajiabadi. This journey would not have been possible without your unwavering support throughout my studies, and I am deeply indebted to you for your guidance. I have learned so much from both of you and will always be grateful to have you as my mentors.

Next, I would like to thank the members of my committee, Professors Eric Blais, Ashwin Nayak, Rafael Oliveira, and Henry Yuen, for taking the time to read this thesis and provide valuable feedback.

This thesis is based on results obtained in collaboration with many amazing researchers, including Eric Culf, Sasha Golovnev, Tom Gur, Alex May, Igor Shinkar, and Sathya Subramanian. These collaborations were invaluable in shaping the researcher I am today, and I am thankful to you all.

During my studies, I had two wonderful opportunities to visit other institutions. I thank Professor Alessandro Chiesa for hosting me at EPFL during the summer of 2023. I am also grateful to Giacomo Fenzi and Ziyi Guan for kindly sharing their office with me during that time and for their continued friendship. I thank Professor François Le Gall for hosting me at Nagoya University in the summer of 2025, and Atsuya Hasegawa for many stimulating and insightful discussions.

During my time at Waterloo, I had the pleasure of working alongside amazing faculty members in the Algorithms & Complexity Group and at the Institute for Quantum Computing. In particular, I would like to thank Professors Sepehr Assadi, Eric Blais, Lap Chi Lau, and Rafael Oliveira for their support and friendship, as well as Professors Elena Grigorescu and Luke Schaeffer, from whom I greatly valued learning, despite our brief interactions.

I would also like to thank many students with whom I had the pleasure of spending time, including Enver Aman, Robert Andrews, Arkaprava Choudhury, Mohammad Hossein Ebtehaj, Remy El Sabeh, Renato Ferreira Pinto Jr., Abhibhav Garg, Parth Gupta, Nathan Harms, Lars Kamin, Danny Kong, Erica Liu, Raymond Liu, Parth Mittal, Argyris Mouzakis, Rian Neogi, Matthew Regehr, Cameron Seth, Vihan Shah, Janani Sundaresan, Dante Tjowasi, Dinis Vitorino, Christopher Xu, and Helia Yazdanyar, as well as many others whom I have likely omitted here because of my goldfish memory! Beyond Waterloo but still within the realm of academia, I am grateful for the friendship of Naren Manoj and Rachel Yun Zhang, whom I met at my first in-person conference and who have been wonderful friends ever since.

I would like to especially thank two of my dear friends, Ali Akhavani and Hossein Ayati. Despite only being able to visit them in person once since I left home for my doctoral studies, they have been truly amazing friends and have listened to my many ramblings and concerns.

Lastly and most importantly, I would like to thank my parents and my brother, Amir Reza, for their constant support throughout this journey. Without your love and encouragement, I would not have lasted more than a few months. I am endlessly grateful to have you by my side.

*There are a hundred thousand baits and traps,
O God, and we're like greedy, helpless birds!
Though we become like falcons or Simorghs,
each single moment we are newly snared.
You free us every moment, and again
we get ensnared, O Self-Sufficient One!*

— Jalāl al-Dīn Rūmī, *The Masnavi*, trans. Alan Williams

Dedication

To my parents.

Table of Contents

Examining Committee Membership	ii
Author’s Declaration	iii
Statement of Contributions	iv
Abstract	v
Acknowledgements	vi
Dedication	viii
I Average-Case Lower Bounds for Classical and Quantum Computation	1
1 Introduction	2
1.1 Classical contributions	4
1.1.1 Algorithms for matrix multiplication	5
1.1.2 Data structures for all linear problems	6
1.1.3 Online matrix-vector multiplication	8
1.1.4 Worst-case to weak-average-case reductions	9
1.2 Quantum contributions	11

2	Preliminaries	15
2.1	Classical reductions	15
2.2	The challenge: low-agreement regime	15
2.3	Local correction via additive combinatorics	17
2.4	Illustrating example: matrix multiplication	20
2.5	Beyond matrix multiplication	23
2.6	Quantum reductions	25
2.7	Quantum local correction via a robust Bogolyubov lemma	27
2.8	A toolkit of quantum algorithms for local correction	30
2.9	Quantum worst-case to average-case reductions	34
3	Additive Combinatorics Toolbox	37
3.1	Probabilistic and quasi-polynomial Bogolyubov-Ruzsa lemmas	38
3.2	Algorithmic construction of the subspace V	46
3.3	Local correction lemma	46
3.4	Robust probabilistic Bogolyubov lemma	48
3.5	Quantum local correction lemma	50
4	Classical Reductions	53
4.1	Worst-case to average-case reductions for matrix multiplication	53
4.1.1	Reduction for matrices over small fields	54
4.1.2	Reduction for matrices over large fields	59
4.2	Worst-case to average-case reductions for online matrix-vector multiplication	61
4.3	Worst-case to average-case reductions for data structures	68
4.3.1	Average-case reductions for all linear problems	68
4.3.2	Weak-average-case data structures	71
4.3.3	Impossibility of weak-average-case reductions for all linear problems	73
4.3.4	Weak-average-case reductions for multivariate polynomial evaluation	74

5	Quantum Reductions	83
5.1	Basics of quantum computation	83
5.1.1	Quantum unitary oracles and Fourier transforms	83
5.1.2	Average-case quantum algorithms	85
5.1.3	Quantum singular value transformation techniques	86
5.2	Quantum worst-case to average-case reductions	88
5.2.1	Quantum algorithms for all linear problems	89
5.2.2	Matrix-vector multiplication in the query model	90
5.3	Quantum toolkit for local correction	91
5.3.1	Flagging correct matrix-vector products in superposition	92
5.3.2	Noisy quantum oracles approximating the indicator function 1_X	99
5.3.3	Quantum sampling from the set of good inputs	101
5.3.4	Learning Bogolyubov subspaces from noisy quantum oracles	102
5.4	Reductions for linear problems	107
5.4.1	Properties of threshold sets	107
5.4.2	Proof of Lemma 5.2.1	109
6	Conclusion	113
6.1	Subsequent work	113
6.2	Open problems	114
II	Lower Bounds on Non-Local Quantum Computation	117
7	Introduction	118
7.1	Contributions	122
7.1.1	Entanglement lower bounds	122
7.1.2	Gate lower bounds	125

8 Preliminaries	128
8.1 Notation and definitions	128
8.2 Communication complexity	130
8.3 f -BB84 and f -routing	133
8.3.1 f -BB84 definition and the strategy model	134
8.3.2 f -routing definition and the strategy model	136
9 Entanglement Lower Bounds	138
9.1 Lower bounds for f -routing	138
9.2 Lower bounds for f -BB84	143
9.3 A new lower bound technique for CDS	147
9.4 Application to concrete functions	150
10 Gate Lower Bounds	152
10.1 Analysis of f -BB84	152
10.2 Analysis of f -routing	155
10.3 Reduction to SMP^* and lower bounds	157
10.3.1 Reduction to SMP^*	157
10.3.2 Explicit lower bounds for f -BB84 and f -routing	159
10.3.3 Nearly matching upper bound for inner product	161
11 Conclusion	163
11.1 Summary of results	163
11.2 Open problems	165
References	167

Part I

Average-Case Lower Bounds for Classical and Quantum Computation

Chapter 1

Introduction

Average-case algorithms are only required to succeed with high probability on a “typical” input, as opposed to *every* input (cf. the standard textbooks [Gol08, AB09]). Worst-case to average-case reductions (also known as random self-reductions) provide a method for transforming such algorithms, that can only solve a problem for a fraction of the inputs, into algorithms that can solve the problem *for all* inputs.

For instance, consider one of the most fundamental algorithmic problems: matrix multiplication. Suppose we have an average-case algorithm **ALG** that can correctly compute the product $A \cdot B$ on an α -fraction of matrices $A, B \in \mathbb{F}^{n \times n}$; that is, $\Pr[\mathbf{ALG}(A, B) = A \cdot B] \geq \alpha$. Is it possible to use **ALG** to obtain an algorithm that computes $A \cdot B$ *for all* input matrices? A worst-case to average-case reduction will give a positive answer to this question, boosting the *success rate* α to 1, without incurring significant overhead. Of course, the same question can be asked with respect to any other computational problem.

In this thesis, we study such reductions for average-case algorithms where the success rate α could be very small, such as in the 1% regime and even when α slowly tends to zero (i.e., for algorithms that are only correct on a vanishing fraction of their inputs)¹, in both classical and quantum models of computation. There are two natural perspectives in which we can view such reductions. On the one hand, they can provide a proof that a problem retains its hardness even in the average case. On the other hand, they provide a paradigm for designing worst-case algorithms, by first constructing algorithms that are

¹We stress that the 1% regime is far more challenging than, say, the 99% regime. Indeed, in the 99% regime, the classical self-correction of Blum, Luby, and Rubinfeld [BLR90] already provides fine-grained worst-case to average-case reductions for a number of problems.

only required to succeed on a small fraction of their inputs, and then using the reduction to obtain algorithms that are correct on all inputs.

Background and context. The study of the average-case complexity originates in the work of Levin [Lev86], and followup works such as [BCGL92]. A long line of works established various barriers to designing worst-case to average-case reductions for NP-complete problems (see, e.g., [IL90, Imp11] and references therein). We refer the reader to the classical surveys by Impagliazzo [Imp95], Bogdanov and Trevisan [BT06], and Goldreich [Gol11] on this topic.

On the positive side, Lipton [Lip91] proved that the matrix permanent problem admits a polynomial-time worst-case to average-case reduction. Ajtai [Ajt96] designed worst-case to average-case reductions for certain lattice problems, which led to constructions of efficient cryptographic primitives from worst-case assumptions [AD97, Reg04]. Many number-theoretic problems in cryptography have long been known to admit such reductions (for fixed parameters) due to random self-reducibility: for example, the discrete logarithm problem, the RSA problem, and the quadratic residuosity problem (see, e.g., [Sho09]). For the matrix multiplication problem, there is a weak reduction that requires the average-case algorithm to succeed with a very high probability, greater than $3/4$ (see Section 2.2). There are also known worst-case to average-case reductions for many problems that are not thought to be in NP [FF93, BFNW93, STV01].

Efficient worst-case to average-case reductions for problems in P have also received a lot of attention in recent years [BRSV17, GR18, DLV20, BABB21, HLS22, NRW22, HS23, GSS24, HS25a, HS25b, SS25]. The study of the fine-grained complexity [Vas18] of algorithmic problems sparked interest in designing *efficient* worst-case to average-case reductions, i.e., reductions that do not suffer a polynomial overhead in the running time. Such reductions are motivated by fine-grained cryptographic applications. A large body of work is devoted to establishing fine-grained worst-case to average-case reductions for orthogonal vectors, 3SUM, k -clique, and various algebraic problems, as well as building certain cryptographic primitives from them [BRSV17, BRSV18, GR18, LLV19, BBB19, DLV20]. Since there are no known constructions of one-way functions and public-key cryptography from well-established fine-grained assumptions, the question of constructing efficient worst-case to average-case reductions for other fine-grained problems still attracts much attention.

Prior to our work, such reductions were only known for a small number of specific problems or restricted computational models. It remained open whether it is possible to construct efficient worst-case to average-case reductions, in a general model of computa-

tion, for expressive classes of functions such as the class of all linear problems. Hirahara and Shimizu [HS23] recently provided a general paradigm that simplified, optimised, and unified known worst-case to average-case reductions for key problems of interest, including matrix multiplication. Unfortunately, their approach inherently cannot capture the class of linear problems, as it does not admit the direct-product structure that is required for their techniques to work. Given these barriers in previous approaches, it is unclear whether such reductions exist at all.

In the quantum setting, worst-case to average case reductions were previously studied in the context of (non-fine-grained) cryptography. These reductions are for a handful specific problems and in the 99% regime, as opposed to the harder 1% regime that is the focus of our work. The celebrated work of Shor [Sho94] gave a polynomial time quantum algorithm breaking number-theoretic cryptosystems and sparking interest in post-quantum cryptography, i.e., cryptography secure even against (polynomial time) quantum adversaries. A number of quantum and classical worst-case to average-case reductions [Ajt96, MR04, Reg09, LPR13, LS15, Gen10] allowed us to base the security of (post-quantum) lattice-based cryptography on the *worst-case* quantum hardness assumptions for certain computational problems. Another interesting example of a (quantum) worst-case to average-case reduction was recently given in [LdW22] for problems related to phase estimation.

Our contributions. We design a framework for local correction based on the Bogolyubov-Ruzsa lemma for showing explicit worst-case to average-case reductions, and we use it to obtain reductions for fundamental problems in a variety of computational models. Informally, we show that if a problem has an algorithm that runs in time T and succeeds on α -fraction of its inputs (even for sub-constant success rate α), then there exists a worst-case algorithm for this problem, which runs in time $\tilde{O}(T)$.

We describe each of our results in detail below.

1.1 Classical contributions

We first present our main contributions in the classical setting. We design worst-case to average-case reductions for the matrix multiplication problem in the setting of algorithms, for the online matrix-vector multiplication problem in the streaming setting, for *all* linear problems in the setting of static data structures, and worst-case to weak-average-case

reductions for the problem of multivariate polynomial evaluation in the data structures setting.

1.1.1 Algorithms for matrix multiplication

Recall that in the matrix multiplication problem, the goal is simply to compute the product of two given matrices $A, B \in \mathbb{F}^{n \times n}$. A long line of research, culminating in the work [ADV⁺25], led to matrix multiplication algorithms performing $O(n^{2.372})$ operations. We present a worst-case to average-case reduction for the matrix multiplication problem over prime fields. Namely, we show that if there exists a (randomized) algorithm that, given two matrices $A, B \in \mathbb{F}^{n \times n}$, runs in time $T(n)$ and correctly computes their product for a small fraction of all possible inputs, then there exists a (randomized) algorithm that runs in $\tilde{O}(T(n))$ time and outputs the correct answer for all inputs. Formally, we have the following theorem.

Theorem 1. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Suppose that there exists an algorithm **ALG** that, on input two matrices $A, B \in \mathbb{F}^{n \times n}$ runs in time $T(n)$ and satisfies*

$$\Pr[\mathbf{ALG}(A, B) = A \cdot B] \geq \alpha,$$

*where the probability is taken over the random inputs $A, B \in \mathbb{F}^{n \times n}$ and the randomness of **ALG**.*

- *If $|\mathbb{F}| \leq 2/\alpha$, then there exists a randomized algorithm **ALG'** that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $\frac{\exp(O(\log^5(1/\alpha)))}{\delta} \cdot T(n)$ and outputs AB with probability at least $1 - \delta$.*
- *If $|\mathbb{F}| \geq 2/\alpha$, then there exists a randomized algorithm **ALG'** that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $O(\frac{1}{\delta \cdot \alpha^4} \cdot T(n))$ and outputs AB with probability at least $1 - \delta$.*

For example, if we have an algorithm that succeeds on α fraction of the inputs for $\alpha > \exp\left(-\sqrt[6]{\log(n)}\right)$ in time $T(n) = n^c$, then we get an algorithm that works for all inputs and runs in time $n^{c+o(1)}$. In particular, if we have an $n^{2+o(1)}$ algorithm that succeeds on $\alpha > \exp\left(-\sqrt[6]{\log(n)}\right)$ fraction of the inputs, then there is a worst case algorithm with running time $n^{2+o(1)}$.

1.1.2 Data structures for all linear problems

The class of linear problems plays a central role in computer science and mathematics, yielding a myriad of applications both in theory and practice. Below we describe worst-case to average-case reductions for static data structures for *all linear problems*. Recall that a linear problem L_A over a field $\mathbb{F}$ is defined by a matrix $A \in \mathbb{F}^{m \times n}$.² An input to the problem is a vector $v \in \mathbb{F}^n$, which is preprocessed into s memory cells. Then, given a query $i \in [m]$, the goal is to output $\langle A_i, v \rangle$, where A_i is the i th row of A , by probing at most t of the memory cells, where t is called the query time.

Note that the trivial solutions for data structure problems are to either: (i) store only $s = n$ memory cells containing the input v , and for each query $i \in [m]$, read v entirely and compute the answer in query time $t = n$; or (ii) use $s = m$ memory cells, where the i th cell contains the answer to the query $i \in [m]$, thus allowing for query time $t = 1$. In a typical application, the number of queries $m = \text{poly}(n) \gg n$, and a data structure is efficient if it uses space $s = \tilde{O}(n)$ (or $s \ll m$) and has query time $t = \text{poly}(\log(n))$ (or $t = n^\varepsilon$ for a small constant $\varepsilon > 0$). Note that the two trivial solutions do not lead to such efficient data structures for $m \gg n$.

We consider randomized data structures, where both the preprocessing stage and the query stage use randomness, and are expected to output the correct answer with high probability (over the randomness of both stages). In average-case randomized data structures, the success rate of the algorithm is taken over both the inner randomness and the random input, whereas in worst-case randomized data structures, the success rate is taken only over the inner randomness of the algorithm (i.e., the algorithm succeeds with high probability *on all* inputs).

We present a worst-case to average-case reduction showing that if there exists a data structure DS that uses s memory cells, has query time t , and success rate such that for a small fraction of inputs the data structure answers all queries correctly, then there exists another data structure DS' that uses $4s$ memory cells, has query time $4t$, and success rate such that *for all inputs* the data structure answers all queries correctly with high probability.

Theorem 2. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $\alpha := \alpha(n) \in (0, 1]$, $n, m \in \mathbb{N}$, and a matrix $A \in \mathbb{F}^{m \times n}$. Denote by L_A the linear problem of outputting $\langle A_i, x \rangle$ on input $x \in \mathbb{F}^n$ and*

²Formally, L_A is defined by an infinite sequence of matrices $(A_n)_{n \geq 1}$, where $A_n \in \mathbb{F}^{m \times n}$ for $m = m(n)$.

query $i \in [m]$. Suppose that

$$L_A \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{x \in \mathbb{F}^n} [\text{DS}_x(i) = \langle A_i, x \rangle \forall i \in [m]] \geq \alpha \end{array} \right].$$

Then for every $\delta > 0$,

$$L_A \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha) \log(n)) \\ \text{query time:} & 4t + O(\log^4(1/\alpha) \log(n)) \\ \text{success rate:} & \forall x \in \mathbb{F}^n \quad \Pr[\text{DS}'_x(i) = \langle A_i, x \rangle \forall i \in [m]] \geq 1 - \delta \end{array} \right].$$

We stress that in the average-case data structure we start with, the probability is taken over a random input (as well as the inner randomness of the algorithm), whereas in the worst-case data structure that we obtain, with high probability the algorithm is successful *on all* inputs.

The reduction above shows that for any linear problem L_A , if a data structure succeeds on an arbitrary small constant $\alpha > 0$ fraction of the inputs, then we can obtain a data structure that succeeds on all inputs with parameters that essentially differ only by a constant multiplicative factor, and the query complexity t translates into query complexity $4t + O(\log(n))$.

We note that the $O(\log^4(1/\alpha) \log(n))$ overhead in the space complexity of the constructed data structure is caused by storing $O(\log^4(1/\alpha))$ numbers from $[n]$. In particular, if the word size of the data structure is $w \geq \log(n)$, then the space complexity of the resulting data structure is $4s + O(\log^4(1/\alpha))$. Similarly, in this case the query complexity of the resulting data structure is $4t + O(\log^4(1/\alpha))$.

Note that for any non-trivial data structure problem, a data structure must use at least $\Omega(n)$ memory cells (only to store a representation of the input). Therefore, even for α as small as $\alpha = 2^{-n^\eta}$ for a small constant $\eta > 0$, the overhead in the space complexity is negligible. For typical query times of data structures, such as $t = \text{poly}(\log(n))$ and $t = n^\epsilon$, the overhead in the query time is negligible even for $\alpha = 1/\text{poly}(n)$ and $\alpha = 2^{-n^\eta}$, respectively.

1.1.3 Online matrix-vector multiplication

Next we turn to the core data structure problem in fine-grained complexity, the online matrix-vector multiplication problem (OMV). In the data structure variant of this streaming problem, one needs to preprocess a matrix $M \in \mathbb{F}^{n \times n}$, such that given a query vector $v \in \mathbb{F}^n$, one can quickly compute Mv . The study of OMV (over the Boolean semiring) and its applications to fine-grained complexity originates from [HKNS15], and [LW17, CKL18] give surprising upper bounds for the problem. Over finite fields, [FHM01, CGL15] give lower bounds for OMV, and [CKLM18] proves lower bounds for a related vector-matrix-vector multiplication problem. We prove an efficient worst-case to average-case reduction for OMV over prime fields. A concurrent and independent work [HLS21] studies worst-case to average-case reductions for OMV over the Boolean semiring and their applications.

Note that in OMV, the average case condition only guarantees success with probability taken over *both* the matrix M as well as the vector v . Nevertheless, we can exploit the fact that each coordinate of the correct output is a linear function in the entries of M , and extend our techniques to the more involved setting of OMV.

Theorem 3. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Consider the matrix-vector multiplication problem $\text{OMV}_{\mathbb{F}}$ for dimension n , and suppose that for some $\alpha > 0$ it holds that*

$$\text{OMV}_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{M,v}[\text{DS}_M(v) = Mv] \geq \alpha \end{array} \right].$$

Then for every $\delta > 0$,

$$\text{OMV}_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & 4p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha)n) + O(n^2) \\ \text{query time:} & (4t + n) \cdot \text{poly}(1/\alpha) \cdot \text{poly} \log(1/\delta) \\ \text{success rate:} & \forall M, v : \Pr[\text{DS}_M(v) = Mv] \geq 1 - \delta \end{array} \right].$$

We stress that in the assumed data structure, the success rate asserts that for a *random* input M and query v , the data structure produces the correct answer with (an arbitrary small) probability $\alpha > 0$, where the probability is over (i) the random input M (ii) random query v (iii) and the randomness of the preprocessing and the query phases of the data structure. On the other hand, the conclusion holds for worst case inputs and queries. That is, *for every* input M and query v , the obtained data structure produces the correct

answer with high probability, where the probability is only over the randomness used in the preprocessing stage and the query phase of the data structure (i.e., with high probability we can compute *all* of the inputs).

To understand the parameters of the reduction, note that in the the OMV problem with $n \times n$ matrices, the preprocessing must be at least n^2 , as this is the size of the input matrix, and the query time must be at least n , as information-theoretically we need to output n field elements. Our worst-case to average-case reduction is essentially optimal in these parameters for a constant α , as a weak data structure that uses s memory cells and query time t is translated into a data structure that works for all inputs and all queries using space $4s + O(n^2)$ and query time $4t + O(n) = O(t)$. In fact, even for α as small as $\alpha = 1/n^{o(1)}$, the space complexity is increased by at most $O(n^2)$, and the query time is multiplied by at most $n^{o(1)}$.

1.1.4 Worst-case to weak-average-case reductions

In the following, we discuss how to obtain worst-case algorithms starting from a very weak, but natural, notion of average-case reductions that we discuss next.

Recall that in the standard definition of average-case data structures, the algorithm preprocesses its input and is then required to correctly answer all queries for an α -fraction of all possible inputs. However, in many cases (such as in the online matrix-vector multiplication problem), we only have an average-case guarantee on both inputs and queries. In this setting, we should first ask what is a natural notion of an average-case condition.

A strong requirement for an average-case algorithm in this case is to correctly answer *all queries* for at least α -fraction of the inputs. However, it is possible to only require the algorithm to correctly answer on an *average input and query*. That is, a *weak average-case* data structure for computing a function $f: \mathbb{F}^n \times [m] \rightarrow \mathbb{F}$ with success rate $\alpha > 0$ receives an input $x \in \mathbb{F}^n$, which is preprocessed into s memory cells. Then, given a query $i \in [m]$, the data structure $\text{DS}_x(i)$ outputs $y \in \mathbb{F}^{n'}$ such that $\Pr_{x \in \mathbb{F}^n, i \in [m]}[\text{DS}_x(i) = f(x, i)] \geq \alpha$.

The challenge in this setting is that the errors may be distributed between both the inputs and the queries. On one extreme, the error is concentrated on selected inputs, and then the data structure computes *all queries* correctly for α -fraction of the inputs. On the other extreme, the error is spread over all inputs, and then the data structure may only answer α -fraction of the queries on any inputs. Of course, the error could be distributed anywhere in between these extremes.

While we showed in [Theorem 2](#) that every linear problem has an efficient worst-case to average-case reduction in the data structures setting, in [Section 4.3.3](#) we will show that not

all linear (and non-linear) problems admit a worst-case to *weak*-average-case reductions in classical models of computation. Nevertheless, still within the classical setting, we are able to overcome this limitation for certain problems of interest. Note that in the more general model of quantum algorithms, as we shall see in [Theorem 5](#), we are able to take advantage of fast quantum algorithms for verifying matrix-vector products in combination with our additive combinatorics framework to show worst-case to average-case reductions for all linear problems.

One of the most-studied problems in static data structures is the polynomial evaluation problem [[KU08](#), [Lar12](#), [DKKS21](#)]. Here, one needs to preprocess a degree- d polynomial $q: \mathbb{F}^m \rightarrow \mathbb{F}$ into s memory cells, and then for a query $x \in \mathbb{F}^m$, quickly compute $q(x)$. We study the problem of evaluating a low degree polynomial in the regime where the average-case data structure might only succeed on a small α fraction of the queries (outside of the unique decoding regime, see discussion below). We show that we can use such an average-case data structure to obtain a worst-case data structure that can compute q on any $x \in \mathbb{F}^m$.

Theorem 4. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $\alpha := \alpha(n) \in (0, 1]$, and let $m, d \in \mathbb{N}$ be parameters. Consider the problem $\text{RM}_{\mathbb{F}, m, d}$ of evaluating polynomials of the form $q: \mathbb{F}^m \rightarrow \mathbb{F}$ of total degree d (i.e., the problem of evaluating the Reed-Muller encoding of block length $n = \binom{m+d}{d}$).*

Suppose that

$$\text{RM}_{\mathbb{F}, m, d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{q, x}[\text{DS}_q(x)] \geq \alpha \end{array} \right].$$

Then

$$\text{RM}_{\mathbb{F}, m, d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p + \exp(\log^4(1/\alpha)) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha) \log(n)) \\ \text{query time:} & O(|\mathbb{F}|^2 \cdot t + |\mathbb{F}| \log^4(1/\alpha) + |\mathbb{F}| \log(n)) \\ \text{success rate:} & \forall q, x : \Pr[\text{DS}_q(x) = q(x)] > 1 - O\left(\sqrt{\frac{d}{|\mathbb{F}|}}\right) \end{array} \right].$$

Here, similarly to [Theorem 3](#), the assumed data structure succeeds only for a small fraction of inputs and queries, while in the conclusion the data structure succeeds with high probability on *every input* and *every query*.

As for the effect of the reduction on the parameters, we see that for any $\alpha > 1/\text{poly}(n)$ the preprocessing time changes only by an additive $\text{poly}(n)$, the space complexity changes from s to $4s + \text{poly}(\log(n))$, and the query time changes from t to $O(|\mathbb{F}|^2 \cdot t + |\mathbb{F}| \cdot \text{poly} \log(n))$. In the data structure setting, the number of queries is usually polynomial in input length. Thus, in a typical setting of parameters for $\text{RM}_{\mathbb{F}, m, d}$, the field size is $|\mathbb{F}| = \text{poly}(\log n)$, and, therefore, the blow-up of $|\mathbb{F}|^2$ is not critical.

A coding-theoretic perspective. For small values of average-case success rate $\alpha > 0$, the polynomial evaluation problem can be cast as *list decoding with preprocessing*, by viewing the outputs of the query phase of the data structure as a function $h: \mathbb{F}^m \rightarrow \mathbb{F}$ that agrees with the input polynomial $q: \mathbb{F}^m \rightarrow \mathbb{F}$ on some small fraction of the queries, and the goal is to recover q from h .

Indeed, note that for a small $\alpha > 0$, if a function $h: \mathbb{F}^m \rightarrow \mathbb{F}$ agrees with some unknown low-degree polynomial q on α fraction of the inputs, then there are potentially $O(1/\alpha)$ possible low-degree polynomials that are equally close to h . Hence, without preprocessing, it is impossible to recover the original polynomial q . However, in the data structure setting, we can use the preprocessing to obtain auxiliary structural information that later allows us to transition from the list decoding regime to the unique decoding regime and, in turn, compute the values of the correct polynomial q with high probability (see more details in [Section 2.5](#)).

1.2 Quantum contributions

We have so far seen worst-case to average-case reductions for the specific problem of matrix vector multiplication, and for all linear problems when preprocessing into data structures is allowed. Do linear problems admit efficient worst-case to average-case reductions in a more general model of computation? Our last main contribution shows that, surprisingly, in the model of quantum computation the answer to this question is positive. In fact, we show that not only are such transformations possible, but that we can construct explicit and efficient quantum worst-case to average-case reductions *for all linear problems*.

A linear problem is characterized by a family of matrices $L_A := \{A_n \in \mathbb{F}^{m \times n}\}_{n \in \mathbb{N}}$, where $m = m(n)$ and on input $v \in \mathbb{F}^n$ the solution to the problem is the vector Av , omitting the subscript in A_n for readability. Linear problems constitute one of the most fundamental classes of problems, generalizing many important computational tasks such as polynomial evaluation (captured by tensor products of Vandermonde matrices), computing discrete

Fourier transformations (captured by the DFT matrix), homology (e.g. finding persistent cycles using a signed incidence matrix), and various computational tasks for error correcting codes (e.g. syndrome evaluation captured by the parity check matrix for linear codes).

In contrast to the setting of data structures (Section 1.1.2), in this section we work with general quantum algorithms, wherein the algorithm knows the matrix A beforehand but cannot perform any preprocessing. It receives v as input and must output the vector Av .

As mentioned above, this stands in stark contrast to the case of classical algorithms, where as we have seen in the preceding sections, such reductions are only known for a small number of specific problems or restricted models.

Generalizing the definition of BQP algorithms to fine-grained search problems in the standard way, we define an average-case quantum algorithm as a uniformly generated set of quantum circuits $\{C_n\}_{n \geq 1}$ which, upon measurement, output correctly with probability α , where the probability is taken over *both* the random input and the measurements. (See formal definitions in Section 5.2.)

Our main result in the quantum setting is a worst-case to average-case reduction which shows that for all linear problems, an efficient quantum algorithm that is only successful on a small fraction of its inputs can be explicitly transformed into a similarly efficient quantum algorithm that is correct with high probability *on all inputs*. Π_x below denotes an orthogonal projection on the output register of **ALG** that represents measuring the outcome $x \in \mathbb{F}^n$ in the standard basis.

Theorem 5 (Informally stated; see Theorem 5.2.2). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $L_A := \{A_n \in \mathbb{F}^{n \times n}\}_{n \in \mathbb{N}}$ be any linear problem, and **ALG** be an average-case quantum algorithm of (gate) complexity $T(n)$ for L_A satisfying*

$$\Pr_{v, \text{ALG}}[\text{ALG}(v) = Av] = \mathbb{E}_{v \in \mathbb{F}^n} [\|\Pi_{Av} \text{ALG} |v\rangle |0\rangle\|^2] \geq \alpha .$$

*Then, for every constant $\delta > 0$, there exists a worst-case quantum algorithm **ALG'** of (gate) complexity $(n^{3/2} + T(n)) \cdot \text{poly}(1/\alpha)$ that succeeds over all inputs with high probability, i.e.,*

$$\forall v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'}[\text{ALG}'(v) = Av] = \|\Pi_{Av} \text{ALG}' |v\rangle |0\rangle\|^2 \geq 1 - \delta .$$

Note that every linear problem can be trivially solved in time $O(n^2)$. Our result shows that any non-trivial (subquadratic) average-case quantum algorithm can be transformed into a non-trivial (sub-quadratic) quantum algorithm that works for *all* inputs. We also

remark that constructing worst-case to average-case reductions for linear problems becomes significantly harder as we consider smaller fields, as we discuss in the technical overview (Section 2.6). We stress that our reductions also hold for small fields, including $\mathbb{F}_2$.

Our proof of Theorem 5 is based on a machinery that we develop in the quantum query model. This allows us to obtain the following worst-case to average-case reduction for the fundamental and well-studied problem of Matrix-Vector Multiplication in the quantum query model [Bv06, Kot14].

Theorem 6 (Informally stated; see Theorem 5.2.3). *Let ALG be an average-case quantum query algorithm with oracle access to the entries of a matrix A and a vector v over a finite field $\mathbb{F}$. Suppose that ALG makes $q(n)$ queries and satisfies*

$$\Pr_{\substack{A,v, \\ \text{ALG}}}[\text{ALG}^{A,v} = Av] = \mathbb{E}_{\substack{A \in \mathbb{F}^{n \times n} \\ v \in \mathbb{F}^n}} \left[\|\Pi_{Av} \text{ALG}^{A,v} |0\rangle\|^2 \right] \geq \alpha .$$

Then, for every constant $\delta > 0$, there exists a worst-case quantum query algorithm ALG' with query complexity $(n^{3/2} + q(n)) \cdot \text{poly}(1/\alpha)$ that succeeds on all inputs with high probability, i.e.,

$$\forall A \in \mathbb{F}^{n \times n}, v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'}[(\text{ALG}')^{M,v} = Mv] = \|\Pi_{Mv} (\text{ALG}')^{M,v} |0\rangle\|^2 \geq 1 - \delta .$$

In the query model, it is known that the worst-case quantum query complexity of Matrix-Vector Multiplication has a tight lower bound of $\Theta(n^2)$ (see, e.g., [Kot14]). Hence as an immediate corollary of Theorem 6, we obtain a tight unconditional lower bound on the average-case quantum query complexity of Matrix-Vector multiplication, showing that the problem remains hard even if the quantum algorithm is only required to succeed on a small fraction of the inputs.

Corollary 1.2.1. *For every constant $\alpha > 0$, every average-case quantum query algorithm for Matrix-Vector Multiplication with success rate α must make $\Omega(n^2)$ queries.*

Remark 1.2.2 (Multi-bit outputs). *We stress that prior to this work, to the best of our knowledge, no techniques were known for proving non-trivial average-case quantum query lower bounds for functions with multi-bit outputs, such as Matrix-Vector Multiplication. However there are average-case query lower bounds known for problems including detecting collisions in random functions [Yue14, Zha15], and polynomial interpolation (which follows from a simple random self-reduction combined with [KK11, MP11]).*

Remark 1.2.3 (The class of all linear problems versus matrix-vector multiplication). *Note that in the setting of all linear problems (Theorem 5), the family of matrices is fixed, and only a vector is given as input. In this setting, many families of matrices (such as the Fourier Transform, the Walsh–Hadamard transform, and Polynomial evaluation) admit sub-quadratic time algorithms.*

In the Matrix-Vector Multiplication problem (the setting of Theorem 6 and Corollary 1.2.1), both a matrix and a vector are given as input, and this problem is $\Omega(n^2)$ -hard [Kot14] in the worst case. Our result for all linear problems (Theorem 5) provides an efficient worst-case algorithm for each family of matrices admitting an efficient average-case algorithm. Our result for the Matrix-Vector Multiplication problem (Theorem 6 and Corollary 1.2.1) shows that this problem remains hard even in the average case.

Chapter 2

Preliminaries

We now provide an overview of the main ideas and techniques that we use to obtain our results. For concreteness, we illustrate our techniques by first focusing on the matrix multiplication problem.

2.1 Classical reductions

We start in [Section 2.2](#), where we explain the challenge and discuss why the naive approach fails. In [Section 2.3](#) we present the technical components that lie at the heart of this work: *local correction lemmas via additive combinatorics*. Equipped with these technical tools, in [Section 2.4](#) we present the main ideas in our worst-case to average-case reduction for matrix multiplication. Finally, in [Section 2.5](#) we briefly discuss how to obtain the rest of our main results.

2.2 The challenge: low-agreement regime

Recall that in the matrix multiplication problem we are given two matrices $A, B \in \mathbb{F}^n$, and the goal is to compute their matrix product $A \cdot B$. For simplifying the exposition, unless specified otherwise, in this overview we restrict our attention to the field $\mathbb{F}_2$, and to constant values of the success rate parameter $\alpha > 0$ of average-case algorithms.

We would like to show that if there is an *average-case* algorithm ALG that can compute matrix multiplication for an α -fraction of pairs of matrices $A, B \in \mathbb{F}^n$ in time $T(n)$, then

there is a *worst-case* randomized algorithm ALG' that runs in time $O(T(n))$ and computes $A \cdot B$ with high probability for *every* pair of matrices A and B .

We start with the elementary case where the average-case guarantee is in the *high-agreement regime*, i.e., where the algorithm succeeds on, say, 99% of the inputs; that is,

$$\Pr_{A,B \in \mathbb{F}^{n \times n}} [\text{ALG}(A, B) = A \cdot B] \geq \alpha, \quad (2.1)$$

for $\alpha = 0.99$. In this case, a folklore local correction procedure (see, e.g., [BLR90]) will yield a worst-case algorithm that succeeds with high probability on all inputs. We next describe this procedure.

Given an *average-case* algorithm ALG satisfying Eq. (2.1) with $\alpha = 0.99$, consider the worst-case algorithm ALG' that receives any two matrices $A, B \in \mathbb{F}^{n \times n}$ and first samples uniformly at random two matrices $R, S \in \mathbb{F}^{n \times n}$. Next, writing $A = R + (A - R)$ and $B = S + (B - S)$, the algorithm ALG' computes

$$M = \text{ALG}(R, S) + \text{ALG}(A - R, S) + \text{ALG}(R, B - S) + \text{ALG}(A - R, B - S). \quad (2.2)$$

Denote by X the set of matrix pairs (A, B) for which $\text{ALG}(A, B) = A \cdot B$, and recall that by Eq. (2.1) the density of X is at least 0.99. Note that: (a) the matrices R , $A - R$, S , and $B - S$ are uniformly distributed, and (b) if the pairs (R, S) , $(A - R, S)$, $(R, B - S)$, and $(A - R, B - S)$ are in the set X , then by Eq. (2.2) we have $M = A \cdot B$, and the algorithm ALG' computes the multiplication correctly. Hence, by a union bound we have $\Pr[M = AB] \geq 1 - 4 \cdot 0.01 > 0.9$ for all matrices $A, B \in \mathbb{F}^{n \times n}$. Of course, the error probability can be further reduced by repeating the procedure and ruling by the majority.

Unfortunately, this argument breaks when the average-case guarantee is weaker; namely, in the *low-agreement regime*, where the algorithm succeeds on, say, only 1% of the inputs. Here, when trying to self-correct as above, the vast majority of random choices would lead to a wrong output, and so at first glance, the self-correction approach may seem completely hopeless.¹

Nevertheless, using more involved tools from additive combinatorics such as a probabilistic version of the quasi-polynomial Bogolyubov-Ruzsa lemma that we show, as well

¹Indeed, consider the counterexample where the average-case algorithm $\text{ALG}(A, B)$ outputs $A \cdot B$ in case the first element of A is 0 and returns the zero matrix in case the first element of A is 1. Note that in this case $\Pr_{A,B \in \mathbb{F}^{n \times n}} [\text{ALG}(A, B) = A \cdot B] \geq 1/2$, yet no decomposition of $A = \sum_i A_i$ and $B = \sum_i B_i$ as described above could self-correct matrix multiplication where the first element of A is 1. Indeed, any such composition would have an A_i with the first element 1, where $\text{ALG}(A_i, B_j)$ fails.

as tools such as small-biased sample spaces and the Goldreich-Levin algorithm, we can construct different local correction procedures that work in the *low-agreement regime*. We proceed to describe our framework for local correction using the aforementioned tools.

2.3 Local correction via additive combinatorics

Additive combinatorics studies approximate notions of algebraic structures via the perspective of combinatorics, number theory, harmonic analysis and ergodic theory. Most importantly for us, it provides tools for transitioning between algebraic and combinatorial notions of approximate subgroups with only a small loss in the underlying parameters (see surveys [Lov15, Lov17]).

The starting point of our approach for local correction is a fundamental result in additive combinatorics, known as *Bogolyubov's lemma*, which shows that the 4-ary sumset of any dense set in $\mathbb{F}_2^n$ contains a large linear subspace. More accurately, recall that the sumset of a set X is defined as $X+X = \{x_1+x_2 : x_1, x_2 \in X\}$, and similarly $4X = \{x_1+x_2+x_3+x_4 : x_1, x_2, x_3, x_4 \in X\}$. These quantities can be thought of as quantifying a combinatorial analogue of an approximate subgroup. Bogolyubov's lemma states that for any subset $X \subseteq \mathbb{F}_2^n$ of density $|X|/2^n \geq \alpha$, there exists a subspace $V \subseteq 4X$ of dimension at least $n - \alpha^{-2}$.

We will show that statements of the above form can be used towards obtaining a far stronger local correction paradigm than the one outlined in Section 2.2. To see the initial intuition, consider an average-case algorithm that is guaranteed to correctly compute α -fraction of the inputs, and denote by X the set of these correctly computed inputs. Then $|X|/2^n \geq \alpha$, and Bogolyubov's lemma shows that there exists a large subspace V such that every $v \in V$ can be expressed as a sum of four elements in X , each of which can be computed correctly by the average-case algorithm.

The approach above suggests a paradigm for local correction, however, there are several non-trivial problems in implementing this idea. For starters, how could we handle inputs that lay *outside of the subspace V* ? To name a few others: how can we amplify the success probability in the low-agreement regime? How do we algorithmically obtain the decomposition? Can we handle finite fields beyond $\mathbb{F}_2^n$? How do we handle average-case where the success rate α is sub-constant?

Indeed, for our worst-case to average-case reductions, we will need local correction lemmas with stronger structural properties than those admitted by Bogolyubov's lemma, as well as new ideas for each one of the settings. In the following, we discuss the main hurdles

for the foregoing approach and the tools that are needed to overcome them, leading to our main technical tool, which is a probabilistic version of the quasi-polynomial Bogolyubov-Ruzsa lemma that we obtain. Then, we present our framework for local correction using these techniques. Finally, in [Sections 2.4](#) and [2.5](#) we show the additional ideas that are necessary for applying the local correction lemmas in the settings of matrix multiplication, online matrix-vector multiplication, data structures, and quantum reductions for linear problems.

A probabilistic Bogolyubov lemma. An immediate problem with the aforementioned local correction scheme is that while Bogolyubov’s lemma asserts that *there exists* a decomposition of each input into a sum of four elements in X , it does not tell us how to obtain this decomposition.

Toward this end, we further show that each vector $v \in V$ has many “representations” as a sum of four elements from X . This way, for any $v \in V$ we can efficiently sample a representation $v = x_1 + x_2 + x_3 + x_4$, where each $x_i \in X$. More accurately, let $X \subseteq \mathbb{F}_2^n$ be a set of density α , let $R = \{r \in \mathbb{F}^n \setminus \{0\} : |\hat{1}_X(r)| \geq \alpha^{3/2}\}$, and let $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$ be a linear subspace defined by R . Then $|R| \leq 1/\alpha^2$ and for all $v \in V$ it holds that

$$\Pr_{x_1, x_2, x_3} [x_1, x_2, x_3, v - x_1 - x_2 - x_3 \in X] \geq \alpha^5.$$

Sparse-shift subspace decomposition. The probabilistic Bogolyubov lemma allows us to locally correct inputs inside the subspace $V \subseteq 4X$. However, we need to be able to handle any vector in the field. Towards that end, we show an algebraic lemma that allows us to decompose each element of the field into a sum of an element v in the subspace V and a *sparse* shift-vector s . More accurately, let $R \subseteq \mathbb{F}^n \setminus \{\vec{0}\}$ and $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$. We show that there exists a collection of $t \leq |R|$ vectors $B = \{b_1, \dots, b_t\}$, $b_i \in \mathbb{F}^n$ and indices $k_1, \dots, k_t \in [n]$ such that $\text{span}(B) = \text{span}(R)$ and every vector $y \in \mathbb{F}^n$ can be written as $y = v + s$, where $v \in V$ and $s = \sum_{j=1}^t c_j \cdot \vec{e}_{k_j}$ for $c_j = \langle y, b_j \rangle$ and $\vec{e}_{k_j}$ is a unit vector.

We stress that the sparsity of the decomposition is essential to our applications, as we cannot locally correct the shift part of the decomposition, and instead we need to compute it explicitly. We remark that for matrix multiplication we can obtain a stronger guarantee by dealing with matrices outside of the subspace V via a low-rank random matrix shifts (see [Section 2.4](#)).

Subspace computation via the Goldreich-Levin lemma. In order to perform local correction using additive combinatorics machinery as above while maintaining computational efficiency, we need to be able to compute the aforementioned basis $b_1, \dots, b_t \in \mathbb{F}^n$ and indices $k_1, \dots, k_t \in [n]$ efficiently. We note that, in essence, this problem reduces to learning the heavy Fourier coefficients of the set X . Thus, using ideas from [BRTW14] and an extension of the Goldreich-Levin algorithm to arbitrary finite fields, we can perform the latter in a computationally efficient way.

Probabilistic quasi-polynomial Bogolyubov-Ruzsa lemma. The main weakness of Bogolyubov's lemma is that the co-dimension of the subspace that it admits is polynomial in $1/\alpha$, where α is the success rate of the average-case algorithm. While this dependency on α allows us to locally correct in the 1% agreement regime, it becomes degenerate when α tends to 0 rapidly.

A natural first step towards overcoming this barrier is to use a seminal result due to Sanders [San12], known as the *quasi-polynomial Bogolyubov-Ruzsa lemma*, which shows the existence of a subspace whose co-dimension's dependency on $1/\alpha$ is *exponentially* better. That is, the lemma shows that for a set $X \subseteq \mathbb{F}_2^n$ of size $\alpha \cdot |\mathbb{F}_2|^n$, where $\alpha \in (0, 1]$, there exists a subspace $V \subseteq \mathbb{F}_2^n$ of dimension $\dim(V) \geq n - O(\log^4(1/\alpha))$ such that $V \subseteq 4X$. However, as in the case of Bogolyubov's lemma, we have the problem that the statement is only *existential*.

We thus prove a probabilistic version of the quasi-polynomial Bogolyubov-Ruzsa lemma (see Lemma 3.1.3) over any field $\mathbb{F} = \mathbb{F}_p$, which asserts that for an α -dense set $X \subseteq \mathbb{F}^n$, there exists a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) \geq n - O(\log^4(1/\alpha))$ such that for all $v \in V$ it holds that

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2 \in A, x_3, x_4 \in -A] \geq \Omega(\alpha^5) ,$$

where $x_4 = v - x_1 - x_2 - x_3$. Furthermore, by combining the techniques above, we show that given a query access to the set X , there is an algorithm that runs in time $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ and with probability $1 - \delta$ computes a set of vectors $R \subseteq \mathbb{F}^n$ such that $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$.

We are grateful to Tom Sanders for providing us with the argument for proving this lemma, and we provide the proof in Chapter 3.

Our classical local correction lemma. We are now ready to provide an informal statement of our local correction lemma, which builds on the machinery above, and in

particular, on the probabilistic quasi-polynomial Bogolyubov-Ruzsa lemma.

Loosely speaking, our local correction allows us to decompose any vector $y \in \mathbb{F}^n$ as a linear combination of the form

$$y = x_1 + x_2 - (x_3 + x_4) + s ,$$

where $x_1, x_2, x_3, x_4 \in X$ and $s \in \mathbb{F}^n$ is a *sparse* vector.

Lemma 2.3.1 (informally stated, see [Lemma 3.3.1](#)). *For a field $\mathbb{F} = \mathbb{F}_p$ and α -dense set $X \subseteq \mathbb{F}^n$, there exists $t \leq 1/\alpha^2$ vectors $b_1, \dots, b_t \in \mathbb{F}_2^n$ and indices $k_1, \dots, k_t \in [n]$ satisfying the following. Given a vector $y \in \mathbb{F}_2^n$, let $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$ we have*

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2 \in X, x_3, x_4 \in -X] \geq \Omega(\alpha^5) ,$$

where $x_4 = y - s - x_1 - x_2 - x_3$.

Furthermore, given an oracle that computes $1_X(x)$ with probability at least $2/3$, there exists an algorithm that makes $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ oracle calls and field operations, and with probability at least $1 - \delta$ outputs $b_1, \dots, b_t$ and $k_1, \dots, k_t$.

Variants of this local correction lemmas lie at the heart of all our average-case to worst-case reductions, which we discuss next.

2.4 Illustrating example: matrix multiplication

We present a high-level overview of our reductions for matrix multiplication, which illustrates the key ideas that go into the proof. Let **ALG** be an average-case algorithm that can compute matrix multiplication for an α -fraction of all pairs of matrices $A, B \in \mathbb{F}^n$ in time $T(n)$. We use the *average-case* algorithm **ALG** to construct a *worst-case* randomized algorithm **ALG'** that runs in time $O(T(n))$ and computes $A \cdot B$ with high probability for *every* pair of matrices A and B . For simplicity of the exposition, in this overview we make the following assumptions: (1) the algorithm **ALG** is *deterministic*, (2) the input is a pair (A, B) such that A is a matrix satisfying $\Pr_{B'}[\mathbf{ALG}(A, B') = A \cdot B'] \geq \alpha$, (3) the success rate α is a constant, and (4) the field $\mathbb{F}$ is $\mathbb{F}_2$.

We start by noting two simple facts. First, given the algorithm's (potentially wrong) output $\mathbf{ALG}(A, B)$, we can efficiently check whether the computation is correct using Freivalds' algorithm ([Lemma 4.1.1](#)). Second, denoting by $X = \{B' \in \mathbb{F}_2^{n \times n} : \mathbf{ALG}(A, B') =$

$A \cdot B'$ the set of “good” matrices, we have that if $B \in X$, then the average-case algorithm correctly outputs $\text{ALG}(A, B) = A \cdot B$. Hence, the main challenge is in dealing with the case that $B \notin X$, in which we need to locally correct the value of the multiplication.

Local correction via Bogolyubov’s lemma. The first idea is to reduce the problem to the case where the set of good matrices contains a large subspace, and hence admits local correction, as discussed in [Section 2.3](#). Specifically, by the probabilistic Bogolyubov lemma, given X we can choose a subspace $V \subseteq \mathbb{F}_2^{n \times n}$ of matrices, where $\dim(V) \geq n^2 - 1/\alpha^2$, such that for any $B' \in V$, if we sample M_1, M_2, M_3 uniformly at random, then

$$\Pr[M_1, M_2, M_3, M_4 \in X] \geq \alpha^5, \text{ where } M_4 = B' - M_1 - M_2 - M_3.$$

Note that if the matrices M_1, M_2, M_3, M_4 produced by our sampling are all in the set of good matrices X , then we can self-correct the value of $\text{ALG}(A, B')$ by evaluating $\{\text{ALG}(A, M_i)\}_{i \in [4]}$ and computing the linear combination

$$\sum_{i=1}^4 \text{ALG}(A, M_i) = \sum_{i=1}^4 A \cdot M_i = A \cdot \left(\sum_{i=1}^4 M_i \right) = A \cdot B'.$$

Note that this event is only guaranteed to occur with probability α^5 , which is far smaller than $1/2$. Nevertheless, since we can verify the computation using Freivalds’ algorithm, we can boost this probability to be arbitrarily close to 1 by repeating the random sampling step $O(1/\alpha^5)$ times, each time computing $\sum_{i=1}^4 \text{ALG}(A, M_i)$ and verifying if the obtained result is indeed correct using Freivalds’ algorithm. Therefore, if B belongs to the (unknown) subspace V , then the algorithm described above indeed computes $A \cdot B$ with high probability in time $O(T(n)/\text{poly}(\alpha)) = O(T(n))$.

However, the approach above does not work for matrices B that do not lie in the subspace V described above. To deal with this case, our next goal is to “shift” the matrix into the subspace V using low-rank random shifts, which can then be computed efficiently and used for local correction. We describe this procedure next.

Low-rank random matrix shifts. We start by making the following key observation: if we have an arbitrary matrix A , and a matrix $B \in \mathbb{F}^{n \times n}$ of rank k , then their product AB can be computed in time $O(kn^2)$, given a rank- k decomposition of B . Details follow.

To see this, suppose that the first k columns of B denoted by $(B_i)_{i=1}^k$, are linearly independent, and for each of the remaining $n - k$ columns $(B_j)_{j=k+1}^n$, we know the linear

combination $B_j = \sum_{i=1}^k d_{i,j} \cdot B_i$ for some coefficients $d_{i,j} \in \mathbb{F}$. We can first multiply A by each of the k linearly independent columns of B . Then, to compute the remaining columns, for each $i = 1, \dots, k$ let $C_i = A \cdot B_i$ be the i th column of the matrix $C = AB$, and observe that if $B_j = \sum_{i=1}^k d_{i,j} B_i$, then $C_j = A \cdot B_j = A \cdot (\sum_{i=1}^k d_{i,j} B_i) = \sum_{i=1}^k d_{i,j} \cdot C_i$, which can be computed in $O(kn)$ time for each j . Therefore the total running time of multiplying A by B is $O(kn^2)$.

We are now ready to describe our method for shifting the matrices into the subspace V using low-rank matrices, capitalizing on the observation above. Given the matrix B (that is, possibly, not in V), we sample a random matrix $R_B \in \mathbb{F}^{n \times n}$ of rank $2k$ by randomly choosing $2k$ columns and filling them with uniformly random field elements. Note that with high probability these $2k$ columns are linearly independent. Then, we let the rest of the columns be random linear combinations of the first $2k$ columns we chose. We observe that if $\dim(V) = n - k$, then

$$\Pr[B + R_B \in V] \geq \frac{1}{2^{|\mathbb{F}|^k}}.$$

If indeed $B + R_B \in V$, then we can compute $A \cdot (B + R_B)$ using the procedure discussed above, by writing $B + R_B$ as a sum of 4 random matrices $B + R_B = M_1 + M_2 + M_3 + M_4$, applying $\text{ALG}(A, M_i)$ for each $i = 1 \dots 4$, and using Freivalds' algorithm to efficiently check if the produced output is correct or not.

Note that since we have a lower bound on the probability that $B + R_B$ belongs to the desired subspace, we have an upper bound on the expected number of attempts required until this event occurs. When we obtain such low-rank matrix shifts, which we verify using Freivalds' algorithm, we proceed by computing $A \cdot R_B$. Since R_B is a matrix of rank at most $2k$, the total running time of this will be $O(kn^2)$. Finally, we return

$$\text{ALG}(A, B + R_B) - A \cdot R_B,$$

which indeed produces the correct answer assuming that $\text{ALG}(A, B + R_B)$ is correct.

Remark 2.4.1. *The discussion above made the simplifying assumption that the inputs we are getting are pairs (A, B) such that A is a matrix satisfying $\Pr_{B'}[\text{ALG}(A, B') = A \cdot B'] \geq \alpha$. The actual proof require also handling the inputs for which the matrix A does not satisfy this requirement, which is done using similar ideas by applying the local correction procedure first to A and then to B .*

2.5 Beyond matrix multiplication

We conclude the technical overview by briefly sketching some of the key ideas in the rest of our worst-case to average-case reductions, building on the local correction lemmas outlined in [Section 2.3](#). Below we assume that all data structures are deterministic, but by standard techniques this assumption is without loss of generality. We start with the simplest setting, and then proceed to the more involved ones.

Worst-case to average-case reductions for all linear data structure problems.

The setting here is the closest to that of matrix multiplication. Let DS_A be an average-case data structure for a linear problem defined by A , where we preprocess an input vector x and the answer to query i is $\langle A_i, x \rangle$, and A_i is the i th row of A .

Given a vector $y \in \mathbb{F}^n$, we use our local correction lemma to obtain a decomposition of the form $y = x_1 + x_2 - (x_3 + x_4) + v$, where $x_1, x_2, x_3, x_4 \in X$ (i.e., on which $\text{DS}_{x_j}(i) = \langle A_i, x_j \rangle$ for all i) and a sparse shift vector $v = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$. We then preprocess each of the x_j 's by applying DS_A to it, and we also compute $\langle A_i, v \rangle$ efficiently by using its sparse representation. The idea is that by the linearity of the problem, we can locally correct according to $\sum_{j=1}^4 \text{DS}_{x_j}(i) + \langle A_i, v \rangle$.

It is important to note that, unlike in the setting of matrix multiplication, we cannot use the random low-rank matrix shifts, nor Freivald's algorithm for verification. However, this is where we rely on the sparse subspace decomposition to shift the input into the subspace V implied by the quasi-polynomial Bogolyubov-Ruzsa lemma. In addition, instead of relying on Freivald's algorithm for verification, here we use the guarantee about the correctness of computation in the subspace V together with the sparsity of the shift vector, which allows us to correct its corresponding contribution via explicit computation. See details in [Section 4.3.1](#).

Online matrix-vector multiplication (OMV). The online setting of the OMV problem poses several additional challenges. Recall that in the average-case reductions above, the input is a vector $x \in \mathbb{F}_2^n$, each query is a coordinate $i \in [n]$, and the matrix $M \in \mathbb{F}_2^{n \times n}$ is a hard-coded parameter. In the OMV problem, the matrix M is the *input*, the vector x is the *query*, and answer to a query is not a scalar but rather a *vector*. Hence we need to use a two-step local correction where we first decompose the matrix and then decompose the vector. Observe that we can use our additive combinatorics mechanism to preprocess the matrix M and get a description of the subspace V that it asserts, as well as the formula that is required to compute the shift vector s given x , but the problem is that here we

cannot preprocess x , as it arrives online. Thus, in the query phase, when the algorithm receives x , we want to find the decomposition $x = x_1 + x_2 + x_3 + x_4 + s$. We then compute the shift vector s , and then sample x_i 's whose sum is $x - s$. However this leaves us with the task of checking that all of the x_i 's are computed correctly. To this end, we rely on a generalization of *small-bias sample spaces* to finite fields in order to obtain an efficient verification procedure. See details in [Section 4.2](#).

Weak-average-case reductions. As discussed in the introduction, in the setting of weak-average-case we cannot expect a reduction for all linear problems. In turn, this leads to substantially different techniques. We concentrate on the multivariate polynomial evaluation problem. Here, we are given a polynomial $p: \mathbb{F}^m \rightarrow \mathbb{F}$ of degree d , where for simplicity, in this overview we fix the parameters $d = \log(n)$, $|\mathbb{F}| = \text{poly}(\log(n))$, and $m = \log(n)/\log \log(n)$, so that we encode n field elements using a codeword of length $\text{poly}(n)$, and the distance is $1 - dm/|\mathbb{F}| > 0.99$. The polynomial is given as input by its $n = d^m$ coefficients, the queries are of the form $x \in \mathbb{F}^m$, and the goal is to output $p(x)$. The key difficulty here, is that for small values of the average-case rate $\alpha > 0$, we need to be able to deal with the *list decoding regime* (see discussion in [Section 1.1.4](#)).

The first step is to rely on our additive combinatorics local correction tools similar to the OMV reduction. Here the idea is to preprocess the polynomial p and obtain a decomposition of the form $p = p_1 + p_2 + p_3 + p_4 + s$, where again s is a sparse shift-vector. We then construct a data structure for each p_i . However, since we cannot process the queries $x \in \mathbb{F}^m$, we are left with the task of locally correcting the noisy polynomials $\{p_i\}$. If a polynomial p_i is only slightly corrupted (i.e., within the unique decoding regime), we can easily locally correct it without using any preprocessing. However, we also need to deal with noisy polynomials p_i in the *list decoding regime* in which only α -fraction of the points are evaluated correctly, for an arbitrarily small α .

We overcome the difficulty above by capitalizing the preprocessing power of the data structure. Namely, we will show how to boost the success probability from the list-decoding regime to the unique-decoding regime, in which case we can perfectly correct the polynomial via the local correction of the Reed–Muller code. The key idea is that by the generalized Johnson bound, there is only a list of $O(1)$ codewords that agree with the average-case data structure on at least $\alpha/2$ -fraction of the points. We thus fix a reference point $w \in \mathbb{F}^m$ and explicitly compute the correct value of $p_i(w)$. Next, we sample a random point r and query the points of line $\ell_{x,w}$ incident to r and the reference point w . Then, we consider the list (of size $O(1)$) of all low-degree univariate polynomials that agree with the queried points on $\ell_{x,w}$, and trim the list by removing each polynomial that does not agree on the

reference point. Using the sampling properties of lines in multivariate polynomials, we can show that answering accordingly to the remaining polynomials in the list would yield the right value with high probability.

2.6 Quantum reductions

We now provide an overview of the techniques used in achieving our quantum reductions, highlighting the main conceptual and technical ideas. To capture all linear problems in the quantum setting presents non-trivial challenges that require new ideas on three fronts: (1) additive combinatorics techniques, (2) new quantum algorithms, and (3) complexity theoretic reduction techniques. We believe these techniques are of independent interest and can be used in other settings.

In [Section 2.7](#), we discuss our new quantum local correction lemma, which is based on a robust, probabilistic version of Bogolyubov’s lemma. Our quantum local correction lemma crucially relies on four quantum algorithms, which we outline in [Section 2.8](#), that allow us to efficiently flag correct solutions in superposition, construct noisy oracles for approximate indicator functions, efficiently sample good inputs, and learn Bogolyubov subspaces from noisy set-indicating quantum oracles. Finally, in [Section 2.9](#), we discuss how to apply the quantum local correction lemma to obtain the desired worst-case to average-case reductions.

In this overview, for simplicity of exposition, we will focus on the proof of [Theorem 6](#) in the quantum query model; we in fact optimise both the query and gate complexities, so that we can then show how to extend the result to (uniform) quantum circuits for all linear problems.

Recall that the Matrix-Vector Multiplication problem (MvM) is defined as follows. The input to the problem is a vector $v \in \mathbb{F}^n$, and oracle access to a matrix $M \in \mathbb{F}^{n \times n}$, and the goal is the output the matrix-vector product Mv .

Let **ALG** be an *average-case* quantum algorithm with oracle access to M and v , defined by

$$U_M |j, k, z\rangle = |j, k, z \oplus M_{jk}\rangle \quad \text{and} \quad U_v |j, z\rangle = |j, z \oplus v_j\rangle,$$

for all indices $j, k \in [n]$ and $z \in \mathbb{F}$. Suppose that using q queries to M and v , the quantum algorithm **ALG** satisfies

$$\Pr_{\substack{M, v, \\ \text{ALG}}} [\text{ALG}^{M, v} = Mv] = \mathbb{E}_{\substack{M \in \mathbb{F}^{n \times n} \\ v \in \mathbb{F}^n}} \left[\|\Pi_{Mv} \text{ALG}^{M, v} |0\rangle\|^2 \right] \geq \alpha,$$

where Π_{Mv} is an orthogonal projection on the output register of **ALG** that indicates whether the algorithm outputs the correct answer.

We would like to explicitly transform **ALG** into a *worst-case* quantum algorithm **ALG'** that computes Mv with high probability for *every* matrix M and vector $v \in \mathbb{F}^n$; that is, show that for every constant $\delta > 0$, there exists a worst-case quantum algorithm **ALG'** satisfying

$$\forall M \in \mathbb{F}^{n \times n}, v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'}[(\text{ALG}')^{M,v} = Mv] = \|\Pi_{Mv}(\text{ALG}')^{M,v} |0\rangle\|^2 \geq 1 - \delta .$$

To simplify the discussion, unless specified otherwise, in this overview we restrict our attention to the field $\mathbb{F} = \mathbb{F}_2$, and to arbitrarily small constant values of the success rate parameter $\alpha > 0$ of average-case algorithms (say $\alpha = 0.01$). We also make the simplifying assumption that the average-case algorithm **ALG** receives a “good” matrix $M \in \mathbb{F}^{n \times n}$ for which it is successful with probability α , taken over the measurement and the random input vector $v \in \mathbb{F}^n$; that is,

$$\Pr_v[\text{ALG}^{M,v} = Mv] = \mathbb{E}_{v \in \mathbb{F}^n} \left[\|\Pi_{Mv} \text{ALG}^{M,v} |0\rangle\|^2 \right] \geq \alpha .$$

Note that by an averaging argument, there exists a set of size $(\alpha \cdot 2^n)/2$ of input vectors $v \in \mathbb{F}^n$ on which **ALG** correctly computes the output with probability at least $\alpha/2$; denote this set by X . Next, observe that Bogolyubov’s lemma shows that there exists a large subspace V such that every $v \in V$ can be decomposed as

$$v = x_1 + x_2 + x_3 + x_4, \text{ for } x_1, x_2, x_3, x_4 \in X . \quad (2.3)$$

Recall that for each $x_i \in X$, the answer Mx_i can be computed correctly by the average-case algorithm with probability at least $\alpha/2$. This suggests the natural approach of locally correcting each $v \in V$ using four inputs upon which the average-case algorithm has a non-negligible success probability.

Note that Bogolyubov’s lemma only guarantees the existence of a decomposition into elements of X , whereas we need to explicitly obtain such a decomposition, as well as the fact that the argument above only holds for correcting inputs that are *inside the Bogolyubov subspace* V , whereas we need to locally correct *all inputs*.

Unfortunately, the classical framework that we develop ([Section 2.3](#)) fails to address the aforementioned problems in the setting of general linear problems. Indeed, it is not clear whether the ambitious task of constructing worst-case to average-case reductions for

all linear problems is at all possible for classical algorithms.

2.7 Quantum local correction via a robust Bogolyubov lemma

Our starting point for addressing the difficulties mentioned before is a new *robust* and *probabilistic* analogue of Bogolyubov’s lemma. These additional structural properties will in turn enable us to deal with inputs outside of the Bogolyubov subspace and efficiently obtain explicit decompositions of inputs inside the Bogolyubov subspace into inputs upon which the average-case algorithm is correct.

Then, using the robust probabilistic Bogolyubov lemma, we will show a local correction lemma for quantum algorithms for linear problems. Our quantum local correction lemma will crucially make use of the robustness and probabilistic properties, as well as the four quantum procedures that we present in [Section 2.8](#).

A robust probabilistic Bogolyubov lemma. Recall that our high-level idea for locally correcting faulty inputs in the subspace V guaranteed by Bogolyubov’s lemma proceeds by decomposing vectors $v \in V$ into a linear combination of *good inputs* from the set X . However, to prove our quantum local correction lemma, we need to be able to: (1) deal with vectors *outside of the Bogolyubov subspace* V , and (2) efficiently obtain an explicit decomposition of vectors inside V into good inputs in X . To this end, we shall first need to strengthen Bogolyubov’s lemma to obtain the following structural properties.

- *Robustness.* Our local correction lemma relies on learning the heavy part of the Fourier spectrum of a noisy representation of an approximate indicator of the set X in superposition, using a quantum procedure that we discuss in [Section 2.8](#). In this setting, we can only learn the heavy Fourier coefficients of a function g satisfying

$$\forall S \subset [n] \quad \left| |\widehat{1_X}(S)|^2 - |\hat{g}(S)|^2 \right| \leq \epsilon ,$$

for a sufficiently small ϵ . In fact, each Fourier coefficient we obtain may come from a *different function* g . To deal with this problem, which is *unique to the quantum setting*, we need a version of Bogolyubov’s lemma where the subspace is defined via a robust set of linear constraints that can accommodate for the noisy Fourier spectrum.

- *Probabilistic decomposition.* Bogolyubov's lemma shows the *existence* of a large subspace $V \subset 4X$, admitting a linear decomposition of each $v \in V$ into four vectors from the set of good inputs X upon which the average-case algorithm is correct. However, we need to efficiently obtain an explicit decomposition $v = x_1 + x_2 + x_3 + x_4$, where each $x_i \in X$, for each vector $v \in V$. Hence, we show that each vector admits sufficiently many such decompositions such that one can be efficiently sampled using a quantum sampling procedure that we show in [Section 2.8](#).

We thus prove the following *robust probabilistic Bogolyubov lemma*, which will allow us to deal with the challenges above. Given an unstructured set $X \subseteq \mathbb{F}^n$, denote the heavy part, exceeding a threshold γ , of the Fourier spectrum of X by $\text{Spec}_X(\gamma) = \{r \in \mathbb{F}^n \setminus \{0\} : |\hat{1}_X(r)| \geq \gamma\}$.

Lemma 2.7.1 (Informally stated, see [Lemma 3.4.1](#)). *For every $X \subseteq \mathbb{F}^n$ of density at least α , let $R \subseteq \mathbb{F}^n$ be a set such that $\text{Spec}_X(\alpha^{3/2}) \subseteq R \subseteq \text{Spec}_X(\frac{\alpha^{3/2}}{2})$. Let $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$. Then $\dim(V) \geq n - O(\alpha^{-2})$, and for all $v \in V$ it holds that*

$$\Pr_{x_1, x_2, x_3 \in X} [v - x_1 - x_2 - x_3 \in X] \geq \alpha^2.$$

We stress that the probability in [Lemma 2.7.1](#) is taken over x_1, x_2, x_3 that are uniformly sampled from X (rather than $\mathbb{F}^n$). To efficiently obtain such a decomposition, we can sample x_1, x_2, x_3 using the quantum sampling procedure shown in [Section 2.8](#).

Locally correcting outside of the Bogolyubov subspace. While the robust probabilistic Bogolyubov lemma allows us to locally correct inputs inside the subspace $V \subseteq 4X$, our goal is to obtain a worst-case quantum algorithm, hence we need to be able to handle any vector $v \in \mathbb{F}^n$, and not just those in V .

Towards this end, we would like to shift each vector $v \in \mathbb{F}^n \setminus V$ into the Bogolyubov subspace. Using the robustness property of [Lemma 2.7.1](#) and the quantum algorithm for learning Bogolyubov subspaces from a noisy representation of indicator functions in superposition (see [Section 2.8](#)), we further decompose each vector in $\mathbb{F}^n$ into a sum of elements in the subspace V and a *sparse* shift-vector s , which can then be corrected for by computing Ms efficiently due to its sparsity, similar to the sparse-shift subspace decomposition used in [Section 2.3](#).

The quantum local correction lemma. Putting together all of the components above, we can now state our quantum local correction lemma, which builds upon the robust

probabilistic version of Bogolyubov's lemma and the quantum procedures that we will present in [Section 2.8](#).

Loosely speaking, our local correction lemma allows us to efficiently obtain an explicit decomposition of any vector $v \in \mathbb{F}^n$ into a linear combination of the form

$$v = x_1 + x_2 + x_3 + x_4 + s ,$$

where $x_1, x_2, x_3, x_4 \in X$ and $s \in \mathbb{F}^n$ is a *sparse* vector.

Specifically, in [Section 2.8](#), we will construct a noisy quantum oracle for approximating the indicator $1_X(v)$, quantum algorithm for uniformly sampling from X , a quantum verification procedure $\tilde{O}_X$ that for each $v \in \mathbb{F}^n$ computes the indicator $1_X(v)$ correctly with high probability, and a quantum algorithm that learns the Bogolyubov subspace from the noisy oracle that approximates 1_X . By combining these four ingredients we prove the following result.

Lemma 2.7.2 (informally stated, see [Lemma 3.5.1](#)). *For a field $\mathbb{F} = \mathbb{F}_p$ and α -dense set $X \subseteq \mathbb{F}^n$, there exists $t \leq 4/\alpha^2$ vectors $b_1, \dots, b_t \in \mathbb{F}_2^n$ and indices $k_1, \dots, k_t \in [n]$ satisfying the following. Given a vector $y \in \mathbb{F}^n$, define $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$, where $(\vec{e}_i)_{i \in [n]}$ is the standard basis. Then,*

$$\Pr_{x_1, x_2, x_3 \in X} [x_4 \in X] \geq \alpha^2 ,$$

where x_4 is the vector such that $y = s + x_1 + x_2 + x_3 + x_4$.

Furthermore, there exists a quantum algorithm that calls the quantum verification procedure $\tilde{O}_X$ for $O(\log(1/\delta) \cdot (1/\alpha^5))$ times, uses $n \cdot \text{poly}(1/\alpha) \cdot \text{poly} \log |\mathbb{F}|$ additional elementary gates, and with probability at least $1 - \delta$ returns the vectors $b_1, \dots, b_t \in \mathbb{F}^n$ and indices $k_1, \dots, k_t \in [n]$.

This quantum local correction lemma forms the cornerstone of our quantum average-case to worst-case reductions, as we explain in [Section 2.9](#). However, before we can turn to the details of the reduction, we need to address three gaps that still remain: How can we efficiently verify Matrix-Vector products, so that we can amplify the success probability of quantum algorithms that are only correct with probability α (which could tend to zero)? Can we efficiently sample from the set of good inputs X ? How can we learn the Bogolyubov subspace from a noisy quantum oracle that encodes an approximation of the indicator 1_X ?

2.8 A toolkit of quantum algorithms for local correction

We address the foregoing questions by presenting the key quantum procedures that are required for our quantum local correction lemma. We stress that these solutions employ the power of quantum algorithms, and indeed, the lack of such procedures in the classical setting is a bottleneck that blocks the path to general results such as [Theorem 5](#) and [Theorem 6](#) for *classical* algorithms.

We achieve quantum speedups for the following four tasks, which are required for our quantum local correction lemma:

1. Flagging correct matrix-vector products in superposition.
2. Constructing noisy quantum oracles approximating the indicator function 1_X .
3. Sampling of vectors from the set X of good vectors.
4. Learning Bogolyubov subspaces from noisy quantum oracles.

Note that it is essential to perform the above tasks in complexity $o(n^2)$, otherwise the entire reduction will become trivial; indeed our quantum algorithms make at most $O(n^{3/2})$ queries and use at most $\tilde{O}(n^{3/2})$ additional one-qubit and two-qubit gates. We now proceed to briefly describe the above quantum procedures, and offer a quick glimpse at our techniques for obtaining them. See [Section 5.3](#) for details.

Flagging correct matrix-vector products in superposition. The first tool we shall need is a quantum subroutine for verifying whether a vector $b \in \mathbb{F}^n$ output by a quantum algorithm **ALG** is in fact the correct matrix-vector product Mv , a task that classically requires $\Omega(n^2)$ queries. This will play an important role in both amplifying the success probability of our local correction lemma (see [Section 2.7](#)) and in the other quantum algorithms in the toolkit we develop.

Since it is crucial for our reduction to use such a verification procedure as a unitary subroutine in other quantum procedures, we unfortunately cannot apply existing verification algorithms from the literature. Furthermore, since the algorithm **ALG** only succeeds *on average* with low probability, different input vectors have significantly different success probabilities. To address this, we give a procedure that flags all of the computational

basis states in a superposition as either right or wrong in a way that respects the success distribution of **ALG**, and then boost the amplitude of the solutions.

However, since the success probabilities of different inputs have high variance, choosing a fixed number of iterations of amplitude amplification causes problems of over-shooting and under-shooting. Instead, we apply a delicate argument involving fixed-point amplitude amplification, which converges monotonically towards the flagged state. Towards this end, we identify that the flagging operation above has the form of a *block encoding*.² Having such an object enables us to use powerful techniques from the repertoire of quantum singular value transformations.

Implementing the strategy above, we construct the following quantum procedure for verifying matrix-vector products, which works *in superposition* and marks an ancillary qubit attached to the output state of **ALG** whenever the state has non-zero overlap with the correct matrix-vector product $|Mv\rangle$. See [Section 5.3.1](#) for details. In the equations below, $|w(\cdot)\rangle$ denotes the additional work registers that the algorithms may use.

Lemma 2.8.1 (Informally stated; see [Lemma 5.3.2](#)). *Given access to a unitary oracle for a matrix $M \in \mathbb{F}^{n \times n}$, and a quantum algorithm **ALG** that takes as input a vector $v \in \mathbb{F}^n$ and produces as output a state that consists of a superposition over vectors in $\mathbb{F}^n$,*

$$\mathbf{ALG} |v, 0\rangle = \sum_{z \in \mathbb{F}^n} \gamma_z^v |v\rangle |z\rangle |w(z, v)\rangle ,$$

*there is a quantum algorithm $\mathbf{ALG}_{\text{verified}}$ that annotates the output state of **ALG** with a flag marking the vector Mv as correct, and marking all other vectors $z \neq Mv$ as incorrect with high probability, i.e.*

$$\mathbf{ALG}_{\text{verified}} |v, 0\rangle \approx \gamma_*^v |v\rangle |Mv\rangle |w(Mv, v)\rangle |1\rangle^{\text{flag}} + \left(\sum_{z \neq Mv} \gamma_z^v |v\rangle |z\rangle |w(z, v)\rangle \right) |0\rangle^{\text{flag}} .$$

$\mathbf{ALG}_{\text{verified}}$ makes $O(1)$ uses of **ALG**, $O(n^{3/2})$ queries to U_M and $U_M^\dagger$, $\tilde{O}(n)$ ancillary qubits, and $\tilde{O}(n^{3/2})$ additional one-qubit and two-qubit gates.

Noisy quantum oracles approximating the indicator function 1_X . A key ingredient in our reductions is the indicator function or membership oracle 1_X for the subset of

²A unitary U with the form $\begin{pmatrix} H & \cdot \\ \cdot & \cdot \end{pmatrix}$, which encodes another (subnormalised) matrix H in its upper left block.

good inputs exceeding probability threshold τ , given by

$$X_\tau = \{v \in \mathbb{F}^n : \Pr[\text{ALG}^M(v) = Mv] \geq \tau\}.$$

Here the notation ALG^M emphasises the fact that ALG has query access to the matrix M . In constructing this indicator function we are faced with yet another challenge: while fixed-point amplitude amplification is able to boost the probability of outputting the correct success/failure flag, it is insufficient to perform the type of thresholding operation that is required to implement the indicator function. The key difficulty is the fact that we need to *simultaneously* mark inputs inside set X_τ with the flag one, and those outside X_τ with the flag zero (with high probability).

To overcome this issue, we note that $\text{ALG}_{\text{verified}}$ is also a block encoding (of another matrix), and by combining it with the heavier machinery of *quantum singular value threshold projection*, we are able to obtain a noisy quantum oracle for a polynomial approximation of the indicator function on the set X_τ of good inputs to ALG . In [Section 5.3.2](#), we show how the technique of quantum singular value transformations can be used to construct a singular value threshold projection built on $\text{ALG}_{\text{verified}}$, which acts as a noisy indicator function on the set of inputs where ALG succeeds with high probability.

Sampling vectors from the set of good inputs. Using the ability to approximately flag correct and incorrect answers in superposition, we can synthesize a uniform superposition over the set X_τ that consists of all inputs on which the average-case algorithm computes correctly with probability at least τ . Hence by preparing this state and measuring it, we obtain the following quantum speedup for sampling from X_τ . This will allow us to boost the performance of our quantum local correction lemma (see [Section 2.7](#)), that needs to sample elements from X_τ for a value of $\tau = \Theta(\alpha)$ in order to correct vectors in the Bogolyubov subspace.

Lemma 2.8.2 (Informally stated; see [Section 5.3.3](#)). *Given an average-case quantum algorithm ALG with success rate α , there is a procedure Q_{samp} which with high probability produces a vector v on which ALG succeeds with probability at least $\alpha/2$. Q_{samp} uses ALG $O(\frac{1}{\alpha})$ times, in addition to $\tilde{O}(n^{3/2})$ additional one-qubit and two-qubit gates.*

Learning Bogolyubov subspaces from noisy quantum oracles. In order to locally correct vectors that are outside of the Bogolyubov subspace V , we shall need to learn an approximation of V , so that we can shift arbitrary vectors into V via sparse vectors (see [Section 2.7](#)).

Standard quantum procedures for sampling characters according to the probabilities defined by the Fourier spectrum of a function f use a unitary oracle for f in superposition, and are built on the Bernstein-Vazirani algorithm. However, in our setting we need to list-decode the heavy Fourier coefficients of a probabilistic implementation of an indicator function that we obtain from the linear verification procedure in [Lemma 2.8.1](#). Hence, we face the challenge of learning the heavy Fourier characters from a noisy implementation of the function. To this end, we generalise the techniques used to prove Adcock and Cleve’s quantum Goldreich-Levin lemma [[AC02](#)], and obtain the following quantum procedure.

Lemma 2.8.3 (Informally stated; see [Lemma 5.3.5](#)). *For a function $f : \mathbb{F}^n \rightarrow \mathbb{F}_2$, given a noisy quantum oracle U_f acting on m qubits such that for every input x , measuring the output qubit of U_f produces $f(x)$ with probability at least $1 - \epsilon$, there is a quantum procedure C_{GL} which produces outputs $y \in \mathbb{F}^n$ with probability p_y that satisfy $|p_y - |\hat{f}(y)|^2| \leq 4\epsilon$. C_{GL} uses U_f and $U_f^\dagger$ once, in addition to $O(n \log |\mathbb{F}|)$ additional one-qubit and two-qubit gates.*

We stress that due to the noisy implementation, we do not obtain the heavy Fourier characters of the actual indicator function of the set of good inputs X , but rather an approximation of these Fourier coefficients. Fortunately, this guarantee suffices for the noise-robust Bogolyubov lemma that we presented in [Section 2.7](#).

In fact, [Lemma 2.8.3](#) is not the last link in the chain: a further complication arises from the fact that any approximation of the indicator function 1_X can only work well in a subset of X and a subset of its complement, necessarily oscillating in a “wasteland” region corresponding to vectors on which the success probability of **ALG** lies in some range $(\tau - \delta, \tau + \delta)$.

The polynomial approximation happens at the level of a real-valued function (the success probability of **ALG** on input vectors) and the corresponding wasteland slice is apparently small, namely, an interval of length 2δ . However this hides the fact that the actual number of input vectors falling into this intermediate set can be alarmingly large, with density as high as $\frac{1-\alpha}{1-\alpha/2}$.

We show that in spite of this difficulty, a combination of careful error analysis of our quantum Bogolyubov subspace learning technique and, as we discuss in [Section 2.9](#), a carefully chosen random selection of the threshold τ , allows us to efficiently learn the Bogolyubov subspace.

2.9 Quantum worst-case to average-case reductions

We first present a high-level overview of our reduction for the Matrix-Vector Multiplication problem in a simplified setting, then sketch how to extend the proof to obtain [Theorem 5](#) and [Theorem 6](#).

Recall that we start with an *average-case* quantum algorithm ALG that is correct with probability α on a randomly chosen input, i.e.,

$$\Pr_{\substack{M,v \\ \text{ALG}}}[\text{ALG}^{M,v} = Mv] = \mathbb{E}_{\substack{M \in \mathbb{F}^{n \times n} \\ v \in \mathbb{F}^n}} \left[\|\Pi_{Mv} \text{ALG}^{M,v} |0\rangle\|^2 \right] \geq \alpha .$$

Our goal is to boost the success rate of the algorithm such that we obtain a *worst-case* quantum algorithm that succeeds with high probability *on all* inputs.

In order to describe the reduction the following notation will be convenient. For each $v \in \mathbb{F}^n$ let $p_v = \Pr_{\text{ALG}}[\text{ALG}(v) = Mv]$ be the probability that ALG correctly computes the output on input v , where the probability is only over the measurement of ALG .

First, we define threshold sets as follows.

$$X_\tau = \{v \in \mathbb{F}^n : p_v > \tau\} .$$

Since $\Pr_{v \in \mathbb{F}^n}[\text{ALG}(v) = Mv] \geq \alpha$, it implies that $\mathbb{E}_{v \in \mathbb{F}^n}[p_v] \geq \alpha$, and hence, by Markov's inequality, for $\tau \leq \alpha/2$ it holds that $\frac{|X_\tau|}{|\mathbb{F}^n|} \geq \alpha/2$.

Ideally, we could construct the worst-case quantum algorithm ALG' as follows. We first learn all of the significant Fourier coefficients of $X_{\alpha/2}$ using the quantum learning procedure in [Lemma 2.8.3](#). Then, by the quantum local correction lemma (i.e., [Lemma 2.7.2](#), which in turn, relies on the robust Bogolyubov lemma), these Fourier coefficients can be used to compute a decomposition of any input $v \in \mathbb{F}^n$ as

$$v = s + x_1 + x_2 + x_3 + x_4, \text{ where } x_1, x_2, x_3, x_4 \in X_{\alpha/2} ,$$

and $s \in \mathbb{F}^n$ is a sparse vector that has only $O(1)$ -non zero entries, which allows us to shift arbitrary inputs into the Bogolyubov subspace.

More specifically, for each input $v \in \mathbb{F}^n$, we can use the Fourier coefficients of $X_{\alpha/2}$ in order to compute the sparse vector s . Then, we sample x_1, x_2, x_3 from $X_{\alpha/2}$, and set $x_4 = v - x_1 - x_2 - x_3 - s$. By the quantum local correction lemma ([Lemma 2.7.2](#)) we have

that $\Pr[x_4 \in X_{\alpha/2}] \geq \text{poly}(\alpha)$, and hence set the algorithm ALG' to compute

$$Ms + \text{ALG}^M(x_1) + \text{ALG}^M(x_2) + \text{ALG}^M(x_3) + \text{ALG}^M(x_4) .$$

Note that computing Ms can be done in $O(n)$ time, where $O(\cdot)$ hides the sparsity of s , and Mx_i can be computed correctly using ALG for each $i \in \{1, 2, 3, 4\}$ with probability $\text{poly}(\alpha)$. In total, the above procedure outputs the correct answer with probability at least $\text{poly}(\alpha)$. By repeating the procedure $\text{poly}(1/\alpha)$ times and verifying the result at each time, we obtain a reduction that succeeds with high probability.

However, the actual reduction is more subtle. The main reason for this is that the lemma we use to learn the Bogolyubov subspace ([Lemma 2.8.3](#)) cannot sample from the Fourier spectrum of the set $X_{\alpha/2}$ exactly, as computing p_v exactly for a particular v requires a large number of samples, and so, we cannot apply [Lemma 2.7.2](#) directly on the set $X_{\alpha/2}$.

Instead, we choose a parameter $\tau \in [\alpha/4, \alpha/2]$, and consider the sets X_τ and $X_{\tau'}$, where $\tau' = \tau - O(\alpha^{3/2})$. See [Corollary 5.4.4](#) for details on how the random threshold τ is chosen. Then, we apply [Lemma 2.7.2](#) with respect to *some* set X^* such that with high probability $X_\tau \subseteq X^* \subseteq X_{\tau'}$. Note that we do not have any structural guarantee about X^* containing a particular vector in $X_{\tau'} \setminus X_\tau$.

We use [Lemma 2.8.3](#) in order to obtain all significant Fourier coefficients of the indicator function of X^* , and we use them to approximate the Fourier coefficients of the indicator function of X_τ .

By the discussion above, we may assume that we know a good approximation of all significant Fourier coefficients of X_τ . See [Section 5.4.1](#) for more details on this. The rest of the reduction follows the same plan as described above.

Extending the average-case over both matrices and vectors. So far, we obtained a worst-case to average-case reduction where the average-case condition only refers to the vectors in the Matrix-Vector Multiplication problem. To extend the average case to both vectors and matrices, we need an additional layer of matrix local correction. To this end, we use the technique of shifting the given matrix M by a random matrix R , which with probability $\Omega(\alpha)$ shifts the input to the set of good matrices, where matrix-vector multiplications are computed correctly for an $\Omega(\alpha)$ -fraction of vectors. See [Section 5.2.2](#) for details.

Quantum algorithms for linear problems. Throughout the argument that we outlined above, we employed a proof strategy that only invokes gate-efficient quantum algo-

rithms (i.e., with gate complexities that are larger than the query complexity by at most a polylog factor). We do this precisely due to our interest in uniform quantum algorithms for linear problems: by a careful instantiation of families of quantum circuits as quantum query algorithms that match the setting of [Theorem 5.2.3](#), we obtain in [Theorem 5.2.2](#) quantum worst-case to average-case reductions for all linear problems. See [Section 5.2.1](#) for details.

Chapter 3

Additive Combinatorics Toolbox

In this chapter, we provide a collection of tools for locally correcting vectors using techniques from additive combinatorics. The toolbox will play a key technical role in all of our worst-case to average-case reductions.

Throughout this chapter, we fix a finite field $\mathbb{F}$. For simplicity, we set $\mathbb{F} = \mathbb{F}_p$ for a prime number p . However, we remark that the following results hold for any finite field, with only a negligible change in parameters (see discussions in relevant places below). Recall that the sumset of a set X is defined as $X + X = \{x_1 + x_2 : x_1, x_2 \in X\}$, and, similarly, $t \cdot X = \{x_1 + \dots + x_t : x_1, \dots, x_t \in X\}$ for an integer $t \geq 1$.

Let $X \subset \mathbb{F}^n$ be a subset of size $|X| = \alpha \cdot |\mathbb{F}|^n$. As we outlined in [Chapter 2](#), our goal is to decompose any vector $y \in \mathbb{F}^n$ as a linear combination of the form

$$y = x_1 + x_2 - (x_3 + x_4) + s ,$$

where $x_1, x_2, x_3, x_4 \in X$, and $s \in \mathbb{F}^n$ is a sparse vector.

Towards this end, we will need additive combinatorics lemmas that will allow us to find a large subspace $V \subseteq 2X - 2X$, so that any vector $v \in V$ can be written as $v = x_1 + x_2 - (x_3 + x_4)$. Crucially, we will show that we can efficiently sample such a decomposition and verify membership in the subspace V .

3.1 Probabilistic and quasi-polynomial Bogolyubov-Ruzsa lemmas

A natural starting point for obtaining a subspace as discussed above is via *Bogolyubov's lemma*, which states that for any subset $X \subseteq \mathbb{F}_2^n$ of density $|X|/2^n \geq \alpha$, there exists a subspace $V \subseteq 4X$ of dimension at least $n - \alpha^{-2}$. However, in addition to minor issues such as being restricted to the field $\mathbb{F}_2$, there are some fundamental problems with using Bogolyubov's lemma for local correction. Most importantly for our application is that while Bogolyubov's lemma asserts that *there exists* a decomposition of each input into a sum of four elements in X , it does not tell us how to obtain this decomposition.

Hence, we further show that each vector $v \in V$ has many “representations” of a sum of 4 elements from X . This way, for any $v \in V$ we can efficiently sample a representation $v = x_1 + x_2 + x_3 + x_4$, where each $x_i \in X$. We refer to this statement as the probabilistic Bogolyubov lemma. To make the following discussion precise, we shall need the following notation.

Given a set $X \subseteq \mathbb{F}^n$, we denote by $1_X: \mathbb{F}^n \rightarrow \{0, 1\}$ the indicator function of the set X . The convolution of two boolean functions f and g we denote by $(f * g)(x) = \mathbb{E}_y[f(y)g(x-y)]$. The Fourier expansion of a function $f: \mathbb{F}^n \rightarrow \mathbb{C}$ is given by $f(x) = \sum_{r \in \mathbb{F}^n} \hat{f}(r) \cdot \chi_r(x)$, where the Fourier coefficients of f are defined as $\hat{f}(r) = \langle f, \chi_r \rangle = \mathbb{E}_x[f(x) \cdot \overline{\chi_r(x)}]$, with $\chi_r(v) = \omega^{\langle v, r \rangle}$ and $\omega = e^{\frac{2\pi i}{p}}$ is the p th root of unity. In particular for convolution of two functions we have $(f * g)(x) = \sum_r \hat{f}(r) \hat{g}(r) \chi_r(x)$.

Lemma 3.1.1 (Probabilistic Bogolyubov lemma). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, and let $X \subseteq \mathbb{F}^n$ be a set of size $|X| = \alpha \cdot |\mathbb{F}|^n$ for some $\alpha \in (0, 1]$. Let $R = \{r \in \mathbb{F}^n \setminus \{0\} : |\hat{1}_X(r)| \geq \alpha^{3/2}\}$, and let $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$ be a linear subspace defined by R . Then $|R| \leq 1/\alpha^2$, and for all $v \in V$ it holds that*

$$\Pr_{x_1, x_2, x_3} [x_1, x_2, x_3, v - x_1 - x_2 - x_3 \in X] \geq \alpha^5.$$

Proof. Note first that by Parseval's identity we have $\alpha = \langle 1_X, 1_X \rangle = \|1_X\|_2^2 = \sum_r |\hat{1}_X(r)|^2$. In particular, for $R = \{r \in \mathbb{F}^n \setminus \{0\} : |\hat{1}_X(r)|^2 > \alpha^3\}$ we have $|R| \leq \frac{\alpha}{\alpha^3} = \frac{1}{\alpha^2}$. Furthermore, we have

$$\sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^4 \leq \alpha^3 \cdot \sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^2 \leq \alpha^3(\alpha - \alpha^2) \leq \alpha^4 - \alpha^5,$$

where the second inequality uses that $\sum_r |\hat{1}_X(r)|^2 = \alpha$, and $|\hat{1}_X(0)|^2 = \alpha^2$.

Noting that for every $v \in V$ we have $\chi_r(v) = \omega^{\langle v, r \rangle} = \omega^0 = 1$ for all $r \in R$, it follows that

$$\begin{aligned}
\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2, x_3, v - x_1 - x_2 - x_3 \in X] &= (1_X * 1_X * 1_X * 1_X)(v) \\
&= \sum_{r \in \mathbb{F}^n} (\hat{1}_X(r))^4 \chi_r(v) \\
&= |\hat{1}_X(0)|^4 \chi_0(v) + \sum_{r \in R} |\hat{1}_X(r)|^4 \chi_r(v) \\
&\quad + \sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^4 \chi_r(v) \\
&\geq \alpha^4 + |R| \cdot \alpha^6 - (\alpha^4 - \alpha^5) \\
&\geq \alpha^5,
\end{aligned}$$

as required. □

In fact, the foregoing lemma suffices for our application for worst-case to average-case reductions where the success rate α is *a constant*. However, to also allow for success rates that tend to zero, we shall need a much stronger statement of the form of the quasi-polynomial Bogolyubov-Ruzsa lemma, due to Sanders [San12] (see also [Lov15, BRTW14]), which admits an exponentially better dependency on α , albeit without the efficient sampling property.

Lemma 3.1.2 (Quasi-polynomial Bogolyubov-Ruzsa lemma [San12]). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, and let $X \subseteq \mathbb{F}^n$ be a set of size $\alpha \cdot |\mathbb{F}|^n$ for some $\alpha \in (0, 1]$. There exists a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) \geq n - O(\log^4(1/\alpha))$ such that $V \subseteq 2X - 2X$.*

The caveat, however, is that while in Lemma 3.1.2 the codimension of V is only *polylogarithmic* in $1/\alpha$ (as opposed to polynomial, as in the probabilistic Bogolyubov lemma), it only guarantees that for each $v \in V$ there exist $x_1, x_2, x_3, x_4 \in X$ such that $x_1 + x_2 + x_3 + x_4 = v$.

Hence, we further show that each vector $v \in V$ has many “representations” in $2X - 2X$. In particular, for any $v \in V$ we can efficiently sample a representation $v = x_1 + x_2 - x_3 - x_4$, where each $x_i \in X$. We are grateful to Tom Sanders for providing us with a modification of his proof that admits a probabilistic version of the quasi-polynomial Bogolyubov-Ruzsa lemma. Furthermore, we rely on the Goldreich-Levin algorithm and

the techniques in [BRTW14] to obtain an efficient algorithm for verifying membership in the implied subspace. This yields the main technical tool that underlies our local correction paradigm.

Lemma 3.1.3 (Probabilistic quasi-polynomial Bogolyubov-Ruzsa lemma). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, and let $A \subseteq \mathbb{F}^n$ be a set of size $|A| = \alpha \cdot |\mathbb{F}|^n$, for some $\alpha \in (0, 1]$. Then, there exists a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) \geq n - O(\log^4(1/\alpha))$ such that for all $v \in V$ it holds that*

$$\Pr_{a_1, a_2, a_3 \in \mathbb{F}^n} [a_1, a_2 \in A, a_3, a_4 \in -A] \geq \Omega(\alpha^5) ,$$

where $a_4 = v - a_1 - a_2 - a_3$. Furthermore, given a query access to the set A , there is an algorithm that runs in time $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ and with probability $1 - \delta$ computes a set of vectors $R \subseteq \mathbb{F}^n$ such that $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$.

Before presenting the proof this lemma, let us establish some notation. For a set $A \subseteq \mathbb{F}^n$, we denote by $1_A: \mathbb{F}^n \rightarrow \{0, 1\}$ the indicator function of A , where $1_A(x) = 1$ if $x \in A$, and $1_A(x) = 0$ otherwise. In particular $\mathbb{E}_{x \in \mathbb{F}^n} [1_A(x)] = \frac{|A|}{|\mathbb{F}|^n}$. We also let $\varphi_A: \mathbb{F}^n \rightarrow \mathbb{R}$ be the normalization of 1_A defined as $\varphi_A(x) = 1_A(x) \cdot \frac{|\mathbb{F}|^n}{|A|}$ so that $\mathbb{E}_{x \in \mathbb{F}^n} [\varphi_A(x)] = 1$. When the set A is a singleton $A = \{a\}$, we will write $\varphi_a = \varphi_{\{a\}}$.

It is easy to verify that for a set A and a function f the convolution $\varphi_A * f$ is given by $\varphi_A * f(x) = \mathbb{E}_{a \in A} [f(x - a)]$. In particular, for $a \in \mathbb{F}^n$ we have $\varphi_a * f(x) = f(x - a)$.

As a starting point, we define the set $D = \{d \in \mathbb{F}^n : 1_A * 1_{-A}(d) \geq \delta\}$ for a parameter $\delta = \alpha^2/20$. That is, D is the set of all *popular differences* of two elements of A . In other words, D consists of all $d \in \mathbb{F}^n$ such that there are $\delta|\mathbb{F}|^n$ pairs $(a, a') \in A^2$ satisfying $d = a - a'$, i.e., $\Pr_{a \in \mathbb{F}^n, a' = d-a} [a \in A, a' \in -A] \geq \delta$. Note first that

$$\langle 1_{A-A}, \varphi_A * \varphi_{-A} \rangle = \mathbb{E}_{x, y \in \mathbb{F}^n} [1_{A-A}(x - y) \varphi_A(x) \varphi_{-A}(-y)] = \mathbb{E}_{x, y \in A} [1_{A-A}(x - y)] = 1.$$

Next, we observe that D approximates $A - A$, in the sense that $\langle 1_D, \varphi_A * \varphi_{-A} \rangle \geq 1 - \frac{\delta}{\alpha^2} =$

0.95. Indeed, using the fact that $D \subseteq A - A$, we have

$$\begin{aligned}
\langle 1_D, \varphi_A * \varphi_{-A} \rangle &= \langle 1_{A-A}, \varphi_A * \varphi_{-A} \rangle - \langle 1_{\mathbb{F}^n \setminus D}, \varphi_A * \varphi_{-A} \rangle \\
&= 1 - \frac{1}{\alpha^2} \langle 1_{\mathbb{F}^n \setminus D}, 1_A * 1_{-A} \rangle \\
&= 1 - \frac{1}{\alpha^2} \cdot \Pr_{d, a \in \mathbb{F}^n} [a \in A, d - a \in -A | d \notin D] \cdot \Pr_{d \in \mathbb{F}^n} [d \notin D] \\
&\geq 1 - \frac{\delta}{\alpha^2}.
\end{aligned} \tag{3.1}$$

We remark that this is one of the main differences in our proof compared to the original proof of Sanders, who only relied on the fact that $\langle 1_{A-A}, \varphi_A * \varphi_{-A} \rangle = 1$.

Our proof of [Lemma 3.1.3](#) follows the approach of Sanders, with several modifications. We follow the exposition of Lovett [[Lov15](#)]. The proof consists of the following two parts.

Lemma 3.1.4. *Let $A \subseteq \mathbb{F}^n$ be a set of size $|A| = \alpha |\mathbb{F}|^n$. Set $t = O(\log(1/\alpha))$. There exists a set $X \subseteq \mathbb{F}^n$ of size $|X| \geq \alpha^{O(\log^3(1/\alpha))} |\mathbb{F}|^n$ such that for all $x_1, x_2, \dots, x_t \in X$ it holds that*

$$\Pr_{a_1, a_2 \in A} [a_1 - a_2 - \sum_{i=1}^t x_i \in D] \geq 0.9. \tag{3.2}$$

Given the set X from [Lemma 3.1.4](#) we use a standard Fourier-analytic argument to define a large subspace V such that $|D \cap V'| \geq 0.8|V|$ where V' is some coset of V . In fact, we show that there are many such cosets. Formally, we prove the following lemma.

Lemma 3.1.5. *Let $A \subseteq \mathbb{F}^n$ be a set of size $|A| = \alpha |\mathbb{F}|^n$. Then, there exists a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) = n - O(\log^4(1/\alpha))$ and a vector $b \in \mathbb{F}^n$ such that if we sample a uniformly random $a \in A$ and $v \in V$ then*

$$\Pr_{a \in A, v \in V} [v + a + b \in D] \geq 0.85. \tag{3.3}$$

Remark 3.1.6. [Lemma 3.1.4](#) corresponds to Lemma 4.2 in [[Lov15](#)]. The only difference is that we claim that the sum belongs to D with high probability, while in [[Lov15](#)] the sum belongs to $A - A$. Note that the two statements are indeed close to each other by [Eq. \(3.1\)](#).

[Lemma 3.1.5](#) roughly corresponds to the conclusion of the section “A Fourier-analytic argument” in [[Lov15](#)].

We show next how to conclude the proof of [Lemma 3.1.3](#) from [Lemma 3.1.5](#). Indeed, by [Eq. \(3.3\)](#) it follows that $\mathbb{E}_{a \in A} [\Pr_{v \in V} [v + a + b \in D]] \geq 0.85$, and hence for at least $0.05|A|$

many $a \in A$ it holds that $\Pr_{v \in V}[v + a + b \in D] \geq 0.8$. In particular, for $C = \{c \in \mathbb{F}^n : |D \cap (V + c)| \geq 0.8|V|\}$ we have $|C| \geq 0.05\alpha|\mathbb{F}|^n$.

Claim 3.1.7. *The set $C \subseteq \mathbb{F}^n$ is symmetric. Namely, if $c \in C$, then $-c \in C$.*

Proof. Let $c \in C$ and let $V + c$ be the corresponding coset of V . We claim that $-c \in C$, i.e., $|\{v \in V : \text{there are } \geq \delta|\mathbb{F}|^n \text{ pairs } (a_1, a_2) \in A^2 \text{ such that } a_1 - a_2 = v - c\}| \geq 0.8|V|$.

For $c \in C$ let P_c be as follows.

$$P_c = \{v \in V : \text{there are } \geq \delta|\mathbb{F}|^n \text{ pairs } (a_1, a_2) \in A^2 \text{ such that } a_1 - a_2 = -v + c\} .$$

By definition of C , we have $|P_c| \geq 0.8|V|$.

To see that $-c \in C$ take any $v \in P_c$, and note that $a_1 - a_2 = -v + c$ if and only if $a_2 - a_1 = v - c$. Therefore, for each $v \in P_c$ there are $\geq \delta|\mathbb{F}|^n$ pairs $(a_1, a_2) \in A^2$ such that $a_2 - a_1 = v - c$, and thus $-c \in C$, as required. $\square$

Let us choose a unique representative c^* for each coset $V + c$ of V such that

$$|D \cap (V + c)| \geq 0.8|V| ,$$

and let $C^* = \{c^* \text{ is the representative of } V + c : |D \cap (V + c)| \geq 0.8|V|\}$. And furthermore, let us assume without loss of generality that C^* is symmetric, i.e. $c^* \in C^*$ implies that $-c^* \in C^*$. Then, the union of all these coset covers is at least 0.05α fraction of $\mathbb{F}^n$, i.e.,

$$|\cup_{c^* \in C^*} (V + c^*)| \geq 0.05\alpha|\mathbb{F}|^n . \quad (3.4)$$

We are now ready to show that

$$\Pr_{\substack{a_1, a_2, a_3 \in \mathbb{F}^n \\ a_4 = v - a_1 - a_2 - a_3}} [a_1, a_2 \in A, a_3, a_4 \in -A] \geq \Omega(\alpha^5) .$$

Proof of Lemma 3.1.3. Fix $v \in V$. Since $|D \cap (V + c^*)| \geq 0.8|V|$ for every coset $V + c^*$ such that $c^* \in C^*$, it follows by the symmetry of C^* that for every $c^* \in C^*$ we have at least $0.1 \cdot |V|$ pairs $(u + c^*, v - u - c^*) \in D^2$ such that $u \in V$. Therefore, by Eq. (3.4) for every $v \in V$ there are at least $0.1 \times 0.05\alpha|\mathbb{F}|^n$ different pairs $(u + c^*, v - u - c^*)$ such that both $u + c^* \in D \cap (V + c^*)$ and $v - u - c^* \in D \cap (V - c^*)$.

Letting $d_1 = u + c^*$, so far we got that for every $v \in V$ we have $\Pr_{d_1 \in \mathbb{F}^n, d_2 = v - d_1} [d_1, d_2 \in D] \geq \Omega(\alpha)$.

Recall that by the definition of D , every $d_1 \in D$ is a popular difference of elements of A , i.e. $\Pr_{\substack{a_1 \in \mathbb{F}^n \\ a_3 = d_1 - a_1}} [a_1 \in A, a_3 \in -A] \geq \delta$. Similarly, for $d_2 \in D$ we have $\Pr_{\substack{a_2 \in \mathbb{F}^n \\ a_4 = d_2 - a_2}} [a_2 \in A, a_4 \in -A] \geq \delta$. This implies that

$$\Pr_{\substack{a_1, a_2, a_3 \in \mathbb{F}^n \\ a_4 = a_1 - a_2 - a_3}} [a_1, a_2 \in A, a_3, a_4 \in -A] \geq \Omega(\alpha \cdot \delta^2) ,$$

as required. $\square$

Finally, we turn to proving each of the two steps stated in [Lemmas 3.1.4](#) and [3.1.5](#).

Proof of Lemma 3.1.4 The proof starts with the following lemma of Croot and Sisask [\[CS10\]](#).

Lemma 3.1.8 (Croot-Sisask [\[CS10, Proposition 3.3\]](#)). *Let $A, B \subseteq \mathbb{F}^n$ be two sets, and let $\varepsilon \in (0, 1)$ and $p \geq 1$. Let $\alpha = \frac{|A|}{|\mathbb{F}|^n} \in (0, 1)$. Then, there exists a set $X \subseteq \mathbb{F}^n$ of size $|X| \geq (\alpha/2)^{O(p/\varepsilon^2)} |\mathbb{F}|^n$ such that for all $x \in X$ it holds that*

$$\|\varphi_x * \varphi_A * 1_B - \varphi_A * 1_B\|_p \leq \varepsilon .$$

Let $p = \log_2(1/\alpha)$, $t = \Theta(\log(1/\alpha))$, and $\varepsilon = (1/40t)$. By applying [Lemma 3.1.8](#) we obtain a set $X \subseteq \mathbb{F}^n$ of size $|X| \geq (\alpha/2)^{O(p/\varepsilon^2)} \geq \alpha^{O(\log^3(1/\alpha))} |\mathbb{F}|^n$. We show below that X satisfies [Eq. \(3.2\)](#).

Fix $x_1, \dots, x_t \in X$, and let $s = \sum_{i=1}^t x_i$. Note first that by setting $B = D$ in [Lemma 3.1.8](#) and combining it with triangle inequality we get that

$$\|\varphi_s * \varphi_A * 1_D - \varphi_A * 1_D\|_p \leq t \cdot \varepsilon \leq 1/40 ,$$

where the last inequality is by the choice of $\varepsilon = 1/40t$. Let $q = p/(p-1)$, then by the choice of $p = \log_2(1/\alpha)$ we have

$$\|\varphi_A\|_q = \left(\alpha \cdot \frac{1}{\alpha^q} \right)^{1/q} = \left(\frac{1}{\alpha} \right)^{1/p} \leq 2 .$$

Then, by applying Hölder's inequality with $q = p/(p-1)$ we get

$$|\langle \varphi_s * \varphi_A * 1_D - \varphi_A * 1_D, \varphi_A \rangle| \leq \|\varphi_s * \varphi_A * 1_D - \varphi_A * 1_D\|_p \cdot \|\varphi_A\|_q \leq 1/20 .$$

By combining the above inequality with [Eq. \(3.1\)](#) we get

$$\begin{aligned}
\Pr_{a_1, a_2 \in A} [a_1 - a_2 - s \in D] &= \langle \varphi_s * \varphi_A * 1_D, \varphi_A \rangle \\
&= \langle \varphi_A * 1_D, \varphi_A \rangle - \langle \varphi_A * 1_D - \varphi_s * \varphi_A * 1_D, \varphi_A \rangle \\
&= \langle 1_D, \varphi_A * \varphi_{-A} \rangle - \langle \varphi_A * 1_D - \varphi_s * \varphi_A * 1_D, \varphi_A \rangle \\
&\geq (1 - \delta/\alpha^2) - 1/20 \geq 0.9 \ ,
\end{aligned}$$

as required.

Proof of Lemma 3.1.5 The proof of this step is essentially Section 5 of [\[Lov15\]](#). The only (minor) difference is that we work over $\mathbb{F}_p$ and not over $\mathbb{F}_2$.

Given the set $X \subseteq \mathbb{F}^n$ from [Lemma 3.1.4](#) of size $|X| \geq \alpha^{O(\log^3(1/\alpha))} |\mathbb{F}|^n$, we define $\text{Spec}_\gamma(X) = \{r \in \mathbb{F}^n : |\widehat{\varphi}_X(r)| \geq \gamma\}$. Since $\sum_{q \in \mathbb{F}^n} |\widehat{\varphi}_X(r)|^2 = \mathbb{E}_z[\varphi_X(z)^2] = 1/\alpha$, it follows that $|\text{Spec}_\gamma(X)| \leq \frac{1}{\alpha\gamma^2}$. Chang's lemma provides a non-trivial bound on the dimension of the subspace containing $\text{Spec}_\gamma(X)$.

Lemma 3.1.9 (Chang [\[Cha02\]](#)). *Let $X \subseteq \mathbb{F}^n$ of size $|X| = \beta \cdot |\mathbb{F}|^n$, and let $\gamma > 0$. Then*

$$\dim(\text{Spec}_\gamma(X)) \leq O\left(\frac{\log(1/\beta)}{\gamma^2}\right) \ .$$

Define the subspace $V = \text{Spec}_{1/2}(X)^\perp = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in \text{Spec}_{1/2}(X)\}$. [Lemma 3.1.9](#) implies that $\dim(V) \geq n - O(\log^4(|\mathbb{F}|^n/|X|)) \geq n - O(\log^4(1/\alpha))$.

We now show that V indeed satisfies the guarantee of [Lemma 3.1.5](#). For $x_1, x_2, \dots, x_t \in X$ let $s = \sum_{i=1}^t x_i$ as in the previous lemma. By [Lemma 3.1.4](#) for all $x_1, x_2, \dots, x_t \in X$ we have

$$\Pr_{a_1, a_2 \in A} [a_1 - a_2 - s \in D] \geq 0.9 \ .$$

Next, we are comparing this probability to the following.

$$\Pr_{\substack{v \in V \\ a_1, a_2 \in A}} [v + a_1 - a_2 - s \in D] \ . \tag{3.5}$$

We claim that if we sample $v \in V$, $a_1, a_2 \in A$, and $x_1, \dots, x_t \in X$ uniformly at random (and let $s = \sum_{i=1}^t x_i$), then the two quantities are close to each other. We prove this by rewriting the two probabilities using the Fourier expansion. Note that the Fourier

coefficients of φ_V are simple to describe since V is a linear subspace, and they are equal to $\widehat{\varphi_V}(r) = 1$ if $r \in V^\perp$ and $\widehat{\varphi_V}(r) = 0$ otherwise. Therefore,

$$\begin{aligned}
\Pr_{\substack{v \in V \\ x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [v + a_1 - a_2 - s \in D] &= \left\langle \varphi_V * \varphi_A * \varphi_{-A} * \varphi_{-X}^{(t)}, \mathbf{1}_D \right\rangle \\
&= \sum_{r \in \mathbb{F}^n} \widehat{\varphi_V}(r) \cdot \widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r) \cdot \widehat{\varphi_X^t}(-r) \cdot \overline{\widehat{\mathbf{1}_D}(r)} \\
&= \sum_{r \in V^\perp} \widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r) \cdot \widehat{\varphi_X^t}(-r) \cdot \overline{\widehat{\mathbf{1}_D}(r)} .
\end{aligned}$$

On the other hand

$$\Pr_{\substack{x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [a_1 - a_2 - s \in D] = \left\langle \varphi_A * \varphi_{-A} * \varphi_{-X}^{(t)}, \mathbf{1} \right\rangle = \sum_{r \in \mathbb{F}^n} \widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r) \cdot \widehat{\varphi_X^t}(-r) \cdot \overline{\widehat{\mathbf{1}_D}(r)} .$$

This implies

$$\begin{aligned}
&\left| \Pr_{\substack{v \in V \\ x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [v + a_1 - a_2 - s \in D] - \Pr_{\substack{x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [a_1 - a_2 - s \in D] \right| \\
&= \sum_{r \notin V^\perp} \widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r) \cdot \widehat{\varphi_X^t}(r) \cdot \overline{\widehat{\mathbf{1}_D}(r)} \\
&\leq \sum_{r \notin V^\perp} \left| \widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r) \cdot 2^{-t} \cdot \overline{\widehat{\mathbf{1}_D}(r)} \right| \\
&\leq 2^{-t} \sum_{r \in \mathbb{F}^n} |\widehat{\varphi_A}(r) \cdot \widehat{\varphi_{-A}}(-r)| \\
&\quad \text{(By Cauchy-Schwarz)} \leq 2^{-t} \sum_{r \in \mathbb{F}^n} |\widehat{\varphi_A^2}(r)| \\
&= 2^{-t} \cdot \mathbb{E}_{x \in \mathbb{F}^n} [\varphi_A^2(x)] \\
&= \frac{1}{\alpha \cdot 2^t} < 0.05 ,
\end{aligned}$$

where the last inequality holds due to the choice of $t = O(\log(1/\alpha))$. Therefore,

$$\Pr_{\substack{v \in V \\ x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [v + a_1 - a_2 - s \in D] \geq \Pr_{\substack{x_1, \dots, x_t \in X \\ a_1, a_2 \in A}} [a_1 - a_2 - s \in D] - 0.05 \geq 0.85 .$$

Finally, we can fix $a_2 + s$ maximizing the probability, and let $b = -a_2 - s$ to conclude the proof of [Lemma 3.1.5](#).

3.2 Algorithmic construction of the subspace V

Given a set A we can construct V using the algorithm described in [\[BRTW14\]](#). Indeed, the only ingredients we need for our construction are the set X from [Lemma 3.1.4](#) and the subspace V guaranteed by [Lemma 3.1.5](#).

A straightforward inspection of the algorithm described in [\[BRTW14\]](#) gives the desired result. Informally, the algorithm works as follows: The set X is defined as the set of all $x \in X$ such that

$$\|\varphi_x * \varphi_A * 1_D - \varphi_A * 1_D\|_p \leq \varepsilon .$$

Note that given $x \in \mathbb{F}^n$ we can estimate the norm efficiently (up to a small error). This gives us a membership oracle to the set X .

Given such a query oracle, we can use Goldreich-Levin algorithm over $\mathbb{F}$ to compute $R = \text{Spec}_{1/2}(X)$ [\[Aka08\]](#), or more precisely its superset that is not too large, and using it we define the subspace $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$.

3.3 Local correction lemma

Using the probabilistic quasi-polynomial Bogolyubov-Ruzsa lemma (i.e., [Lemma 3.1.3](#)), for any vector $v \in V$ we can efficiently sample $x_1, x_2, x_3, x_4 \in X$ such that we can write

$$v = x_1 + x_2 - (x_3 + x_4) .$$

However, we need to be able to handle any vector $y \in \mathbb{F}^n$, and not just vectors in the subspace V . Towards that end, we show that since the subspace implied by the probabilistic quasi-polynomial Bogolyubov-Ruzsa lemma is of large dimension (i.e., of dimension

$\dim(V) \geq n - O(\log^4(1/\alpha))$, we can decompose any vector $y \in \mathbb{F}^n$ as a linear combination of the form

$$y = x_1 + x_2 - (x_3 + x_4) + s ,$$

where $x_1, x_2, x_3, x_4 \in X$ and $s \in \mathbb{F}^n$ is a *sparse* vector. We stress that the sparsity of the decomposition is essential to our applications, as we cannot locally correct the shift part of the decomposition, and instead we need to compute it explicitly.

The above captures our local correction lemma which will be used throughout Part I of this thesis.

Lemma 3.3.1 (Efficient local correction). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, and let $X \subseteq \mathbb{F}^n$ be a set of size $|X| = \alpha \cdot |\mathbb{F}|^n$, for some $\alpha \in (0, 1]$. Then, there exists a non-negative integer $t \leq O(\log^4(1/\alpha))$, a collection of t vectors $B = \{b_1, \dots, b_t \in \mathbb{F}^n\}$, and t indices $k_1, \dots, k_t \in [n]$ satisfying the following:*

Given a vector $y \in \mathbb{F}^n$, define $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$ where $(\vec{e}_i)_{i \in [n]}$ is the standard basis. Then

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2 \in X, x_3, x_4 \in -X] \geq \Omega(\alpha^5) ,$$

where $x_4 = y - s - x_1 - x_2 - x_3$.

Furthermore, suppose we have a randomized membership oracle O_X that for every input $x \in \mathbb{F}^n$, computes the indicator $1_X(x)$ correctly with probability at least $2/3$. Then, there exists an algorithm that makes $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ oracle calls to O_X , performs $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ field operations, and with probability at least $1 - \delta$ returns vectors $b_1, \dots, b_t \in \mathbb{F}^n$ and indices $k_1, \dots, k_t \in [n]$ as described above.

Proof. Fix a set $X \subseteq \mathbb{F}^n$ of size $|X| = \alpha \cdot |\mathbb{F}|^n$ for some $\alpha \in (0, 1]$. By applying [Lemma 3.1.3](#), we obtain a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) = n - t$ for $t = O(\log^4(1/\alpha))$. Let $R \subseteq \mathbb{F}_2^n \setminus \{\vec{0}\}$ be a set of vectors in $\mathbb{F}^n$ of size t such that $V = \{v \in \mathbb{F}_2^n : \langle v, r \rangle = 0 \ \forall r \in R\}$. Indeed, we can let R be a set of t linearly independent vectors in $V^\perp$.

By writing the vectors of R in a matrix and diagonalizing the matrix, we obtain: (1) a set of vectors $B = \{b_1, \dots, b_t \in \mathbb{F}_2^n\}$ such that $\text{span}(B) = \text{span}(R)$, and (2) the corresponding pivot indices $k_1, \dots, k_t \in [n]$ such that $b_j[k_j] = 1$ and $b_j[k_{j'}] = 0$ for all $j \neq j'$.

Given a vector $y \in \mathbb{F}^n$, define $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$, where $(\vec{e}_i)_{i \in [n]}$ is the standard basis, and let $v = y - s$. It is straightforward to verify that $v \in V$. Then for any $j \in [t]$ we have

$$\langle v, b_j \rangle = \langle y, b_j \rangle - \sum_{j=1}^t c_j \cdot \langle \vec{e}_{k_j}, b_j \rangle \stackrel{(*)}{=} \langle y, b_j \rangle - c_j \cdot \langle \vec{e}_{k_j}, b_j \rangle \stackrel{(**)}{=} \langle y, b_j \rangle - \langle y, b_j \rangle = 0 ,$$

where (*) is because $\langle \vec{e}_{k_{j'}}, b_j \rangle = b_j[i_{j'}] = 0$ for $j \neq j'$, and (**) is because $\langle \vec{e}_{k_j}, b_j \rangle = b_j[i_j] = 1$.

Now, since $v \in V$, by the guarantees of [Lemma 3.1.3](#) it follows that

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1 \in X, x_2 \in X, x_3 \in -X, v - x_1 - x_2 - x_3 \in -X] \geq \Omega(\alpha^5) .$$

For the furthermore part, note that we can boost the success probability of the membership oracle O_X . That is, given a query x we can decide if $x \in X$ with confidence $1 - \frac{1}{t}$ by repeatedly calling it $O(\log(t))$ times and taking the majority vote. In particular, for $t = \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$, by making $O(\text{poly} \log(1/\alpha) + \log(1/\delta) + \log(n))$ calls to $O_X(x)$ for each element $x \in \mathbb{F}^n$ we need to query, we may assume that all queries output whether $x \in X$ or not correctly.

The furthermore part of the lemma follows immediately from the computational guarantees of [Lemma 3.1.3](#) together with the diagonalization procedure described above. $\square$

3.4 Robust probabilistic Bogolyubov lemma

We now turn to proving our main technical tool for the quantum reduction: a robust quantum local correction lemma for linear problems.

Recall that Bogolyubov's lemma states that for any subset $A \subseteq \mathbb{F}_2^n$ of density $|A|/2^n \geq \alpha$, there exists a subspace $V \subseteq 4A$ of dimension at least $n - \alpha^{-2}$.

We would like to use Bogolyubov's lemma to locally correcting faulty inputs by *explicitly* computing a decomposition into a linear combination of *good inputs*, shifted by a sparse vector (see [Section 2.7](#)). Computing the sparse shift vector requires learning the (significant) Fourier spectrum of the subspace implied by Bogolyubov's lemma. The caveat is that our quantum algorithm for learning the subspace encoded in a noisy quantum state (see [Section 5.3.4](#)) can only obtain an approximation of the spectrum.

Hence, we need to strengthen Bogolyubov's lemma to obtain the following structural properties:

1. *Robustness*, in the sense that the linear constraints that define the Bogolyubov subspace can lie within a range of Fourier thresholds; and

2. *Density*, in the sense that each element of the Bogolyubov subspace admits many decompositions into valid inputs, and in turn can be sampled probabilistically using our quantum sampling procedure (see [Section 5.3.3](#)).

Next, we prove a generalization of Bogolyubov's lemma, which achieves the aforementioned structural properties. In the following, given a set $X \subseteq \mathbb{F}^n$ define $\text{Spec}_X(\gamma) = \{r \in \mathbb{F}^n \setminus \{0\} : |\hat{1}_X(r)| \geq \gamma\}$.

Lemma 3.4.1 (Robust Bogolyubov lemma). *Let $\mathbb{F} = \mathbb{F}_p$ be a finite field, and let $X \subseteq \mathbb{F}^n$ be a set of size $|X| = \alpha \cdot |\mathbb{F}|^n$ for some $\alpha \in (0, 1]$. Let $R \subseteq \mathbb{F}^n$ be a set such that $\text{Spec}_X(\alpha^{3/2}) \subseteq R \subseteq \text{Spec}_X(\frac{\alpha^{3/2}}{2})$.*

Let $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$. Then $\dim(V) \geq n - 4/\alpha^2$, and for all $v \in V$ it holds that

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2, x_3, v - x_1 - x_2 - x_3 \in X] \geq \alpha^5,$$

or equivalently

$$\Pr_{x_1, x_2, x_3 \in X} [v - x_1 - x_2 - x_3 \in X] \geq \alpha^2.$$

Proof. Note first that by Parseval's identity we have

$$\alpha = \langle 1_X, 1_X \rangle = \|1_X\|_2^2 = \sum_r |\hat{1}_X(r)|^2.$$

In particular,

$$\frac{\alpha^3}{4} \cdot |R| \leq \sum_{r \in R} |\hat{1}_X(r)|^2 \leq \sum_r |\hat{1}_X(r)|^2 = \alpha,$$

and hence $|R| \leq \frac{4}{\alpha^2}$. In particular, $\dim(V) \geq n - |R| \geq n - 4/\alpha^2$.

Furthermore, we have

$$\sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^4 \leq \alpha^3 \cdot \sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^2 \leq \alpha^3(\alpha - \alpha^2) \leq \alpha^4 - \alpha^5,$$

where the second inequality uses that $\sum_r |\hat{1}_X(r)|^2 = \alpha$, and $|\hat{1}_X(0)|^2 = \alpha^2$. Noting that

for every $v \in V$ we have $\chi_r(v) = \omega^{\langle v, r \rangle} = \omega^0 = 1$ for all $r \in R$, it follows that

$$\begin{aligned}
\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2, x_3, v - x_1 - x_2 - x_3 \in X] &= (1_X * 1_X * 1_X * 1_X)(v) \\
&= \sum_{r \in \mathbb{F}^n} (\hat{1}_X(r))^4 \chi_r(v) \\
&= |\hat{1}_X(0)|^4 \chi_0(v) + \sum_{r \in R} |\hat{1}_X(r)|^4 \chi_r(v) \\
&\quad + \sum_{r \in \mathbb{F}^n \setminus (R \cup \{0\})} |\hat{1}_X(r)|^4 \chi_r(v) \\
&\geq \alpha^4 + (\alpha^{3/2}/2)^4 \cdot |R| - (\alpha^4 - \alpha^5) \\
&\geq \alpha^5,
\end{aligned}$$

as required. $\square$

3.5 Quantum local correction lemma

Equipped with the noise-robust Bogolyubov lemma, we can now employ the four quantum procedures we showed in [Section 5.3](#) and prove the following quantum local correction lemma for linear problems, which lies at the heart of our worst-case to average-case reductions.

Lemma 3.5.1 (Quantum local correction). *Let $\mathbb{F} = \mathbb{F}_p$ be a finite field, and let $X \subseteq \mathbb{F}^n$ be a set of size $|X| = \alpha \cdot |\mathbb{F}|^n$, for some $\alpha \in (0, 1]$. Then, there exists a non-negative integer $t \leq 4/\alpha^2$, a collection of t vectors $B = \{b_1, \dots, b_t \in \mathbb{F}^n\}$, and t indices $k_1, \dots, k_t \in [n]$ satisfying the following:*

Given a vector $y \in \mathbb{F}^n$, define $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$ where $(\vec{e}_i)_{i \in [n]}$ is the standard basis. Then

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1, x_2, x_3, x_4 \in X] \geq \alpha^5,$$

where $x_4 = y - x_1 - x_2 - x_3 - s$. Equivalently, we have

$$\Pr_{x_1, x_2, x_3 \in X} [x_4 \in X] \geq \alpha^2.$$

Furthermore, suppose we have a quantum membership oracle $\tilde{O}_{X^*}$ (that we can query in superposition) for a set X^* satisfying the following conditions

- $|\hat{1}_{X^*}(r) - \hat{1}_X(r)| \leq \frac{\alpha^{3/2}}{8}$ for all $r \in \mathbb{F}^n$;
- for every input $x \in \mathbb{F}^n$, $\tilde{O}_{X^*}$ computes the indicator $1_{X^*}(x)$ correctly with probability at least $1 - \alpha^{3/2}/10$.

Then, there exists a quantum algorithm that with probability at least $1 - \delta$ returns vectors $b_1, \dots, b_t \in \mathbb{F}^n$ and indices $k_1, \dots, k_t \in [n]$ as described above. This algorithm makes $O(\alpha^{-7/2})$ blackbox queries to $\tilde{O}_{X^*}$, uses an additional number

$$O\left(n\alpha^{-7/2} \cdot \log n \cdot \text{poly log } |\mathbb{F}|\right)$$

of one-qubit and two-qubit gates, and $O(n\alpha^{-2} \log |\mathbb{F}|)$ ancillary qubits.

Proof. Fix a set $X \subseteq \mathbb{F}^n$ of size $|X| = \alpha \cdot |\mathbb{F}|^n$ for some $\alpha \in (0, 1]$. By applying [Lemma 3.4.1](#), we obtain a subspace $V \subseteq \mathbb{F}^n$ of dimension $\dim(V) = n - t$ for $t = 4/\alpha^2$. Let $R \subseteq \mathbb{F}^n \setminus \{0\}$ be a set of vectors in $\mathbb{F}^n$ of size t such that $V = \{v \in \mathbb{F}^n : \langle v, r \rangle = 0 \ \forall r \in R\}$. Indeed, we can let R be a set of t linearly independent vectors in $V^\perp$.

By writing the vectors of R in a matrix and diagonalizing the matrix, we obtain: (1) a set of vectors $B = \{b_1, \dots, b_t \in \mathbb{F}^n\}$ such that $\text{span}(B) = \text{span}(R)$, and (2) the corresponding pivot indices $k_1, \dots, k_t \in [n]$ such that $b_j[k_j] = 1$ and $b_j[k_{j'}] = 0$ for all $j \neq j'$.

Given a vector $y \in \mathbb{F}^n$, define $s = \sum_{j=1}^t \langle y, b_j \rangle \cdot \vec{e}_{k_j}$, where $(\vec{e}_i)_{i \in [n]}$ is the standard basis, and let $v = y - s$. It is straightforward to verify that $v \in V$. Then for any $j \in [t]$ we have

$$\langle v, b_j \rangle = \langle y, b_j \rangle - \sum_{j=1}^t c_j \cdot \langle \vec{e}_{k_j}, b_j \rangle \stackrel{(*)}{=} \langle y, b_j \rangle - c_j \cdot \langle \vec{e}_{k_j}, b_j \rangle \stackrel{(**)}{=} \langle y, b_j \rangle - \langle y, b_j \rangle = 0 \ ,$$

where $(*)$ is because $\langle \vec{e}_{k_{j'}}, b_j \rangle = b_j[i_{j'}] = 0$ for $j \neq j'$, and $(**)$ is because $\langle \vec{e}_{k_j}, b_j \rangle = b_j[i_j] = 1$.

Now, since $v \in V$, by the guarantees of [Lemma 3.4.1](#) it follows that

$$\Pr_{x_1, x_2, x_3 \in \mathbb{F}^n} [x_1 \in X, x_2 \in X, x_3 \in X, v - x_1 - x_2 - x_3 \in X] \geq \alpha^5 \ ,$$

which is equivalent to

$$\Pr_{x_1, x_2, x_3 \in X} [v - x_1 - x_2 - x_3 \in X] \geq \alpha^2 \ .$$

For the furthermore part, consider the oracle $\tilde{O}_{X^*}(x)$. By the closeness between the Fourier coefficients of 1_X and 1_{X^*} , we may apply [Lemma 3.4.1](#). Indeed, letting

$$R = \text{Spec}_{X^*}(\frac{3}{4}\alpha^{3/2}) ,$$

we have a set of Fourier coefficients satisfying the requirements of [Lemma 3.4.1](#), namely that $\text{Spec}_{X^*}(\alpha^{3/2}) \subseteq R \subseteq \text{Spec}_{X^*}(\alpha^{3/2}/2)$.

By definition, for every input $x \in \mathbb{F}^n$, $\tilde{O}_{X^*}(x)$ computes the indicator $1_{X^*}(x)$ correctly with probability at least $1 - \alpha^{3/2}/10$. Now, we can apply [Corollary 5.3.7](#) to find a $y \in \mathbb{F}^n$ such that $\hat{1}_{X^*}(y) \geq 3\alpha^{3/2}/4$ with $O(\alpha^{-3/2})$ blackbox queries to $\tilde{O}_{X^*}$, and $O(n\alpha^{-3/2} \cdot \text{poly log } |\mathbb{F}|)$ additional one-qubit and two-qubit gates. By the closeness between the Fourier coefficients of 1_X and 1_{X^*} , these y 's also satisfy $\hat{1}_X(y) \geq \alpha^{3/2}/2$. Since there are at most $t \leq 4/\alpha^2$ such coefficients, to find all of them with probability $1 - \delta$, we need to repeat the sampling procedure above $O(\alpha^{-2})$ times. Hence, the total query complexity is $O(\alpha^{-7/2})$ blackbox queries to $\tilde{O}_{X^*}$, the total number of additional gates is

$$O\left(n\alpha^{-7/2} \cdot \log n \cdot \text{poly log } |\mathbb{F}|\right) ,$$

and the number of ancillary qubits is $O(n \cdot \alpha^{-2} \log |\mathbb{F}|)$. □

Chapter 4

Classical Reductions

In this chapter, we formally state and prove the reductions in classical computational models. We start by proving the reduction for matrix multiplication, and then move on to prove the reduction for OMV and data structures.

4.1 Worst-case to average-case reductions for matrix multiplication

We prove the worst-case to average-case reduction for matrix multiplication problem in this section. Let's restate [Theorem 1](#) below.

Theorem 1. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Suppose that there exists an algorithm **ALG** that, on input two matrices $A, B \in \mathbb{F}^{n \times n}$ runs in time $T(n)$ and satisfies*

$$\Pr[\text{ALG}(A, B) = A \cdot B] \geq \alpha,$$

*where the probability is taken over the random inputs $A, B \in \mathbb{F}^{n \times n}$ and the randomness of **ALG**.*

- *If $|\mathbb{F}| \leq 2/\alpha$, then there exists a randomized algorithm **ALG'** that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $\frac{\exp(O(\log^5(1/\alpha)))}{\delta} \cdot T(n)$ and outputs AB with probability at least $1 - \delta$.*

- If $|\mathbb{F}| \geq 2/\alpha$, then there exists a randomized algorithm **ALG'** that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $O(\frac{1}{\delta \cdot \alpha^4} \cdot T(n))$ and outputs AB with probability at least $1 - \delta$.

We divide the proof into two parts, namely for when $|\mathbb{F}| \geq \alpha/2$ and when $|\mathbb{F}| \leq \alpha/2$. The proof for the former case is given in [Section 4.1.2](#) and the proof of the latter is given in [Section 4.1.1](#).

We would like to point out that when the field size is large enough, we can use the standard interpolation techniques for low-degree polynomials to prove the reduction. However, the problem becomes more challenging when the field size is small (say, $\mathbb{F} = \mathbb{F}_2$), and showing the reduction in this case requires novel ideas. We will discuss both cases in detail in the following sections.

4.1.1 Reduction for matrices over small fields

In this section, we show a worst-case to average-case reduction for matrix multiplication problem over small fields, namely, where $|\mathbb{F}| \leq 2/\alpha$, where α is the success rate of the average-case algorithm. Informally, we will show that if there exists an algorithm that is able to compute the multiplication for a small percentage of matrices, then it is possible to boost this algorithm such that it works for all matrices, without sacrificing the running time too much. Before we proceed with the formal result, we need the following lemma known as Freivalds' algorithm.

Lemma 4.1.1 (Freivalds' Algorithm [\[Fre77\]](#)). *Given matrices $A, B, C \in \mathbb{F}^{n \times n}$ there exist a probabilistic algorithm that verifies whether $A \cdot B = C$ with failure probability 2^{-k} where the algorithm runs in $O(kn^2)$.*

Throughout the proof, we will use Freivalds' algorithm to verify the result of matrix multiplication instances,

In particular, it suffices to design an algorithm that given two matrices A, B outputs their product with some non-negligible probability $\varepsilon > 0$. By repeating the algorithm $O(1/\varepsilon)$ times, we can boost the probability of outputting the correct answer to a constant arbitrarily close to 1. This is done by applying Freivalds' algorithm on each of the outputs of the algorithm, rejecting incorrect outputs with high probability, and accepting when the correct answer is found.

We now demonstrate the main result of this section, which corresponds to the first case in [Theorem 1](#).

Theorem 4.1.2. Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Suppose that there exists an algorithm **ALG** that, on input two matrices $A, B \in \mathbb{F}^{n \times n}$ runs in time $T(n)$ and satisfies

$$\Pr[\text{ALG}(A, B) = A \cdot B] \geq \alpha,$$

where the probability is taken over the random inputs $A, B \in \mathbb{F}^{n \times n}$ and the randomness of **ALG**. If $|\mathbb{F}| \leq 2/\alpha$, then there exists a randomized algorithm **ALG'** that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $O(\frac{\exp(O(\log^5(1/\alpha)))}{\delta} \cdot T(n))$ and outputs AB with probability at least $1 - \delta$.

Below we will prove the theorem assuming the algorithm **ALG** is deterministic, but a straightforward generalization of the proof works for randomized algorithms as well. To proceed with the proof, we first need the following definitions.

Definition 4.1.3. Let X be the set of matrices A such that **ALG** computes their product with matrices B with probability at least $\alpha/2$. More formally

$$X = \{A : \Pr_B[\text{ALG}(A, B) = A \cdot B] \geq \alpha/2\}.$$

Similarly, for each $A \in \mathbb{F}^{n \times n}$, we define Y_A to be the set of matrices B such that given A and B , **ALG** correctly computes $A \cdot B$. In other words

$$Y_A = \{B : \text{ALG}(A, B) = A \cdot B\}.$$

Claim 4.1.4. X and Y_A , where $A \in X$, have density at least $\alpha/2$.

Proof. Let P_A be the random variable $P_A := \Pr_B[\text{ALG}(A, B) = A \cdot B]$. From the definition, it is clear that $\mathbb{E}_A[P_A] \geq \alpha$. Now, by contradiction, if $\Pr_A[P_A \geq \alpha/2] < \alpha/2$ then we have

$$\mathbb{E}_A[P_A] = \mathbb{E}_A \Pr_B[\text{ALG}(A, B) = A \cdot B] < \alpha/2 \cdot 1 + (1 - \alpha/2) \cdot \alpha/2 < \alpha.$$

Hence, $\Pr[P_A \geq \alpha/2] \geq \alpha/2$ and X has density greater than or equal to $\alpha/2$. It follows from the definitions of X and Y_A that for all $A \in X$, Y_A has density at least $\alpha/2$. $\square$

Next, we need the following definition.

Definition 4.1.5. For a matrix $A \in \mathbb{F}^{n \times n}$ and $k \in [n]$, let $M_A^k = A + L_A^k$ where we define the matrix L_A^k as follows.

1. First, choose a random subset S of size k from $[n]$.

2. For each $i \in S$ let the i th row of L_A^k be uniformly random in $\mathbb{F}^n$.
3. For all $j \in [n] \setminus S$, let the j th row of L_A^k be a random linear combination of the rows in S .

Remark 4.1.6. For matrix L_A^k we have that $\text{rank}(L_A^k) \leq k$, because every row indexed by $j \in [n] \setminus S$ is a linear combination of the rows in S .

Remark 4.1.7. If the random rows in S are not linearly independent, we can throw them away and repeat Step 2. It is not hard to see that this event happens only with constant probability, and we can check this in $O(nk^2)$ time.

The following lemma shows that matrix $M_A^{2k} = A + (L_A^{2k})$ belongs to any subspace of matrices of constant co-dimension k with constant probability.

Lemma 4.1.8. Given a matrix $A \in \mathbb{F}^{n \times n}$ and $k \in [n]$, for any subspace $V \subseteq \mathbb{F}^{n \times n}$ of $\dim(V) \geq n^2 - k$ we have

$$\Pr[M_A^{2k} \in V] \geq \frac{1}{2^{|\mathbb{F}|^k}} .$$

Proof. Since V has co-dimension k , it can be defined by k linear constraints on the elements of the matrix as follows.

$$M_A^{2k}(i_0, j_0) + M_A^{2k}(i_1, j_1) + \dots + M_A^{2k}(i_r, j_r) = 0 ,$$

where $r \in [1, n^2]$ denotes the number of coordinates that this constraint depends on. By re-writing M_A^{2k} as a vector $\mathbf{m} \in \mathbb{F}^{n^2}$, we can construct the system of equations $G \cdot \mathbf{m} = \mathbf{0}$ for membership in V . Here, G denotes the matrix of size $k \times n^2$, where each row specifies one single constraint of the aforementioned form. Now, if we diagonalize G using Gaussian elimination, we can re-write the system of equations in the form $G' \cdot \mathbf{m} = \mathbf{0}$ for a matrix G' , where for each row a in G' , there exists a column b_a which has value 1 in this row and 0 in the other rows.

For all b_a with $a \in [k]$, we consider the coordinate $\mathbf{m}_{b_a}$. The set of these k coordinates $\{\mathbf{m}_{b_1}, \mathbf{m}_{b_2}, \dots, \mathbf{m}_{b_k}\}$ corresponds to k pairs of coordinates $\{(c_1, c'_1), (c_2, c'_2), \dots, (c_k, c'_k)\}$ in the original matrix. Note that these k coordinates belong to at most k rows in M_A^{2k} , and we want to lower bound the probability that these rows in L_A^{2k} are linear independent. Let Z

be the event that these k rows are linearly independent in L_A^{2k} . Then

$$\begin{aligned}\Pr[Z] &= \left(1 - \frac{1}{2^{2k}}\right) \left(1 - \frac{2}{2^{2k}}\right) \left(1 - \frac{4}{2^{2k}}\right) \dots \left(1 - \frac{2^{k-1}}{2^{2k}}\right) \\ &\geq \left(1 - \frac{2^{k-1}}{2^{2k}}\right)^k \geq \left(1 - \frac{1}{2^{k+1}}\right)^k \\ &\geq 1 - \frac{k}{2^{k+1}} \geq \frac{1}{2} .\end{aligned}$$

Now, we observe that if Z happens, it means that for all k constraints, there exist a coordinate which we denote by (c_i, c'_i) in M_A^{2k} such that none of the other constraints depend on the value of $M_A^{2k}(c_i, c'_i)$, and the value of $M_A^{2k}(c_i, c'_i)$ is chosen uniformly at random. Hence, this random value is equal to the unique solution which satisfies i th constraint (assuming values of all other coordinates involved in this constraint are determined beforehand) with probability $1/|\mathbb{F}|$. Therefore, the probability that $M_A^{2k} \in V$ is bounded by

$$\begin{aligned}\Pr[M_A^{2k} \in V] &= \Pr[\text{All } k \text{ constraints are satisfied}] \\ &\geq \Pr[Z] \cdot \frac{1}{|\mathbb{F}|^k} \\ &= \frac{1}{2|\mathbb{F}|^k} .\end{aligned}\quad \square$$

Proof of Theorem 4.1.2. To prove this theorem, we design the following algorithm and prove that this algorithm outputs the correct answer for the matrix multiplication problem with high probability.

Algorithm 1 : Matrix multiplication reduction over small fields

Input: ALG , $A, B \in \mathbb{F}^{n \times n}$

Output: $A \cdot B$

1. Set k to be $O(\log^4(1/\alpha))$.
2. For matrices A and B , construct the matrices M_A^{2k} and M_B^{2k} .
3. Sample 3 random matrices $R_1, R_2, R_3 \in \mathbb{F}^{n \times n}$ and set $R_4 = R_1 + R_2 - R_3 - A - M_A^{2k}$ so that $A + M_A^{2k} = R_1 + R_2 - R_3 - R_4$.
4. Sample 12 random matrices $S_1^{(t)}, S_2^{(t)}, S_3^{(t)} \in \mathbb{F}^{n \times n}$ and set $S_4^{(t)} = S_1^{(t)} + S_2^{(t)} - S_3^{(t)} - B - M_B^{2k}$ for $t \in \{1, 2, 3, 4\}$, so that $B + M_B^{2k} = S_1^{(t)} + S_2^{(t)} - S_3^{(t)} - S_4^{(t)}$.
5. Compute $O_L = \sum_{t=1}^4 \sum_{s=1}^4 \text{sign}_{t,s} \text{ALG}(R_t, S_s^{(t)})$, where $\text{sign}_{t,s} = -1$ if $\{t, s\} \cap \{1, 2\} = 1$, and $\text{sign}_{t,s} = 1$ otherwise.
6. Compute $O = O_L - A \cdot L_B^{2k} - L_A^{2k} \cdot B - L_A^{2k} \cdot R_B^{2k}$.
7. If $O = A \cdot B$ (check using [Lemma 4.1.1](#)), then return O .

Correctness: Let's consider X as it is defined in [Definition 4.1.3](#). By applying [Lemma 3.1.3](#) on X , we can conclude that there exists subspace V_X with co-dimension at most $O(\log^4(1/\alpha))$ and the guaranteed properties. On the other hand, by [Lemma 4.1.8](#) we have that

$$\Pr[M_A^{2k} \in V_X] \geq \frac{1}{2^{|\mathbb{F}|^{O(\log^4(1/\alpha))}}}.$$

Assuming $M_A^{2k} \in V_X$, by [Lemma 3.1.3](#), we have that

$$\Pr_{R_1, R_2, R_3}[R_1, R_2, -R_3, -R_4 \in X] \geq \Omega(\alpha^5).$$

Now, for each R_t with $t \in \{1, 2, 3, 4\}$, we can consider Y_{R_t} using [Definition 4.1.3](#). Similarly, since each Y_{R_t} has density at least $\alpha/2$, we can apply [Lemma 3.1.3](#) on Y_{R_t} to define the corresponding subspaces $V_{Y_{R_t}}$. Having these four subspaces, we define $V_Y = V_{Y_{R_1}} \cap V_{Y_{R_2}} \cap V_{Y_{-R_3}} \cap V_{Y_{-R_4}}$. It is not hard to see that since each of the four subspaces has co-dimension

$O(\log^4(1/\alpha))$, the co-dimension of V_Y is at most $4 \cdot O(\log^4(1/\alpha))$. Thus, by [Lemma 4.1.8](#)

$$\Pr[M_B^{2k} \in V_Y] \geq \frac{1}{2^{|\mathbb{F}|^{O(\log^4(1/\alpha))}}}.$$

Given $M_B^{2k} \in V_Y$, for each $t \in \{1, 2, 3, 4\}$ by [Lemma 3.1.3](#)

$$\Pr_{S_1^{(t)}, S_2^{(t)}, S_3^{(t)}}[S_1^{(t)}, S_2^{(t)}, -S_3^{(t)}, -S_4^{(t)} \in Y_{R_t}] \geq \Omega(\alpha^5).$$

It is important to note that since L_A^{2k} and L_B^{2k} have rank less than or equal to $2k$, and all linear combinations of their rows are known previously, we can compute the multiplications in [Step 6](#) in time $O(n^2 \cdot \log^4(1/\alpha))$.

Since all the events defined above are independent, we conclude that the algorithm succeeds with the following probability

$$\Pr[\text{Algorithm 1 succeeds}] \geq \frac{\Omega(\alpha^{25})}{O(|\mathbb{F}|^{O(\log^4(1/\alpha))})} \geq \frac{\Omega(\alpha^{25})}{O(\frac{10}{\alpha})^{O(\log^4(1/\alpha))}} \geq \exp(-\log^5(1/\alpha)).$$

Therefore, by repeating the algorithm $\frac{\exp(\log^5(1/\alpha))}{\delta}$ times, and using Freivalds' algorithm for verification, we obtain an algorithm that solves the matrix multiplication on all instances with probability at least $1 - \delta$.

Running time: The running time of the procedure described above is essentially dominated by $\frac{\exp(\log^5(1/\alpha))}{\delta}$ calls to the weak average case algorithm, and hence the total running time is $\frac{\exp(\log^5(1/\alpha))}{\delta} \cdot T(n)$. In particular, even if the algorithm succeeds on a sub-constant fraction of inputs $\alpha = \exp(\log^{0.199}(n))$, the reduction turns it into an algorithm that works for worst case instances in time $T(n) \cdot n^{o(1)}$. $\square$

4.1.2 Reduction for matrices over large fields

We now prove case 2 of [Theorem 1](#) in this section. For concreteness, we restate this result.

Theorem 4.1.9. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Suppose that there exists an algorithm **ALG** that, on input two matrices $A, B \in \mathbb{F}^{n \times n}$ runs in time $T(n)$*

and satisfies

$$\Pr[\text{ALG}(A, B) = A \cdot B] \geq \alpha,$$

where the probability is taken over the random inputs $A, B \in \mathbb{F}^{n \times n}$ and the randomness of ALG . If $|\mathbb{F}| \geq 2/\alpha$, then there exists a randomized algorithm ALG' that for every input $A, B \in \mathbb{F}^{n \times n}$ and $\delta > 0$, runs in time $O(\frac{1}{\delta \cdot \alpha^4} \cdot T(n))$ and outputs AB with probability at least $1 - \delta$.

Proof. To prove [Theorem 4.1.9](#), we use the following algorithm and we prove that this algorithm outputs the correct answer for the matrix multiplication problem with high probability.

Algorithm 2 : Matrix multiplication reduction over large fields

Input: $\text{ALG}, A, B \in \mathbb{F}^{n \times n}$

Output: $A \cdot B$

1. Let X and Y be matrices chosen uniformly at random from $\mathbb{F}^{n \times n}$, and let i, j , and k be chosen uniformly at random from $\mathbb{F}$.
2. Compute $\text{ALG}(A+iX, B+iY)$, $\text{ALG}(A+jX, B+jY)$, and $\text{ALG}(A+kX, B+kY)$.
3. If the computations are correct (check using [Lemma 4.1.1](#)), then compute $A \cdot B$ by interpolating $(A+iX, B+iY)$, $(A+jX, B+jY)$, and $(A+kX, B+kY)$.

Correctness: Again, we prove the result assuming the algorithm ALG is deterministic, but a straightforward generalization of the proof works for randomized algorithms as well. We define the set of good pairs of matrices for ALG as follows.

Definition 4.1.10. Let $S \subseteq (\mathbb{F}^{n \times n} \times \mathbb{F}^{n \times n})$ be the set of pairs of matrices such that for all $(M, N) \in S$ we have $\text{ALG}(M, N) = M \cdot N$. More formally

$$S = \{(M, N) : \text{ALG}(M, N) = M \cdot N, M \in \mathbb{F}^{n \times n}, N \in \mathbb{F}^{n \times n}\}.$$

Note that by definition, S has density at least α .

Claim 4.1.11. Let $\ell_{X,Y} = \{(A+iX, B+iY) : i \in \mathbb{F}\}$ be the line that passes through (A, B) and is defined by matrices X and Y . Then, with probability $\alpha/2$, at least $\alpha/2$ fraction of pairs of matrices on this line belong to S .

Proof. Let $P_{(X,Y)}$ be the random variable $P_{(X,Y)} = \Pr_i[\text{ALG}(A+iX, B+iY) = (A+iX) \cdot (B+iY)]$. From the definition, $\mathbb{E}[P_{(X,Y)}] \geq \alpha$. Now, by contradiction, if $\Pr[P_{(X,Y)} \geq \alpha/2] < \alpha/2$ then we have

$$\begin{aligned} \mathbb{E}[P_{(X,Y)}] &= \mathbb{E}_{X,Y} \Pr_i[\text{ALG}(A+iX, B+iY) = (A+iX) \cdot (B+iY)] \\ &< \alpha/2 \cdot 1 + (1 - \alpha/2) \cdot \alpha/2 < \alpha . \end{aligned}$$

Hence, $\Pr[P_{(X,Y)} \geq \alpha/2] \geq \alpha/2$. □

Having [Definition 4.1.10](#), we can make the following claim.

Claim 4.1.12. *The three pairs of matrices defined in Step 2 of Algorithm 2 belong to S with probability at least $\alpha^4/16$.*

Proof. Note that in Step 1 of Algorithm 2, we are sampling a line $\ell_{X,Y}$, which by [Claim 4.1.11](#) has density $\alpha/2$ of good pair of matrices with probability $\alpha/2$. Also, in Step 2, we are sampling three uniformly random pairs on this line. Assuming $\ell_{X,Y}$ is a line with density $\alpha/2$ of good pairs of matrices, with probability at least $(\alpha/2)^3$ these three pairs belong to S . Hence, total probability that we sample three pairs such that they all belong to S is at least $(\alpha/2)^4 = \alpha^4/16$. □

Since matrix multiplication is a polynomial of degree 2 in the entries of the matrices, having 3 pairs where **ALG** outputs correct answers enables us to interpolate the value of $A \cdot B$. Thus, the algorithm succeeds with probability $O(\alpha^4)$. By repeating the algorithm and verifying the answer using Freivalds' algorithm, one can amplify the success probability to any arbitrary constant. □

4.2 Worst-case to average-case reductions for online matrix-vector multiplication

In this section we prove [Theorem 3](#), which we restate below.

Theorem 3. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Consider the matrix-vector multiplication problem $\text{OMV}_{\mathbb{F}}$ for dimension n , and suppose that for some*

$\alpha > 0$ it holds that

$$OMV_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{M,v}[\text{DS}_M(v) = Mv] \geq \alpha \end{array} \right].$$

Then for every $\delta > 0$,

$$OMV_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & 4p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha)n) + O(n^2) \\ \text{query time:} & (4t + n) \cdot \text{poly}(1/\alpha) \cdot \text{poly} \log(1/\delta) \\ \text{success rate:} & \forall M, v : \Pr[\text{DS}_M(v) = Mv] \geq 1 - \delta \end{array} \right].$$

Remark 4.2.1 (Large fields). *We would like to point out that this problem is more interesting when the field $\mathbb{F}$ is small. Indeed, if the size of $\mathbb{F}$ is relatively large (say, $|\mathbb{F}| > 2/\alpha$), then we can think of the matrix-vector multiplication as a polynomial of degree at most one in the elements of the vector. Therefore, we can use the standard self-correction techniques for evaluating low-degree polynomials to solve this problem. In particular, given a query $v \in \mathbb{F}^n$ we can sample a line $\ell \in \mathbb{F}^n$ that passes through v , and query two random vectors that belong to ℓ , and compute Mv by interpolating the two queried points.*

Before proceeding with the formal proof of [Theorem 3](#), we informally outline the argument.

Proof sketch: The proof of [Theorem 3](#) relies on [Lemma 3.3.1](#), which shows that any vector in $\mathbb{F}^n$ can be self-corrected via a linear constraint involving four vectors $x_1, x_2, x_3, x_4 \in X$ and a shift by sparse vector u . This result will be used several times in the proof of [Theorem 3](#), which we explain below.

- First, note that if $\Pr_{M,v}[\text{DS}_M(v) = Mv] \geq \alpha$, then there is a collection $Z \subseteq \mathbb{F}^{n \times n}$ of size $|Z| \geq \alpha/2 \cdot |\mathbb{F}|^{n^2}$ of *good* matrices, i.e., those matrices M on which DS succeeds to compute Mv on some non-negligible fraction of vectors v . More formally, $Z = \{M \in \mathbb{F}^{n \times n} : \Pr_v[\text{DS}_M(v) = Mv] \geq \alpha/2\}$.
- First time we apply [Lemma 3.3.1](#) on the set $Z \subseteq \mathbb{F}^{n \times n}$. (That is, we identify $n \times n$ matrices with vectors of length $N = n^2$.) Roughly speaking, given an arbitrary matrix M we will apply the lemma so that we can write $M = M_1 + M_2 - M_3 - M_4 + U$, where

$M_1, M_2, M_3, M_4 \in Z$ and U is a sparse matrix. By the assumption in [Theorem 3](#) each M_i succeeds on a non-negligible fraction of vectors v .

- Second time [Lemma 3.3.1](#) will be used with the set $X = X_{M_i}$ of vectors $v \in \mathbb{F}^n$ on which DS outputs $M_i v$ correctly, where M_i is each of the matrices above. Using the lemma we will be able to represent every vector $v \in \mathbb{F}^n$ as $v = x_1 + x_2 - x_3 - x_4 + u$, where the x_j 's belong to X_{M_i} , i.e., the data structure outputs $M_i x_j$ correctly for all $j = 1, 2, 3, 4$, and $u \in \mathbb{F}^n$ is a sparse vector.
- In particular, for each of the matrices M_i the data structure computes correctly $M_i x_j$ for all $j = 1, 2, 3, 4$, and $M_i u$ can be computed in the query phase by reading only $O(1)$ columns of M_i , as u is a sparse vector.

Before proceeding with the formal proof of [Theorem 3](#), we need the following definition of small-bias sample spaces, and the theorem regarding their existence.

Definition 4.2.2 (Small-bias sample spaces). *A sample space S over $\mathbb{F}^n$ is called ε -biased if for every $r \in \mathbb{F}^n \setminus \{0\}$ and every $b \in \mathbb{F}$ it holds that*

$$\left| \Pr_{s \in S}[\langle s, r \rangle = b] - \frac{1}{|\mathbb{F}|} \right| \leq \varepsilon .$$

In other words, a sample space is ε -biased if it ε -fools every nontrivial linear test, i.e., for any $r \in \mathbb{F}^n \setminus \{0\}$ the distribution of $\langle s, r \rangle$ with s sampled from S is close in distribution to $\langle s, r \rangle$ for a *uniformly random* $s \in \mathbb{F}^n$. When the exact value of ε is not important, e.g., by setting $\varepsilon = 0.1$, we usually call such sample spaces *small-bias sets*. These objects have been introduced in the work of Naor and Naor [[NN93](#)], followed by a long line of work culminating in the recent almost optimal construction of Ta-Shma [[TS17](#)], who showed an efficient construction of such sets of size $O(n/\varepsilon^{2+o(1)})$ over $\mathbb{F}_2$. For our purposes, even a randomized construction will be sufficient (see, e.g., Corollary 3.3 in [[AMN98](#)]).

Theorem 4.2.3. *For every finite field $\mathbb{F}$, constant $\varepsilon \in [0, 1]$ and $n \in \mathbb{N}$, a random set $S \subseteq \mathbb{F}^n$ of size $O(n \log |\mathbb{F}|)$ is an ε -biased with high probability. For the field $\mathbb{F} = \mathbb{F}_2$, there exists an explicit construction of size $|S| = O(n)$.*

For concreteness, we will take $\varepsilon = 0.1$, which suffices for our application.

We are now ready to prove [Theorem 3](#).

Proof of Theorem 3. For each matrix $M \in \mathbb{F}^{n \times n}$, let DS_M be the weak average-case data structure implied by the hypothesis of the theorem, and denote by

$$X_M = \{x \in \mathbb{F}^n : \text{DS}_M(x) = Mx\} ,$$

the set of vectors on which the data structure outputs the correct answer. Let $Z \subseteq \mathbb{F}^{n \times n}$ be the set of matrices on which the data structure outputs the correct answer on at least $\frac{\alpha}{2}$ -fraction of the inputs; that is, $Z = \{M \in \mathbb{F}^{n \times n} : |X_M| \geq \frac{\alpha}{2} |\mathbb{F}|^n\}$. By Markov's inequality, we have $|Z| \geq \frac{\alpha}{2} \cdot |\mathbb{F}|^{n^2}$. Observe that given access to DS_M , we can easily construct a probabilistic oracle for approximate membership in Z .

Claim 4.2.4. *There exists a probabilistic membership oracle O_Z that for any query $M \in \mathbb{F}^{n \times n}$ makes $t = O(1/\alpha)$ calls to $\text{DS}_M(x)$ for uniformly random $x \in \mathbb{F}^n$, compares the result to Mx , and accepts if and only if at least $\alpha/3$ fraction of the calls to $\text{DS}_M(x)$ output the correct answer. The oracle has the following guarantees.*

- If $M \in Z$, then $\Pr[O_Z(M) = \text{ACCEPT}] > 2/3$.
- If $|X_M| \leq \frac{\alpha}{4} |\mathbb{F}|^n$ then $\Pr[O_Z(M) = \text{REJECT}] > 2/3$.

Using the weak average-case data structure DS_M , we construct a worst-case data structure DS' as follows. First, we describe the preprocessing stage of the data structure $\text{DS}'^{(M)}$.

Preprocessing:

Input: A matrix $M \in \mathbb{F}^{n \times n}$

1. **Self-correcting M :** Using [Lemma 3.3.1](#) with probability $1 - \delta$ we represent the matrix M as $M = M_1 + M_2 - M_3 - M_4 + U$, where each M_i has a large X_{M_i} and U is a t -sparse matrix for $t = O(\log^4(1/\alpha))$. The running time of this step is $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$
2. **Self-correcting x :** Then, for each M_i , we apply [Lemma 3.3.1](#) on $X_{M_i} = \{x \in \mathbb{F}^n : \text{DS}_{M_i}(x) = M_i x\}$, and compute a collection of $t \leq O(\log^4(1/\alpha))$ vectors $B_i = \{b_1^{(i)}, \dots, b_t^{(i)} \in \mathbb{F}^n\}$ and t indices $k_1^{(i)}, \dots, k_t^{(i)} \in [n]$ that allow us to represent each vector $x = x_1 + x_2 - x_3 - x_4 + u_i$, where u_i has at most $O(\log^4(1/\alpha))$ non-zero elements, and $x_j \in X_{M_i}$ for all $i = 1, 2, 3, 4$.
3. Let $S \subseteq \mathbb{F}^n$ be a small-biased set obtained by taking $O(n)$ uniformly random vertices in $\mathbb{F}^n$. Note that for $\mathbb{F} = \mathbb{F}_2$ we can take the explicit set S from [Theorem 4.2.3](#) with $\varepsilon = 0.1$.
4. For each $e \in S$ compute the multiplication from the left eM_i of e with each of the M_i , and store the pairs (e, eM_i) in the data structure.

Overall, the data structure stores the following information in the preprocessing step:

- The sparse matrix $U \in \mathbb{F}^{n \times n}$ with at most $O(\log^4(1/\alpha))$ non-zero entries, obtained in Step 1;
- For each M_i , the weak average-case data structure DS_{M_i} for M_i , which outputs $\text{DS}_{M_i}(x) = M_i \cdot x$ correctly on at least $\alpha/4$ fraction of x 's.
- For each M_i , the corresponding $t = O(\log^4(1/\alpha))$ vectors $B_i = \{b_1^{(i)}, \dots, b_t^{(i)} \in \mathbb{F}^n\}$, and t indices $k_1^{(i)}, \dots, k_t^{(i)} \in [n]$, obtained in Step 2;
- The pairs (e, eM_i) for every vector e in the small-biased set $S \subseteq \mathbb{F}^n$ and for every M_i , obtained in Step 3.

Preprocessing time: The preprocessing time is determined by the time needed to find matrices M_i , and the preprocessing time of the weak-average-case data structures. It is easy to verify that the preprocessing time is bounded by $4p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$, where p is the preprocessing time of the weak-average-case data structure.

Memory used: In the preprocessing step, we store (1) 4 weak-average-case data structures of size s for each of the matrices M_i , (2) for each matrix M_i we store the collection of $t = O(\log^4(1/\alpha))$ vectors B_i and t indices, (3) a representation of the sparse matrix U using $O(\log^4(1/\alpha)\log(n))$ field elements. (4) The pairs (e, eM_i) for every vector e in the small-biased set S , which is of size $O(n)$. Hence, the total space used is $4s + O(\log^4(1/\alpha)n) + O(n^2)$.

Next we describe the query phase of the worst-case data structure. Recall, for each matrix M_i where $i = 1, 2, 3, 4$, we store the vectors $B_i = \{b_1^{(i)}, \dots, b_h^{(i)} \in \mathbb{F}^n\}$ and indices $k_1^{(i)}, \dots, k_h^{(i)} \in [n]$ in our data structure, to compute u_i every vector in $\mathbb{F}^n$ can be written as a linear combination of four vectors in X_{M_i} . The query phase works as follows.

Query phase:

Input: A query $x \in \mathbb{F}^n$

1. For $i \in \{1, 2, 3, 4\}$, sample random $x_1^{(i)}, x_2^{(i)}, x_3^{(i)} \in \mathbb{F}^n$ and let $x_4^{(i)}$ be such that $x = u_i + x_1^{(i)} + x_2^{(i)} - x_3^{(i)} - x_4^{(i)}$.
2. For each matrix M_i and for $j \in \{1, 2, 3, 4\}$, apply $\text{DS}_\alpha^{(M_i)}$ to $x_j^{(i)}$.
3. Verify that $\text{DS}^{M_i}(x_j^{(i)}) = M_i x_j^{(i)}$ using the small biased set S . Specifically, sample $O(\log(1/\delta))$ vectors $e \in S$, and check that

$$\langle e, \text{DS}_{M_i}(x_j^{(i)}) \rangle = \langle e M_i, x_j^{(i)} \rangle .$$

If the answer of $\text{DS}_{M_i}(x_j^{(i)})$ outputs the correct answer, then the inner products will all be equal. If $\text{DS}_{M_i}(x_j) \neq M_i x_j^{(i)}$, then a random $e \in S$ will catch an inequality with probability at least 0.4.

4. By repeating the sampling above for $O(\log(1/\delta) \cdot 1/\alpha^5)$ times, for each $i \in \{1, 2, 3, 4\}$ we will find such $x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, x_4^{(i)}$ on which $\text{DS}_{M_i}(x_j^{(i)})$ outputs the correct answer with high probability.
5. Compute $M_i u_i$ directly. Since u_i has at most $O(\log^4(1/\alpha))$ non-zero coordinates, it follows that $M_i u_i$ can be computed in time $O(\log^4(1/\alpha)n)$.
6. For $i \in \{1, 2, 3, 4\}$ compute $M_i y$ by taking $M_i x_1^{(i)} + M_i x_2^{(i)} - M_i x_3^{(i)} - M_i x_4^{(i)} + M_i u_i$.
7. Compute Uy directly. Since U has at most $O(\log^4(1/\alpha))$ non-zero elements, this can be done in time $O(\log^4(1/\alpha) \cdot \log(n))$.
8. Return $My = M_1 y + M_2 y - M_3 y - M_4 y + Uy$.

Correctness: To prove the correctness, we bound the failure probability of the algorithm. Note that the failure of the algorithm only depends on the verification procedure in Step 3. In other words, if $\text{DS}'_M(x) \neq Mx$, then at least for one pair of (i, j) we have that $\text{DS}_{M_i}a(x_j^{(i)}) \neq M_i x_j^{(i)}$, and none of the sampled vectors e has caught this inequality. On the other hand, this event happens with probability at most $0.6^{O(\log(1/\delta))}$, bounding the failure probability to be at most δ .

Query time: The query time consist of the time required to compute $M_i x_j^{(i)}$, the time required to compute $M_i u_i$, and the time needed to compute Uy where $i, j \in \{1, 2, 3, 4\}$. The sampling in Step 1 will be done $O(\log(1/\delta) \cdot 1/\alpha^5)$ times, and for each sampled vectors, DS_{M_i} is applied 4 times. Also, the verification in Step 3 consists of computing the inner product for $O(\log(1/\delta))$ many vectors. Thus, the total query time is equal to

$$\begin{aligned} & O(\log(1/\delta) \cdot 1/\alpha^5) \cdot 4 \cdot (t + O(\log(1/\delta)n) + O(n \log^4(1/\alpha))) \\ &= (4t + n) \cdot \text{poly}(1/\alpha) \cdot \text{poly} \log(1/\delta) . \end{aligned}$$

This completes the proof of [Theorem 3](#). □

4.3 Worst-case to average-case reductions for data structures

In this section, we show worst-case to average-case reductions in the setting of static data structures. We start by showing a reduction for all linear data structure problems in [Section 4.3.1](#).

Then, we consider a more powerful type of reductions, which can be used to derive worst-case algorithms from data structures that only satisfy a weak average case condition over both inputs and queries (similarly to the setting of online matrix-vector multiplication). On the negative side, we give a counterexample, showing that general weak-average-case reductions cannot hold for all linear problems. On the positive side, we show that the problem of evaluating a multivariate polynomial admits such a weak-average-case reduction. We stress that as opposed to the online matrix-vector multiplication problem discussed above, the problem of multivariate polynomial evaluation is an example of a non-linear problem admitting such a reduction.

4.3.1 Average-case reductions for all linear problems

Recall that in the setting of data structures, a linear problem over a field $\mathbb{F}$ is defined by a matrix $A \in \mathbb{F}^{m \times n}$. The input to the data structure is a vector $x \in \mathbb{F}^n$, which is preprocessed into s memory cells. Then, given queries of the form $i \in [m]$, the goal of the data structure is to output $\langle A_i, x \rangle$, where A_i is the i th row of A . We show a worst-case to average-case reduction for data structures for *all* linear problems.

Remark 4.3.1. We note that the presented reduction results in uniform data structures. That is, we give an efficient and simple procedure that, given an average-case data structure, creates a worst-case data structure in a black-box way that works for all values of n .

There is a trivial folklore argument that transforms an average-case data structure into a non-uniform worst-case data structure as follows. Let $X \subseteq \mathbb{F}^n$ of size $|X| \geq \alpha|\mathbb{F}|^n$ be the set where for a given n , the average-case data computes all queries correctly. By the probabilistic method, there exists $(n/\alpha) \log |\mathbb{F}|$ shifts of X that cover all of $\mathbb{F}^n$. For every n , a non-uniform data structure will remember all of those shifts, and for each shift $s \in \mathbb{F}^n$, it will also remember the product As . Now, given an input vector x , in the preprocessing stage, the data structure just stores the index of a shift s such that $x + s \in X$, and in the query phase it reads the index of the shift and, thus, learns As . Now, since $x + s \in X$, we can use the average-case data structure to compute $A(x + s)$, and, finally, compute $Ax = A(x + s) - As$. This results in a non-uniform worst-case data structure whose space complexity and query time differ from those of the average-case data structure by an additive term of $\log((n/\alpha) \log |\mathbb{F}|)$.

Theorem 2. Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $\alpha := \alpha(n) \in (0, 1]$, $n, m \in \mathbb{N}$, and a matrix $A \in \mathbb{F}^{m \times n}$. Denote by L_A the linear problem of outputting $\langle A_i, x \rangle$ on input $x \in \mathbb{F}^n$ and query $i \in [m]$. Suppose that

$$L_A \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{x \in \mathbb{F}^n} [\text{DS}_x(i) = \langle A_i, x \rangle \forall i \in [m]] \geq \alpha \end{array} \right].$$

Then for every $\delta > 0$,

$$L_A \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha) \log(n)) \\ \text{query time:} & 4t + O(\log^4(1/\alpha) \log(n)) \\ \text{success rate:} & \forall x \in \mathbb{F}^n \quad \Pr[\text{DS}'_x(i) = \langle A_i, x \rangle \forall i \in [m]] \geq 1 - \delta \end{array} \right].$$

Proof. Consider the data structure DS for the matrix A , implied by the assumption of the theorem. There exists a subset $X \subseteq \mathbb{F}^n$ of size $|X| \geq \alpha|\mathbb{F}|^n$ such that for every input $x \in X$ the data structure answers correctly all queries to the data structure, i.e., $\text{DS}_x(i) = \langle A_i, x \rangle$ for all $i \in [m]$ and $x \in X$.

We design a data structure DS' that outputs correct answers in worst case as follows. Let $x \in \mathbb{F}^n$ be the input to DS' . We start by describing preprocessing and query phases of the data structure.

Worst-case data structure for L_A

Preprocessing: Given $x \in \mathbb{F}^n$ we apply the [Lemma 3.3.1](#) on x with the set X , and obtain a non-negative integer $t \leq O(\log^4(1/\alpha))$, a vector $v \in \mathbb{F}^n$ with at most t non-zero entries, and $x_1, x_2, x_3, x_4 \in X$ such that

$$x = x_1 + x_2 - x_3 - x_4 + v .$$

Furthermore, by [Lemma 3.3.1](#) such decomposition can be found using $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$ field operations with probability at least $1 - \delta$. Then, we use the preprocessing algorithm of the average-case data structure on each one of the x_j 's to obtain the algorithms $DS_{x_1}(\cdot), DS_{x_2}(\cdot), DS_{x_3}(\cdot), DS_{x_4}(\cdot)$. Finally, we store the sparse shift vector v by storing the t coordinates, and their values. Therefore, the amount of memory used is $4s + O(\log^4(1/\alpha) \log(n))$.

Query phase: Given a query $i \in [m]$, we invoke our four instantiations of the average-case data structure stored in the preprocessing stage and compute $DS_{x_1}(i), DS_{x_2}(i), DS_{x_3}(i), DS_{x_4}(i)$. We then compute $\langle A_i, v \rangle$ and return

$$DS_{x_1}(i) + DS_{x_2}(i) - DS_{x_3}(i) - DS_{x_4}(i) + \langle A_i, v \rangle .$$

Complexity: The time and amount of memory used follow immediately from the description. Namely, note that applying the local correction lemma, which dominates the time complexity, is done in time $\exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n)$. Hence the total preprocessing time is $4p + \exp(\log^4(1/\alpha)) \cdot \text{poly}(n)$.

In terms of memory, we store 4 instances of the average-case data structure DS , where each instance requires s memory cells. In addition we store the sparse vector v , by storing its t non-zero indices and their values. Hence the total memory required is $4s + O(\log^4(1/\alpha) \cdot \log(n))$.

Finally, the bound on the query time consists of 4 queries to the the average case data structure DS , as well as the computation of $\langle A_i, v \rangle$. The latter can be done by reading the description of the t -sparse vector v , and computing their inner product with the corresponding t entries in the i th row of A . Hence the total query time is $4t + O(\log^4(1/\alpha) \cdot \log(n))$.

Correctness: By [Lemma 3.3.1](#) we have

$$x = x_1 + x_2 - (x_3 + x_4) + v ,$$

where $x_1, x_2, x_3, x_4 \in X$. By the definition of X , this implies that the average-case data structure DS computes these points correctly, hence

$$\text{DS}_{x_1}(i) + \text{DS}_{x_2}(i) - \text{DS}_{x_3}(i) - \text{DS}_{x_4}(i) = \langle A_i, x_1 \rangle + \langle A_i, x_2 \rangle - \langle A_i, x_3 \rangle - \langle A_i, x_4 \rangle ,$$

Furthermore, we directly compute $\langle A_i, v \rangle$, and hence, by the linearity of the inner product operation, it follows that

$$\langle A_i, x \rangle = \sum_{j=1}^4 \langle A_i, x_j \rangle + \langle A_i, v \rangle .$$

This concludes the proof of [Theorem 2](#). □

4.3.2 Weak-average-case data structures

In [Section 4.3.1](#), we showed a worst-case to average-case reduction for all linear problems in the setting of data structures. In the following, we show how to obtain worst-case algorithms starting from a very weak, but natural, notion of average-case reductions that we discuss next.

Recall that in the standard definition of average-case data structures, the algorithm preprocesses its input and is then required to correctly answer all queries for an α -fraction of all possible inputs. However, in many cases (such as in the online matrix-vector multiplication problem), we only have an average-case guarantee on both inputs and queries. In this setting, we should first ask what is a natural notion of an average-case condition.

A strong requirement for an average-case algorithm in this case is to correctly answer *all queries* for at least α -fraction of the inputs. However, a more desirable condition is to require the algorithm to correctly answer on an *average input and query*. This is captured by the following definition.

Definition 4.3.2. A weak average-case data structure for computing a function $f: \mathbb{F}^n \times Q \rightarrow \mathbb{F}^k$ with success rate $\alpha > 0$ receives an input $x \in \mathbb{F}^n$, which is preprocessed into s memory cells. Then, given a query $q \in Q$, the data structure $\text{DS}_x(q)$ outputs $y \in \mathbb{F}^k$ such that

$$\Pr_{x \in \mathbb{F}^n, q \in Q} [\text{DS}_x(q) = f(x, q)] \geq \alpha .$$

The challenge in this setting is that the errors may be distributed between both the inputs and the queries. On one extreme, the error could be concentrated on selected inputs, and then the data structure computes *all queries* correctly for α -fraction of the inputs. On the other extreme, the error could be spread over all inputs, and then the data structure may only answer α -fraction of the queries on any inputs. Of course, the error could be distributed anywhere in between these two extremes.

Weak-average-case reductions for matrix-vector multiplication. As a first example of the weak-average-case paradigm, we note that our reduction for the online matrix-vector multiplication problem in [Section 4.2](#) can be cast as a worst-case to weak-average-case reduction. Namely, we start with a data structure that receives a matrix $M \in \mathbb{F}^{n \times n}$ as an input, preprocesses it into s memory cells. Then, on query $v \in \mathbb{F}^n$ the data structure algorithm DS_M satisfies

$$\Pr_{M \in \mathbb{F}^{n \times n}, v \in \mathbb{F}^n} [\text{DS}_M(v) = Mv] \geq \alpha.$$

Hence we immediately obtain the following statement.

Theorem 3. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Consider the matrix-vector multiplication problem $\text{OMV}_{\mathbb{F}}$ for dimension n , and suppose that for some $\alpha > 0$ it holds that*

$$\text{OMV}_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{M,v} [\text{DS}_M(v) = Mv] \geq \alpha \end{array} \right].$$

Then for every $\delta > 0$,

$$\text{OMV}_{\mathbb{F}} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & 4p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha)n) + O(n^2) \\ \text{query time:} & (4t + n) \cdot \text{poly}(1/\alpha) \cdot \text{poly} \log(1/\delta) \\ \text{success rate:} & \forall M, v : \Pr[\text{DS}_M(v) = Mv] \geq 1 - \delta \end{array} \right].$$

An immediate question is whether it is possible to obtain worst-case to weak-average-case reductions not only for the matrix-vector multiplication problem, but rather for *all* linear problems, as we have in the setting of (standard) average-case data structure. Alas, as we show next, such a general result is impossible.

4.3.3 Impossibility of weak-average-case reductions for all linear problems

We observe that for weak-average-case data structures, there is a simple counterexample which shows that it is *impossible* to obtain worst-case to weak-average-case reductions for *all* linear problems. Nevertheless, we later show that it is possible to obtain such reductions for specific natural problems beyond matrix-vector multiplication, namely for the (non-linear) problem of multivariate polynomial evaluation.

To see the counterexample, first note that a weak-average-case data structure can be equivalently thought of as a data structure where the answer to each input is a vector, rather than a scalar, and the requirement is that the algorithms on average outputs a partially correct vector. That is, a weak-average-case data structure computes a function $f: \mathbb{F}^n \rightarrow \mathbb{F}^m$ with success rate $\alpha > 0$ if after the preprocessing, on query $x \in \mathbb{F}^n$ it satisfies that $\Pr_{x \in \mathbb{F}^n, i \in [m]}[\text{DS}(x)_i = f(x)_i] \geq \alpha$.

Weak-average-case circuits. For simplicity, we start with a counterexample for weak-average-case circuits, then extend it to the setting of data structures. Note that the number of linear functions $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ is 2^{nm} . The total number of (not necessarily linear) circuits with g gates is $2^{O(g \log(g))}$ (see, e.g., Lemma 1.12 in [Juk12]). Therefore, by a simple counting argument, a random *linear* function requires a circuit with $g \geq \Omega\left(\frac{mn}{\log(mn)}\right)$ gates.

Now fix a linear function f of complexity at least $\Omega\left(\frac{mn}{\log(mn)}\right)$, and an arbitrarily small constant $\varepsilon > 0$. Let us consider the function $h: \mathbb{F}_2^n \rightarrow \mathbb{F}_2^{m/\varepsilon}$ that embeds f as follows: the first m outputs of $h(x)$ compute $f(x) \in \mathbb{F}^m$, and the remaining $m/\varepsilon - m$ outputs are always zeros. Note that h is also a linear function that requires a circuit with at least $\Omega\left(\frac{mn}{\log(mn)}\right)$ gates.

On the other hand, note that the trivial circuit outputting m/ε zeros well-approximates the function h , i.e., it satisfies the weak-average-case with success rate $\alpha = (1 - \varepsilon)$. Thus, any worst-case to average-case reduction for all functions in this setting would have to blow up the size of the trivial circuit computing 0 to the size of at least $\Omega\left(\frac{mn}{\log(mn)}\right)$, which is almost the biggest circuit with this given number of inputs and outputs. Therefore, such a reduction would be degenerate.

Weak-average-case data structures. Moving on to the setting of data structures, here there is an issue with such an argument. To see that, recall that we want start by picking a

linear function that is hard even against *non-linear* data structures. While the number of linear functions is still 2^{mn} , a data structure can compute s *arbitrary* (i.e., not necessarily linear) functions in the preprocessing stage. The problem is that even one such function gives a data structure 2^{2^n} possibilities which is already larger than the number of *linear* functions, and so the counting argument here doesn't work.

Nevertheless, we can still get essentially the same result for data structures by the following argument. Let $C(n, m)$ be the complexity of the hardest data structure for a *linear* problem. Then, again, we take the hardest linear function from n bits to m bits, and extend it to a function with m/ε output bits (where $m/\varepsilon - m$ outputs are constant zeros). The worst-case complexity of this function is at least $C(n, m)$, while the average-case complexity is 0. Hence, every worst-case to average-case reduction will blow up the size from 0 to $C(n, m)$. Since every linear function can be computed by a data structure of size $C(n, m/\varepsilon)$ without any reduction, such a reduction is also degenerate.

4.3.4 Weak-average-case reductions for multivariate polynomial evaluation

Our main result in the weak-average-case setting is a worst-case to weak-average-case reduction for data structures computing the (non-linear) problem of multivariate polynomial evaluation. In this problem, the input is a polynomial $q: \mathbb{F}^m \rightarrow \mathbb{F}$ of total degree at most d , given as its coefficients. That is, the length of the input is $n = \binom{m+d}{d}$. Given the input polynomial, it is preprocessed, and then in the query phase the goal is to respond to each query $x \in \mathbb{F}^m$ with the value $q(x)$. We restate and prove [Theorem 4](#) below.

Theorem 4. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $\alpha := \alpha(n) \in (0, 1]$, and let $m, d \in \mathbb{N}$ be parameters. Consider the problem $\text{RM}_{\mathbb{F}, m, d}$ of evaluating polynomials of the form $q: \mathbb{F}^m \rightarrow \mathbb{F}$ of total degree d (i.e., the problem of evaluating the Reed-Muller encoding of block length $n = \binom{m+d}{d}$).*

Suppose that

$$\text{RM}_{\mathbb{F}, m, d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{q, x}[\text{DS}_q(x)] \geq \alpha \end{array} \right].$$

Then

$$\text{RM}_{\mathbb{F},m,d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p + \exp(\log^4(1/\alpha)) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha) \log(n)) \\ \text{query time:} & O(|\mathbb{F}|^2 \cdot t + |\mathbb{F}| \log^4(1/\alpha) + |\mathbb{F}| \log(n)) \\ \text{success rate:} & \forall q, x : \Pr[\text{DS}_q(x) = q(x)] > 1 - O\left(\sqrt{\frac{d}{|\mathbb{F}|}}\right) \end{array} \right].$$

[Theorem 4](#) follows from the following two lemmas.

Lemma 4.3.3. *Let $\mathbb{F}$ be a prime field, and $d \leq |\mathbb{F}|/10$. Suppose that for $\alpha > 2\sqrt{\frac{d}{|\mathbb{F}|}}$ we have*

$$\text{RM}_{\mathbb{F},m,d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \Pr_{q,x}[\text{DS}_q(x) = q(x)] \geq \alpha \end{array} \right].$$

Then

$$\text{RM}_{\mathbb{F},m,d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & 4p + \exp(\log^4(1/\alpha)) \cdot \text{poly} \log(1/\delta) \cdot \text{poly}(n) \\ \text{memory used:} & 4s + O(\log^4(1/\alpha)) \\ \text{query time:} & 4|\mathbb{F}| \cdot t + O(\log^4(1/\alpha)) + O(\log(n)) \\ \text{success rate:} & \forall q \text{ with } \deg(q) \leq d: \Pr_x[\text{DS}_q(x) = q(x)] \geq 1 - O\left(\sqrt{\frac{d}{\alpha \cdot |\mathbb{F}|}}\right) \end{array} \right].$$

Lemma 4.3.4. *Let $\mathbb{F}$ be a prime field, and $d \leq |\mathbb{F}|/10$. Suppose that for $\gamma < 0.1$ we have*

$$\text{RM}_{\mathbb{F},m,d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & t \\ \text{success rate:} & \forall q \text{ with } \deg(q) \leq d: \Pr_x[\text{DS}_q(x) = q(x)] \geq 1 - \gamma \end{array} \right].$$

Then

$$\text{RM}_{\mathbb{F},m,d} \in \text{DS} \left[\begin{array}{ll} \text{preprocessing time:} & p \\ \text{memory used:} & s \\ \text{query time:} & |\mathbb{F}| \cdot t \\ \text{success rate:} & \forall q \text{ with } \deg(q) \leq d, \forall x \in \mathbb{F}^n: \Pr[\text{DS}_q(x) = q(x)] \geq 1 - 4\gamma \end{array} \right].$$

Before proceeding with the proofs of the lemmas above, we will need the following proposition.

Proposition 4.3.5. *Let $\mathbb{F}$ be a prime field, $d \leq |\mathbb{F}|/10$, and let $\alpha > 2\sqrt{\frac{d}{|\mathbb{F}|}}$. Let $n = \binom{d+m}{m}$ be the input length—the number of coefficients in a polynomial $q: \mathbb{F}^m \rightarrow \mathbb{F}$ of total degree at most d ,*

Let DS be a data structure for $\text{RM}_{\mathbb{F},m,d}$ with preprocessing time p , that stores s field elements, and has query time t . Then there exists another data structure DS' for $\text{RM}_{\mathbb{F},m,d}$ with preprocessing time $p+n$, that stores $s+m+1$ field elements, has query time $|\mathbb{F}|t$, and satisfies the following guarantee for all input polynomials q of degree at most d .

$$\text{If } \Pr_x[\text{DS}_q(x) = q(x)] \geq \alpha, \text{ then } \Pr_x[\text{DS}'_q(x) = q(x)] \geq 1 - \sqrt{\frac{d}{\alpha \cdot |\mathbb{F}|}}.$$

We emphasize that in the claim above the data structures do not depend on the polynomial q . The proposition says that if q is an input such that DS outputs the correct evaluation $q(x)$ for at least α fraction of the queries x , then DS' (which also does not depend on any particular q) succeeds on $1 - \sqrt{\frac{d}{\alpha \cdot |\mathbb{F}|}}$ fraction of the same input q .

Proof of Proposition 4.3.5. Denote by DS the data structure for $\text{RM}_{\mathbb{F},m,d}$ that outputs the correct answer for at least α fraction of inputs to q . Below we describe the data structure DS' .

Data Structure DS'

Preprocessing: Given the polynomial q of degree at most d

1. Run the preprocessing procedure for DS on the input q .
2. Choose a random *reference point* $\vec{w} \in \mathbb{F}^m$ and compute $q(\vec{w})$.
3. Store $\vec{w}$ and $q(\vec{w})$ in the memory.

Query: Given a query $\vec{x} \in \mathbb{F}^m$

1. Consider the line $\ell_{\vec{x}, \vec{w}} = \{\vec{x} + r(\vec{w} - \vec{x}) : r \in \mathbb{F}\}$ going through $\vec{x}$ and $\vec{w}$.
2. Use the query algorithm of DS to compute $(DS_q(z) : z \in \ell_{\vec{x}, \vec{w}})$.
3. Let $Q = \{q_1, q_2, \dots, q_k\}$ be all the univariate polynomials of degree at most d that agree with $(DS_q(z) : z \in \ell_{\vec{x}, \vec{w}})$ on at least $\alpha/2$ fraction of points in $\ell_{\vec{x}, \vec{w}}$. (It is possible that $Q = \emptyset$.)
4. Use the value $q(\vec{w})$ from the preprocessing phase, and let $Q' = \{q' \in Q : q'(\vec{w}) = q(\vec{w})\}$. (It is possible that $Q' = \emptyset$.)
5. Choose $q' \in Q'$ arbitrarily and output $q'(\vec{x})$.

Next, we claim that if $\alpha > 2\sqrt{\frac{d}{|\mathbb{F}|}}$, then for at least $1 - O\left(\sqrt{\frac{d}{|\mathbb{F}|\alpha}}\right)$ fraction of the queries $\vec{x}$ the query phase correctly outputs $q(\vec{x})$.

It will be convenient to consider a function $A: \mathbb{F}^m \rightarrow \mathbb{F}$ defined as $A(z) = DS_q(z)$ for all $z \in \mathbb{F}^n$. Note that A agrees with q on at least α -fraction of points. Furthermore, note that for simplicity we may assume that q is the all zeros polynomial. Indeed, we can define $A'(x) := A(x) - q(x)$, and consider the case where the input is the all zeros polynomial, and the query algorithm is A' . Therefore, (1) we have $\Pr_{x \in \mathbb{F}^m}[A(x) = 0] \geq \alpha$, and (2) in the preprocessing phase we know that $q(\vec{w}) = 0$ for a random point $\vec{w}$, though it is not necessarily true that $A(\vec{w}) = 0$.

The following three claims complete the proof of [Proposition 4.3.5](#).

Claim 4.3.6. *In the preprocessing phase, for a random choice of the reference point $\vec{w}$ with high probability over $\vec{x}$ it holds that $\vec{0} \in Q$, and hence in Q' . More formally, we have*

$$\mathbb{E}_{\vec{w} \in \mathbb{F}^m} [\Pr_x[\vec{0} \in Q]] \geq 1 - \frac{4}{|\mathbb{F}|\alpha}.$$

In particular, by Markov's inequality, for at least $1 - \sqrt{\frac{4}{|\mathbb{F}|}\alpha}$ of $\vec{w}$'s it holds that

$$\Pr_x[\vec{0} \in Q] \geq 1 - \sqrt{\frac{4}{|\mathbb{F}|\alpha}}. \quad (4.1)$$

Proof. It is a standard fact in the literature on derandomization (see, e.g., [MR06, Corollary 1.2]) that for any set $O \subseteq \mathbb{F}^n$ of size $|O| = \alpha|\mathbb{F}|^n$, and a random line $\ell_{\vec{x}, \vec{w}} = \{\vec{x} + t(\vec{w} - \vec{x}) : t \in \mathbb{F}\}$ that passes through uniformly random $\vec{x}, \vec{w} \in \mathbb{F}^n$ it holds that

$$\Pr \left[\left| \frac{|\ell_{\vec{x}, \vec{w}} \cap O|}{|\ell_{\vec{x}, \vec{w}}|} - \alpha \right| > \varepsilon \right] \leq \frac{1}{|\mathbb{F}|} \frac{\alpha}{\varepsilon^2}.$$

The conclusion of the claim follows by letting $O = \{x \in \mathbb{F}^n : A(x) = 0\}$, and $\varepsilon = \alpha/2$. $\square$

Claim 4.3.7 ([MR06, Proposition 3.5]). *Choose $\vec{w}$ and $\vec{x}$ uniformly at random and consider the line $\ell_{\vec{x}, \vec{w}}$. Let $Q = \{q_1, q_2, \dots, q_k\}$ be all the univariate polynomials of degree at most d that agree with A on at least $\alpha/2$ fraction of points in $\ell_{\vec{x}, \vec{w}}$. If $\alpha > 2\sqrt{\frac{d}{|\mathbb{F}|}}$, then $k \leq 2/\alpha$.*

Claim 4.3.8. *Choose $\vec{w}$ and $\vec{x}$ uniformly at random and consider the line $\ell_{\vec{x}, \vec{w}}$. Let $Q = \{q_1, q_2, \dots, q_k\}$ be all the univariate polynomials of degree at most d that agree with A on at least $\alpha/2$ fraction of points in $\ell_{\vec{x}, \vec{w}}$. Then, for all $q_i \in Q$ that are not identically zero it holds that $\Pr[q_i(\vec{w}) = 0] \leq \frac{d}{|\mathbb{F}|}$.*

In particular, if $\alpha > 2\sqrt{\frac{d}{|\mathbb{F}|}}$ then

$$\Pr_{\vec{w}} \left[\Pr_{\vec{x}} [\forall q_i \in Q \setminus \{\vec{0}\} : q_i(\vec{w}) \neq 0] \geq 1 - \sqrt{\frac{2d}{\alpha|\mathbb{F}|}} \right] \geq 1 - \sqrt{\frac{2d}{\alpha|\mathbb{F}|}}.$$

Proof. For any choice of $\vec{x}$ if $\vec{w}$ is chosen uniformly at random, and each univariate polynomial q_i is of degree d , then by Schwarz-Zippel lemma

$$\Pr[q_i(\vec{w}) = 0] \leq \frac{d}{|\mathbb{F}|}.$$

Also, by Claim 4.3.7, we know that $|Q| = k \leq 2/\alpha$, and hence, by union bound

$$\Pr_{\vec{x}, \vec{w}} \left[\exists q_i \in Q \setminus \{\vec{0}\} : q_i(\vec{w}) = 0 \right] \leq \frac{kd}{|\mathbb{F}|} \leq \frac{2d}{\alpha|\mathbb{F}|}.$$

This implies

$$\mathbb{E}_{\vec{w}} \left[\Pr_{\vec{x}} [\forall q_i \in Q \setminus \{\vec{0}\} : q_i(\vec{w}) \neq 0] \right] \geq 1 - \frac{kd}{|\mathbb{F}|} \geq 1 - \frac{2d}{\alpha|\mathbb{F}|} .$$

The claim follows by Markov's inequality. $\square$

We now return to the proof of [Proposition 4.3.5](#). By the claims above, for most $\vec{w}$'s it holds that if we choose $\vec{w}$ as a reference point, then for most $\vec{x}$'s, we have $\vec{0} \in Q$, and there is no other univariate polynomial q_i in Q such that $q_i(\vec{w}) = 0$. More precisely, by combining [Claim 4.3.6](#) with [Claim 4.3.8](#) for at least $1 - O\left(\sqrt{\frac{d}{\alpha|\mathbb{F}|}}\right)$ fraction of $\vec{w}$'s it holds that

$$\Pr_x [\vec{0} \in Q \wedge \forall q \in Q : q \not\equiv 0, q(\vec{w}) \neq 0] \geq 1 - O\left(\sqrt{\frac{d}{\alpha|\mathbb{F}|}}\right) , \quad (4.2)$$

as required. $\square$

Now we proceed with proving [Lemma 4.3.3](#).

Proof of [Lemma 4.3.3](#). Suppose there is a data structure DS as in the assumption of [Lemma 4.3.3](#), with success probability $\Pr_{q,x}[\text{DS}_q(x) = q(x)] \geq \alpha$. We show below how to construct a data structure DS' that will work for all input polynomial q and for most queries x .

Preprocessing:

Input: An input polynomial q of degree at most d

1. Identify q with a vector $q \in \mathbb{F}^n$ of its coefficients for $n = \binom{m+d}{d}$.
2. Let $Z = \{q \in \mathbb{F}^n : |X_q| \geq \frac{\alpha}{2} \cdot |\mathbb{F}|^n\}$.
3. Let O_Z be a membership oracle for Z that given a polynomial q' estimates $\frac{|X_{q'}|}{|\mathbb{F}|^n}$, the fraction of points on which $\text{DS}_{q'}$ outputs $q'(x)$ correctly, within an additive error of $\alpha/10$, and returns *ACCEPT* if and only if the estimated fraction is more than $\alpha/3$.^a
4. By applying [Lemma 3.3.1](#), with probability $1 - \delta$ we obtain a vector u with at most $O(\log^4(1/\alpha))$ non-zero elements such that

$$\Pr_{q_1, q_2, q_3 \in \mathbb{F}^n} [q_1, q_2, -q_3, -q_4 \in Z] \geq \Omega(\alpha^5),$$

where $q_4 \in \mathbb{F}^n$ is such that $q - u = q_1 + q_2 - q_3 - q_4$.

5. Therefore, given q and s we can sample $O(\log(1/\delta) \cdot \log^4(1/\alpha))$ triplets of vectors until we find a triplet (q_1, q_2, q_3) and let $q_4 = q_1 + q_2 - q_3 - q - u$ satisfying $q_1, q_2 \in Z, -q_3, -q_4 \in Z$. Note that we can use the membership oracle O_Z to check that the vectors belong to Z .
6. Note that since each v_i belongs to Z , we have that DS outputs $p_i(x)$ correctly on at least $\alpha/4$ fraction of inputs. Thus, we can apply [Proposition 4.3.5](#) on DS and obtain the data structure DS' such that $\Pr_{x \in \mathbb{F}^n} [\text{DS}'_{q_i}(x) = q_i(x)] \geq 1 - O\left(\sqrt{\frac{d}{\alpha \cdot |\mathbb{F}|}}\right)$ for all $i = 1, 2$, and $\Pr_{x \in \mathbb{F}^n} [\text{DS}'_{-q_j}(x) = -q_j(x)] \geq 1 - O\left(\sqrt{\frac{d}{\alpha \cdot |\mathbb{F}|}}\right)$ for $j = 3, 4$.
7. We store all the memory obtained by preprocessing the polynomials q_1, q_2, q_3, q_4 with DS . We also store the sparse vector s by storing the $O(\log^4(1/\alpha))$ non-zero coordinates, and their values.

^aThis is done by sampling $O(1/\alpha^2)$ uniformly random x 's in $\mathbb{F}^n$, computing $\text{DS}_{q'}(x)$ and $q'(x)$, and comparing the two results. In particular, if $|X_{q'}| \geq \frac{\alpha}{2} \cdot |\mathbb{F}|^n$, then $O_Z(q') = \text{ACCEPT}$ with probability $1 - \varepsilon$, and if $|X_{q'}| \leq \frac{\alpha}{4} \cdot |\mathbb{F}|^n$, then $O_Z(q') = \text{REJECT}$ with probability $1 - \varepsilon$.

For every polynomial q of degree at most d , let $X_q = \{x \in \mathbb{F}^m : \text{DS}_q(x) = q(x)\}$.

By averaging, there is a set Z of degree d polynomials such that $|Z| \geq \alpha/2 \cdot |\mathbb{F}|^n$, and $|X_q| \geq \alpha/2 \cdot |\mathbb{F}|^m$ for every $q \in Z$. Furthermore, note that it is straightforward to construct a membership oracle O_Z for Z , that given a polynomial q and access to DS_q estimates the fraction of queries x on which $\text{DS}_q(x) = q(x)$.

Below we describe the preprocessing phase and the query phase of DS' .

The preprocessing on an input q works as follows. By identifying q with the vector of its coefficients in $\mathbb{F}^n$ with $n = \binom{m+d}{m}$, we use [Lemma 3.3.1](#) to represent the vector q as $q = q_1 + q_2 - q_3 - q_4 + u$, where each $q_i \in Z$ and u is a sparse vector. Then, for each q_i we the reduction from [Proposition 4.3.5](#) to obtain a data structure that works for each of the q_i for almost all queries x .

In the query phase, we use the data structures for each q_i to compute $q_i(x)$, and for the sparse polynomial s , we simply compute $u(x)$ using brute force. Finally, we return $q_1(x) + q_2(x) - q_3(x) - q_4(x) + u(x)$.

Preprocessing time and space: In the preprocessing step, we store the memory of the preprocessing for each of q_i and in addition the sparse vector u . Hence, the total space used is $4s + \log^4(1/\alpha)$ field elements + additional $\log^4(1/\alpha)$ coordinates of the input. Also, the running time is determined by number of samples needed to construct the oracle in Step 1 using [Lemma 3.3.1](#). Both these steps take at most $O(\log^4(1/\alpha) \cdot \log(1/\delta))$ samples, bounding the running time of the preprocessing step.

Next we describe the query phase of our data structure.

Query phase:

Input: A query $\vec{x} \in \mathbb{F}^m$.

Recall that for the polynomial q , we have stored high-agreement data structures for evaluating q_1, q_2, q_3, q_4 , together with a polynomial which is represented by a sparse vector of coefficient u .

1. For each polynomial q_i let $y_i = \text{DS}'_{q_i}(\vec{x})$.
2. Compute $u(x)$. Since u has at most $O(\log^4(1/\alpha))$ non-zero coordinates, it follows that $u(x)$ can be computed in query time $O(\log^4(1/\alpha) \log(n))$.
3. Return $y_1 + y_2 - y_3 - y_4 + u(x)$.

Query time: The query time consists of querying the data structure 4 times, and evaluating $u(x)$. Note that the high-agreement data structure makes $|\mathbb{F}|$ queries to the weak-average-case data structure, and each query takes time t . Thus, the total query time is $4|\mathbb{F}| \cdot t + O(\log^4(1/\alpha) \log(n))$.

Correctness: To prove the correctness, we bound the failure probability of the algorithm. Note that the algorithm returns the correct answer, unless for one of the polynomials q_i it holds that, $\text{DS}'_{q_i}(\vec{x}) \neq q_i(\vec{x})$. This event happens with probability at most $O(\sqrt{\frac{d}{|\mathbb{F}|\alpha}})$. Hence, by applying union bound we can bound the failure probability to $O(\sqrt{\frac{d}{|\mathbb{F}|\alpha}})$. $\square$

Finally, we prove [Lemma 4.3.4](#).

Proof of Lemma 4.3.4. The proof of this lemma basically relies on the local decoding algorithm for Reed-Muller codes. Given a point $\vec{x}$, the query algorithm samples a random line $\ell_{x,y} = \{\vec{x} + r(\vec{y} - \vec{x}) : r \in \mathbb{F}\}$ passing through $\vec{x}$, and queries the data structure for all the points on this line. Given these values, the algorithm finds the closest univariate polynomial of degree at most d , call it h , and outputs $h(\vec{x})$. It is not hard to see that the algorithm succeeds with probability at least $1 - 4\gamma$.

For correctness since DS_p agrees with q on $1 - \gamma$ fraction of the points $z \in \mathbb{F}^n$, it follows that for a random line ℓ through x the data structure DS satisfies $\Pr[\text{agr} \geq 3/4] \geq 1 - 4\gamma$, where agr denotes the fraction of points $z \in \ell$ with $\text{DS}_{q_i}(z) = q_i(z)$. For each such line ℓ the only polynomial that agrees with DS on ℓ is q_i , and hence with probability at least $1 - 4\gamma$ the data structure outputs $q_i(x)$, as required. $\square$

Putting it all together: Below we summarize the reductions above, and describe the full reduction that given a weak-average-case data structure DS that computes the correct answer for only α fraction of (q, x) , gives us a data structure that works with high probability for all inputs q and all queries x .

We first use [Lemma 4.3.4](#), reducing the problem to evaluating p on a random line ℓ passing through $\vec{x}$. Then, we apply [Lemma 4.3.3](#), to write p as sum of 5 polynomials, for which we know one of them is sparse and can be computed efficiently, and the other four belong to the set of *good* polynomials, i.e., those polynomials for which the data structure succeeds in evaluating them on all but a small fraction of inputs. Finally, for each of these polynomials we apply the reduction in [Proposition 4.3.5](#). This last step corresponds to choosing a random reference point $\vec{w} \in \mathbb{F}^n$, and passing lines between every $\vec{z} \in \ell$ and $\vec{w}$, and evaluating each of the q_i on each of the $\mathbb{F}$ lines.

Chapter 5

Quantum Reductions

In this chapter, we formally state and prove our quantum worst-case to average-case reduction for linear problems. We will start the chapter by providing the necessary background on quantum computation and algorithms.

5.1 Basics of quantum computation

We refer the reader to standard textbooks such as [NC10] for preliminaries and notation regarding quantum algorithms.

5.1.1 Quantum unitary oracles and Fourier transforms

We start by providing basic notation and definitions regarding discrete and quantum Fourier transforms, as well as quantum oracles.

Discrete and quantum Fourier transforms. Let $\mathbb{F} = \mathbb{F}_p$ be the prime field of size p . For a function $f : \mathbb{F}^n \rightarrow \mathbb{R}$, the Fourier coefficient $\hat{f}(y)$ for $y \in \mathbb{F}^n$ representing characters χ_y is given by

$$\hat{f}(y) = \frac{1}{p^n} \sum_{x \in \mathbb{F}^n} \omega^{x \cdot y} f(x) , \tag{5.1}$$

where $\omega = e^{2\pi i/p}$ is a primitive p^{th} root of unity.

Denote the quantum Fourier transform over $\mathbb{F}$ by QFT_p , having the action

$$\text{QFT}_p |x\rangle = \frac{1}{\sqrt{p}} \sum_{y \in \mathbb{F}_p} \omega^{x \cdot y} |y\rangle .$$

Quantum unitary oracles. We let quantum algorithms access a matrix $M \in \mathbb{F}^{n \times n}$ via a unitary oracle U_M that performs the map

$$U_M |j, k, z\rangle = |j, k, z \oplus M_{jk}\rangle , \quad (5.2)$$

for all indices $j, k \in [n]$ and $z \in \mathbb{F}$, where $\oplus$ denotes addition over $\mathbb{F}$. For vectors $v \in \mathbb{F}^n$, oracle access means a unitary U_v that returns components of the vector when queried with an index, analogously to Eq. (5.2), i.e.,

$$U_v |j, z\rangle = |j, z \oplus v_j\rangle , \quad (5.3)$$

for all $j \in [n]$ and $z \in \mathbb{F}$. Unless otherwise stated, whenever we assume access to a unitary U as an oracle we also assume access to $U^\dagger$.

The Matrix-Vector Multiplication problem. In this setting we consider algorithms that have access to a matrix M , and are given a vector v as input.

MATRIX-VECTOR MULTIPLICATION (MvM)

Input: Oracle access to a matrix $M \in \mathbb{F}^{n \times n}$ and a vector $v \in \mathbb{F}^n$.

Output: The matrix-vector product Mv .

Linear Problems. A linear problem is characterized by a family of matrices $\mathcal{M} := \{M_n \in \mathbb{F}^{n \times n}\}_{n \in \mathbb{N}}$, where on input $v \in \mathbb{F}^n$ the solution to the problem is the vector Mv , omitting the subscript on M for readability.

LINEAR PROBLEM $\mathcal{M}$

Input: A vector $v \in \mathbb{F}^n$.

Output: The matrix-vector product Mv .

This notion captures a wide variety of problems, including such fundamental ones as computing Discrete Fourier transforms, and polynomial evaluation (Vandermonde matrices). As discussed in the introduction, in this work, we show that for *every* linear problem $\mathcal{M}$ there exists an efficient quantum worst-case to average-case reduction, a result for which no analogue is known in the case of uniform classical algorithms.

Next, we define the central notion that we study in this and the following sections, namely the average-case behaviour of quantum algorithms.

5.1.2 Average-case quantum algorithms

An average-case quantum algorithm $\mathbf{ALG}$ is one that succeeds with probability at least α in expectation over (uniformly) random inputs. That is, if $\mathbf{ALG}$ is to compute some function f mapping some known measurable domain V to some co-domain W , then

$$\Pr_{\substack{\mathbf{ALG} \\ v \in V}}[\mathbf{ALG}(v) = f(v)] := \mathbb{E}_{v \in V} [|\Pi_{f(v)} \mathbf{ALG}(v)|^2] \geq \alpha . \quad (5.4)$$

Here $\Pr_{\mathbf{ALG}}$ denotes the probability over the internal (quantum) randomness of the algorithm arising from its unitary nature and final measurements. Note that the probability above is taken over the inputs *as well as* the internal quantum randomness of the algorithm. This is highlighted by the notation we use, which we elaborate more on below.

This notion of average-case quantum algorithms immediately suggests considering the following natural modification of quantum oracles.

Noisy quantum oracles. For a function $f : \mathbb{F}^n \rightarrow \mathbb{F}^m$, we can consider a unitary oracle U_f which on input $x \in \mathbb{F}^n, z \in \mathbb{F}^m$ performs the map

$$U_f |x\rangle |z\rangle = \beta_{\text{succ}}^x |x\rangle |z + f(x)\rangle + \beta_{\text{fail}}^x |x\rangle |\psi(x)\rangle , \quad (5.5)$$

where $\beta_{\text{succ}}^x, \beta_{\text{fail}}^x \in \mathbb{C}$ such that $|\beta_{\text{succ}}^x|^2 + |\beta_{\text{fail}}^x|^2 = 1$, the normalized state $|\psi(x)\rangle$ could be an arbitrary superposition over $|z + v\rangle$ for vectors $v \in \mathbb{F}^m \setminus \{f(x)\}$, and $+$ denotes component-wise addition for vectors over $\mathbb{F}$. We can interpret U_f as a quantum analogue of a classical probabilistic algorithm for computing f — for an input $x \in \mathbb{F}^n$, it outputs the correct value of $f(x) \in \mathbb{F}^m$ with probability $|\beta_{\text{succ}}^x|^2$ when the second register is measured in the computational basis.

More generally, we consider oracles that may entangle a workspace register with the output

$$U_f |x\rangle |w\rangle |z\rangle = \beta_{\text{succ}}^x |x\rangle |z + f(x)\rangle |w(x, z, f(x))\rangle + \beta_{\text{fail}}^x |x\rangle |\Psi(x)\rangle , \quad (5.6)$$

where $|\Psi(x)\rangle$ is now a normalized state of the form

$$|\Psi(x)\rangle = \sum_{\substack{v \in \mathbb{F}^n \\ v \neq f(x)}} \gamma_v^x |z + v\rangle |w(x, z, v)\rangle .$$

Quantum algorithms for linear problems. A quantum algorithm for a linear problem $\mathcal{M}$ outputs the correct answer with some probability arising from measurement. Such an algorithm is represented by a unitary **ALG** which on an input vector $v \in \mathbb{F}^n$ has the action

$$\text{ALG} |v\rangle |0\rangle = \beta_{\text{succ}}^v |v\rangle |Mv\rangle |w_0(v)\rangle + \beta_{\text{fail}}^v |v\rangle |\Psi(v)\rangle , \quad (5.7)$$

where $\beta_{\text{succ}}^v, \beta_{\text{fail}}^v \in \mathbb{C}$ are complex amplitudes such that $|\beta_{\text{succ}}^v|^2 + |\beta_{\text{fail}}^v|^2 = 1$, $|w_0(v)\rangle$ is an arbitrary normalized state of a workspace register, and

$$|\beta_{\text{succ}}^v|^2 = \|\mathbb{1} \otimes \Pi_{Mv} \text{ALG} |v\rangle |0\rangle\|^2$$

is the success probability of the algorithm, where $\Pi_{Mv} := |Mv\rangle\langle Mv|$ is an orthogonal projection on to the correct answer subspace of the output register. $|\Psi(v)\rangle$ is a normalized state that is orthogonal to the state $|Mv\rangle$ that encodes the correct matrix vector product, i.e. $\forall v \in \mathbb{F}^n, \langle Mv | \Psi(v)\rangle = 0$. In general, it takes the form

$$|\Psi(v)\rangle = \sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v |z\rangle |w(v, z)\rangle .$$

5.1.3 Quantum singular value transformation techniques

In this section, we provide self-contained statements of the quantum singular value transformations (QSVT) based techniques that we make use of.

Fixed-point amplitude amplification An issue that we frequently face while using standard amplitude amplification is the inability to *a priori* estimate the number of iterations to perform without overshooting the target state. When the goal is to output a classical solution to a search problem, one can sidestep this issue by iterating over the

number of rounds of amplification with a multiplicative step size (often called exponential search). But if the goal of amplification is to obtain a unitary subroutine that can later be composed with another unitary, this problem can be addressed by the technique of fixed-point amplitude amplification, which generalizes the notion of fixed-point search [Gro05] and converges monotonically towards the marked state as the number of iterations is increased. We use this machinery primarily in the proof of Lemma 5.3.1.

The precise version of fixed-point amplitude amplification that we employ is expressed in the framework of quantum singular value transformations [GSLW19, Theorem 27, arXiv version], as follows.

Theorem 5.1.1 (Fixed-point amplitude amplification). *Let U be a unitary acting on $k+1$ qubits and Π an orthogonal projector. If for an input state $|\phi_{\text{in}}\rangle$ and $p > \delta > 0$ we have $\Pi U |\phi_{\text{in}}\rangle = p |\phi_{\text{tar}}\rangle$, then for every $\epsilon > 0$ there is a unitary U_q such that $\| |\phi_{\text{tar}}\rangle - U_q |\phi_{\text{in}}\rangle \| \leq \epsilon$. This U_q requires $q = O(\frac{1}{\delta} \log \frac{1}{\epsilon})$ uses of U and $U^\dagger$, one ancillary qubit, and $O(q)$ additional one and two-qubit gates.*

Singular value threshold projections To obtain our unitary implementation of the (noisy) indicator function in Section 5.3.2, we need two conditions to be simultaneously satisfied: (1) the singular values that are larger than a threshold t are boosted close to unity, and (2) the singular values that are below the threshold are suppressed close to zero. In particular, fixed-point amplitude amplification only guarantees the former condition and is not the right tool for the task. We hence invoke a more sophisticated tool, namely, quantum singular value threshold projection. We quote the following result that we use [GSLW19, Theorem 31, arXiv version].

Theorem 5.1.2 (Singular value threshold projections). *Let U be a unitary acting on $k+1$ qubits and $\Pi, \tilde{\Pi}$ be orthogonal projectors. Suppose*

$$\tilde{\Pi} U \Pi = \sum_{j=1}^m \zeta_j |w_j\rangle\langle v_j|$$

for $\zeta_j \in (0, 1)$ and $\{w_j\}_{j=1}^m$ and $\{v_j\}_{j=1}^m$ two sets of orthonormal vectors. Then for any threshold $t \in (0, 1)$ and $\epsilon, \delta > 0$, there exists a unitary U_q that makes $q = O(\frac{1}{\delta} \log \frac{1}{\epsilon})$

queries to U and $U^\dagger$, and uses $O(q)$ additional one- and two-qubit gates, such that

$$\tilde{\Pi}U_q\Pi = \sum_{j=1}^m \zeta'_j |w_j\rangle\langle v_j|, \quad s.t. \quad \forall j \in [m], \quad \begin{cases} \zeta'_j \geq 1 - \epsilon & \zeta_j \in [t + \delta/2, 1] \\ |\zeta'_j| \leq 1 & \text{if } \zeta_j \in (t - \delta/2, t + \delta/2) \\ \zeta'_j \leq \epsilon & \zeta_j \in [0, t - \delta/2] \end{cases}$$

Computing the circuits for U_q . The two theorems stated above are in fact constructive and provide explicit circuits for the unitaries U_q . In the interest of space, we simply note here that there is a classical runtime overhead of $O(q^3 \text{poly} \log(q/\epsilon))$, and refer the interested reader to [GSLW19, Haa19] for details. For our applications, this overhead is of order $\tilde{O}(n^{3/2})$.

5.2 Quantum worst-case to average-case reductions

In this section, we provide our quantum worst-case to average-case reductions. To obtain our results in a modular way, we start by stating a general technical lemma from which we can easily derive both Theorem 5.2.2 and Theorem 5.2.3.

Lemma 5.2.1. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Let ALG^M be a quantum query algorithm that has oracle access to a matrix $M \in \mathbb{F}^{n \times n}$, gets as input a vector $v \in \mathbb{F}^n$, makes $Q(n)$ queries, and attempts to compute Mv . Then, for every constant $\delta > 0$, there exists a quantum algorithm $(\text{ALG}')^M$ that makes $O(\alpha^{-7/2})$ uses of ALG^M , $O((Q(n) + n^{3/2}) \cdot \alpha^{-7/2})$ queries to U_M and $U_M^\dagger$, uses $O(n^{3/2} \alpha^{-7/2} \cdot \log n \cdot \text{poly}(\log |\mathbb{F}|))$ additional one-qubit and two-qubit gates, and $O(\alpha^{-2} \cdot n \log n)$ ancillary qubits such that the following holds.*

For every matrix $M \in \mathbb{F}^{n \times n}$ such that ALG^M computes Mv correctly with probability α :

$$\Pr_{v, \text{ALG}}[\text{ALG}^M(v) = Mv] = \mathbb{E}_{v \in \mathbb{F}^n} [\|\Pi_{Mv} \text{ALG}^M |v\rangle |0\rangle\|^2] \geq \alpha,$$

the algorithm $(\text{ALG}')^M$ computes Mv correctly for every $v \in \mathbb{F}^n$ with probability $1 - \delta$:

$$\forall v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'}[(\text{ALG}')^M(v) = Mv] = \|\Pi_{Mv} (\text{ALG}')^M |v\rangle |0\rangle\|^2 \geq 1 - \delta.$$

Here and in the following, when we say that ALG' makes k calls to ALG , we mean that ALG' on inputs of length n makes k calls to ALG on inputs of length n .

We prove [Lemma 5.2.1](#) in [Section 5.4](#), after we develop a toolkit of quantum algorithms in [Section 5.3](#) and a quantum local correction lemma in [Section 3.4](#), which will be necessary for our proof.

Equipped with the technical lemma above, in [Section 5.2.1](#) we formally restate [Theorem 5](#) and show how it follows from [Lemma 5.2.1](#). Then, in [Section 5.2.2](#), we formally restate [Theorem 6](#) and show how to prove it using [Lemma 5.2.1](#).

5.2.1 Quantum algorithms for all linear problems

In this section we deal with uniform quantum algorithms; that is, uniformly generated families of quantum circuits $\{C_n\}_{n \in \mathbb{N}}$ for each input length $n \in \mathbb{N}$. Following the literature on quantum algorithms for matrix problems we allow C_n to use a unitary gate U_M that computes matrix entries of M as defined in [Eq. \(5.2\)](#). We obtain the following quantum worst-case to average-case reduction for linear problems.

Theorem 5.2.2. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Let $\mathcal{M} := \{M_n \in \mathbb{F}^{n \times n}\}_{n \in \mathbb{N}}$ be any linear problem, and ALG be an average-case quantum algorithm for $\mathcal{M}$ that takes as input a vector $v \in \mathbb{F}^n$ and satisfies*

$$\Pr_{v, \text{ALG}} [\text{ALG}(v) = Mv] = \mathbb{E}_{v \in \mathbb{F}^n} [\|\Pi_{Mv} \text{ALG} |v\rangle |0\rangle\|^2] \geq \alpha.$$

Then, for every constant $\delta > 0$, there exists a worst-case quantum algorithm ALG' satisfying

$$\forall v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'} [\text{ALG}'(v) = Mv] = \|\Pi_{Mv} \text{ALG}' |v\rangle |0\rangle\|^2 \geq 1 - \delta.$$

ALG' makes $O(\alpha^{-7/2})$ uses of ALG , $O(n^{3/2}\alpha^{-7/2})$ uses of U_M and $U_M^\dagger$, $O(n^{3/2}\alpha^{-7/2} \log n \cdot \text{poly}(\log |\mathbb{F}|))$ additional one-qubit and two-qubit gates, and $O(\alpha^{-2} \cdot n \log n)$ ancillary qubits.

This result follows immediately from [Lemma 5.2.1](#). Observe that [Theorem 5.2.2](#) instantiates a quantum query algorithm ALG as a family $\{C_n\}_{n \in \mathbb{N}}$ of quantum circuits, and considers them as having access to explicit circuits implementing U_M . Thus in particular, the guarantees of [Lemma 5.2.1](#) about the number of queries to U_M and the number of additional gates used by ALG' hold. There are no additional overheads in circuit size from implementing access to sampled vectors, which only cost $\tilde{O}(n)$ gates.

In general, an algorithm that solves a linear problem may exploit the special structure of M , or in the circuit model the matrix entries may be hard-coded. However, note that there

is always a trivial circuit of size $O(n^2)$ that solves a linear problem, both in the classical and quantum settings. Our focus in this work is on fine-grained complexity, and our results hold for the stronger case of non-trivial quantum circuits $\{C_n\}_{n \in \mathbb{N}}$ of *sub-quadratic* size.

5.2.2 Matrix-vector multiplication in the query model

Next, we provide a quantum worst-case to average-case reduction for the Matrix-Vector Multiplication problem in the quantum query model.

Theorem 5.2.3 (Query complexity of Matrix-Vector Multiplication). *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Suppose that there exists a quantum query algorithm ALG that has oracle access to a matrix $M \in \mathbb{F}^{n \times n}$ and a vector v , makes $Q(n)$ queries, and satisfies*

$$\Pr_{\substack{M, v, \\ \text{ALG}}} [\text{ALG}^{M, v} = Mv] = \mathbb{E}_{\substack{M \in \mathbb{F}^{n \times n} \\ v \in \mathbb{F}^n}} \left[\|\Pi_{Mv} \text{ALG}^{M, v} |0\rangle\|^2 \right] \geq \alpha.$$

Then, for every constant $\delta > 0$, there exists a worst-case quantum algorithm ALG' that makes $O(\alpha^{-9/2})$ uses of ALG , makes $O((Q(n) + n^{3/2}) \cdot \alpha^{-9/2})$ queries to U_M and $U_M^\dagger$, and succeeds on all inputs with high probability:

$$\forall M \in \mathbb{F}^{n \times n}, v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'} [(\text{ALG}')^{M, v} = Mv] = \|\Pi_{Mv} (\text{ALG}')^{M, v} |0\rangle\|^2 \geq 1 - \delta.$$

We will prove this worst-case to average-case reduction for the Matrix-Vector Multiplication problem in the quantum query model using [Lemma 5.2.1](#) and an efficient quantum verification algorithm for matrix-vector products (see [Lemma 2.8.1](#) from [Section 2.8](#), or [Lemma 5.3.1](#) for a formal statement).

Proof of Theorem 5.2.3. The algorithm $(\text{ALG}')^{M, v}$ repeats the following procedure $O(1/\alpha)$ times. Sample a uniformly random matrix $R \in \mathbb{F}^{n \times n}$, define $M' = M - R$, and generate unitary oracle $U_{M'}$ for M' . Each query to $U_{M'}$ costs only one query to U_M . Now use [Lemma 5.2.1](#) with $U_{M'}$ to try to compute $b' = M' \cdot v$. Finally, compute $b_R = R \cdot v$ directly without any queries to M , and set $b = b' + b_R$. Verify whether $b = Mv$ using [Lemma 5.3.1](#) with $\varepsilon = \delta/2$, and output b if the verification test passed.

The complexity of the algorithm is determined by $O(1/\alpha)$ applications of [Lemma 5.2.1](#) and verifications from [Lemma 5.3.1](#), and $O(n)$ queries to read the coordinates of v . In

particular, the described algorithm $(\text{ALG}')^{M,v}$ performs $O(\alpha^{-9/2})$ calls of $\text{ALG}^{M,v}$, and $O((Q(n) + n^{3/2}) \cdot \alpha^{-7/2})$ queries to U_M and $U_M^\dagger$.

In order to analyze the correctness of the algorithm $(\text{ALG}')^{M,v}$, we introduce the following notation. For a fixed matrix $M \in \mathbb{F}^{n \times n}$, let $p_M = \Pr_{v, \text{ALG}}[\text{ALG}^{M,v} = Mv]$ denote the probability of computing $\text{ALG}^{M,v} = Mv$ correctly for the fixed value of M and a uniformly random vector $v \in \mathbb{F}^n$. Let us define the set $X \subseteq \mathbb{F}^{n \times n}$ of good matrices where the average-case algorithm ALG succeeds with probability at least $\alpha/2$:

$$X := \left\{ M \in \mathbb{F}^{n \times n} : p_M \geq \frac{\alpha}{2} \right\}.$$

First we observe that $|X| \geq \alpha |\mathbb{F}|^{n^2}/2$. Indeed, by the assumption of the theorem we have that $\mathbb{E}_{M \in \mathbb{F}^{n \times n}}[p_M] \geq \alpha$. Then,

$$\alpha \leq \mathbb{E}_M[p_M] < 1 \cdot \Pr_M[p_M \geq \alpha/2] + \alpha/2 \cdot \Pr_M[p_M < \alpha/2] \leq \Pr_M[p_M \geq \alpha/2] + (\alpha/2) \cdot 1.$$

Thus, $|X| = |\mathbb{F}|^{n^2} \cdot \Pr_M[p_M \geq \alpha/2] \geq \alpha |\mathbb{F}|^{n^2}/2$.

Since in every iteration of the algorithm, the matrix $M' = M - R$ is a uniformly random matrix, we have that $\Pr[M' \in X] \geq \alpha/2$. In the case when $M' \in X$, [Lemma 5.2.1](#) can correctly compute $b' = M'v$ with probability $2/3$. Thus, in every iteration, the described procedure computes $b = Mv$ with probability $\Omega(\alpha)$. By repeating this procedure $O(1/\alpha)$ times (each time verifying whether $b = Mv$ using [Lemma 5.3.1](#) with $\varepsilon = \delta/2$), we have that for every matrix M and every vector v , we compute the product Mv correctly with probability $1 - \delta$. $\square$

5.3 Quantum toolkit for local correction

In this section, we provide a toolkit of quantum algorithms that will allow us to later obtain a quantum local correction lemma (in [Section 3.4](#)), which will underlie our quantum worst-case to average-case reductions. Specifically, in [Section 5.3.1](#), we construct a quantum procedure for flagging correct matrix-vector products in superposition; we use this in [Section 5.3.2](#) to obtain a unitary implementation of an approximation to the indicator function on the set X of good vectors, and in [Section 5.3.3](#) to provide a quantum procedure for efficiently sampling from X ; finally, in [Section 5.3.4](#), we construct a quantum procedure for learning Bogolyubov subspaces from noisy quantum oracles.

5.3.1 Flagging correct matrix-vector products in superposition

As discussed in the technical overview in [Section 2.6](#), a key bottleneck in the classical setting is the efficient verification of computing a matrix-vector product. In the setting of quantum query complexity, this is the following problem:

MATRIX-VECTOR PRODUCT VERIFICATION (MvPV)

Input: Quantum oracles U_M , U_v , and U_b for a matrix $M \in \mathbb{F}^{n \times n}$ and vectors $v, b \in \mathbb{F}^n$.

Output: 1 if $Mv = b$ and 0 otherwise.

Classical query algorithms would analogously have access to an oracle that returns matrix (vector) entries when queried with a row and column index pair.

The quantum query complexity of MvPV has been studied in detail, starting with the work of [\[Bv06\]](#) who showed that matrix-matrix products can be verified with $O(n^{5/3})$ quantum queries, a bound that is sublinear in the size of the input (which is n^2). [\[MNRS07\]](#) later showed that the classical techniques of Freivalds can be adapted to the quantum setting to make this algorithm time-efficient. The special case of matrix multiplication over the Boolean semiring has close relations to path and triangle finding in graphs, and its quantum query complexity has also been studied in great depth [\[MSS07, VW10, Le 12, CKK12, JKM13\]](#). [\[Kot14\]](#), gives a detailed review of the complexities and relationships between different variants of the matrix multiplication and MvPV problems over arbitrary semirings.

Problem	Inputs/Output	Quantum query complexity
Matrix-vector Multiplication (MvM)	In: $M \in \mathbb{F}^{n \times n}, v \in \mathbb{F}^n$ Out: Mv	$\Theta(n^2)$
Matrix-vector Product Verification (MvPV)	In: $M \in \mathbb{F}^{n \times n}, v, b \in \mathbb{F}^n$ Out: $Mv = b?$	$\Theta(n^{3/2})$

Table 5.1: Worst-case quantum query complexity of MvM and MvPV [\[Kot14\]](#).

This extensive literature focuses on the more general problem of matrix-matrix multiplication and product verification over semirings, especially the Boolean case; on the other hand, our interest here lies in the special case of matrix-vector products over finite

fields $\mathbb{F}_p$. In particular, all the results about matrix-vector product verification that we have come across in past work deal exclusively with the query complexity, leaving the algorithm achieving the upper bound, and its computational or gate complexity, implicit. Furthermore, techniques such as those used in [Bv06] do not extend well to our setting — (1) because they use the computation of a logical AND via Grover search, they do not directly lead to a unitary algorithm that can be queried in superposition, due to the traditional problems of overshooting associated with vanilla quantum search; and (2) because they are phrased in the usual manner of performing amplitude amplification followed by measurements to extract the output with constant success probability.

In this section, we construct an efficient quantum algorithm for addressing **MvPV**, with query and gate complexities bounded by the optimal $O(n^{3/2})$, which will be conducive to querying in superposition and consequently to composition with our subsequent subroutine for learning Bogolyubov subspaces. As discussed in Section 2.8, the main technical ingredients we use are fixed-point amplitude amplification and quantum singular value threshold projection.

A simple case of MvPV

We first consider the standard setting where the input vectors v and b are both given by the usual exact quantum query oracles. We later use this simple variant to argue about the case where we only have a noisy version of b , accessed via a noisy quantum oracle.

Lemma 5.3.1 (Quantum MvPV). *Suppose we are given a quantum oracle U_M for a matrix $M \in \mathbb{F}^{n \times n}$*

$$U_M |j, k, z\rangle = |j, k, z \oplus M_{jk}\rangle$$

for all indices $j, k \in [n]$ and $z \in \mathbb{F}$, and quantum oracles U_v, U_b for input vectors $v, b \in \mathbb{F}$

$$U_v |j, z\rangle = |j, z \oplus v_j\rangle$$

$$U_b |j, z\rangle = |j, z \oplus b_j\rangle$$

for all $j \in [n]$ and $z \in \mathbb{F}$. Then there is a gate-efficient quantum algorithm Q_{verify} that accepts with certainty if $Mv = b$, and rejects with probability $1 - \epsilon$ if $Mv \neq b$. Furthermore, Q_{verify} uses $q = O(n^{3/2} \cdot \log \frac{1}{\epsilon})$ queries to U_M and $U_M^\dagger$, $O(n)$ queries to U_v, U_b and their Hermitian conjugates, $O(q \log n \cdot \text{poly log } |\mathbb{F}|)$ additional one-qubit and two-qubit gates, and $O(n \log |\mathbb{F}|)$ ancillary qubits.

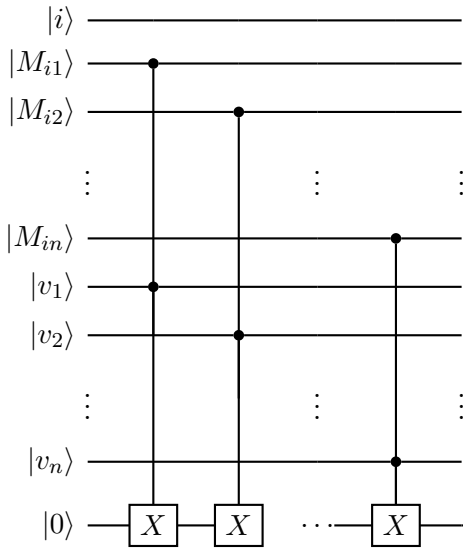
Proof. We denote multiplication and controlled addition over a finite field by a CCX gate

defined by

$$\text{CCX} |s_1\rangle |s_2\rangle |z\rangle = |s_1\rangle |s_2\rangle |z + s_1 \cdot s_2\rangle,$$

where $s_1, s_2, z \in \mathbb{F}$ and $+$ and $\cdot$ are addition and multiplication in $\mathbb{F}$. Implementing such an operation requires only $O(\text{poly log } |\mathbb{F}|)$ elementary two-qubit gates [BBF03]. Throughout this chapter we assume that all algorithms use quantum registers of dimension $p = |\mathbb{F}|$ and perform arithmetic over $\mathbb{F}$, since such arithmetic can be simulated using $O(\log |\mathbb{F}|)$ qubits with $O(\text{poly log } |\mathbb{F}|)$ overhead in gate complexity.

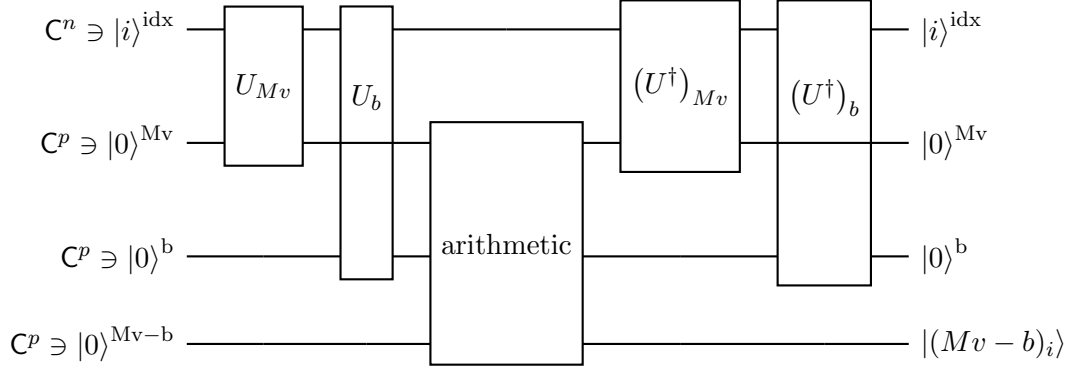
Circuit U_{Mv} : Quantum circuit for entrywise Matrix-vector product



U_{Mv} — Quantum circuit that uses n oracle calls each to U_M , U_v , $U_M^\dagger$, $U_v^\dagger$, and $n \cdot O(\text{poly log } |\mathbb{F}|)$ elementary gates, to implement oracle access to the entries of the Matrix-vector product Mv . The index i is given in an additional register, using which the matrix entries M_{ij} and vector entries V_j are loaded by oracle calls to U_M and U_v (suppressed here for readability); accounting for all ancillary registers, the width of the circuit is $O(n \log |\mathbb{F}|)$ qubits, and its depth is $O(n \text{ poly log } |\mathbb{F}|)$.

The unitaries U_{Mv} , U_b and their Hermitian conjugates can be used once each, along with quantum arithmetic operations over $\mathbb{F}$ using circuits of size $O(\text{poly log } |\mathbb{F}|)$, to obtain a similar oracle U_{Mv-b} for the vector $Mv - b$. The registers used for computing $(Mv)_i$ and b_i can both be perfectly uncomputed and returned to their initial value in this case. Using quantum adder circuits [Gid18] it is possible to construct a comparator circuit with $O(1)$ ancillas and $O(\log |\mathbb{F}|)$ gates that checks whether an entry $(Mv - b)_i$ is zero, and sets a flag qubit to 1 if not. This becomes an oracle to the component-wise indicator function $\mathbf{1}_{(Mv-b)} : [n] \times \mathbb{F} \rightarrow \{0, 1\}$ mapping an index $i \in [n]$ to 0 if $(Mv)_i = b_i$, and 1 otherwise. Denote this oracle by $U_{Mv \stackrel{?}{=} b}$.

Circuit U_{Mv-b} : Quantum circuit for $Mv - b$



U_{Mv-b} — Quantum circuit to implement oracle access to the entries of $Mv - b$ (where we suppress the workspace registers required by U_{Mv} etc.). The width of the circuit is $O(n \log |\mathbb{F}|)$ qubits, and the arithmetics step can be implemented with circuits of size $O(\text{poly log } |\mathbb{F}|)$.

We can then perform a quantum search for nonzero entries in $Mv - b$ using $U_{Mv \stackrel{?}{=} b}$. We start by preparing the uniform superposition over indices, query $U_{Mv \stackrel{?}{=} b}$, and treat nonzero entries as marked.

Define the states $|\psi_0\rangle$, $|\psi_1(v, b)\rangle$, and $|\psi_0(v, b)\rangle$ as the uniform superposition over all indices, and uniform superpositions over indices at which $Mv - b$ has nonzero and zero entries respectively

$$\begin{aligned}
 |\psi_0\rangle &= \frac{1}{\sqrt{n}} \sum_{i \in [n]} |i\rangle \\
 |\psi_1(v, b)\rangle &= \frac{1}{\sqrt{m_{vb}}} \sum_{\substack{i \in [n] \\ (Mv)_i \neq b_i}} |i\rangle \\
 |\psi_0(v, b)\rangle &= \frac{1}{\sqrt{n - m_{vb}}} \sum_{\substack{i \in [n] \\ (Mv)_i = b_i}} |i\rangle,
 \end{aligned} \tag{5.8}$$

and $m_{vb} = |\{i \in [n] : (Mv)_i \neq b_i\}|$. Let $k := \lceil \log n \rceil$. Without loss of generality we can assume n is a power of two since we can pad vectors with zeros if not, while at most

doubling the complexity. Consider the operator that creates the uniform superposition over indices and queries $Mv \stackrel{?}{=} b$, i.e.

$$U = (H^{\otimes k} \otimes \mathbb{1}) U_{Mv \stackrel{?}{=} b}.$$

We omit the workspace registers used by $U_{Mv \stackrel{?}{=} b}$ because in this case the workspace can be uncomputed and returned to the initial all-zeros state.

The action of U on the $|0^{k+1}\rangle$ state is

$$U |0^{k+1}\rangle = \begin{cases} |\psi_0\rangle |0\rangle & Mv = b \\ \sqrt{\frac{n-m_{vb}}{n}} |\psi_0(v, b)\rangle |0\rangle + \sqrt{\frac{m_{vb}}{n}} |\psi_1(v, b)\rangle |1\rangle & Mv \neq b. \end{cases} \quad (5.9)$$

Since we do not know m_{vb} beforehand, and since we will need a unitary subroutine that is run over a superposition of input vectors in $\mathbb{F}^n$ later on, we use fixed point amplitude amplification [Gro05, YLC14, Gue19, GSLW19]. This allows the number of iterations of amplitude amplification to be chosen uniformly for all vectors v without worrying about the problem of under- or over-shooting faced in vanilla amplitude amplification. For completeness, we give a brief overview and technical statement of fixed-point amplitude amplification in Section 5.1.3.

The unitary U of Eq. (5.9) satisfies the conditions required by Theorem 5.1.1 with $\Pi = \mathbb{1}_k \otimes |1\rangle\langle 1|$, preparing a state that has a component flagged with $|1\rangle$ when $Mv \neq b$. In the worst case for $Mv \neq b$, there exists exactly one coordinate $i \in [n]$ at which $(Mv)_i \neq b_i$, so that $m_{vb} = 1$. We hence use fixed point amplification with the worst-case lower bound of $p > \frac{1}{2\sqrt{n}}$ to obtain a unitary U' with the action

$$U' |0^{k+1}\rangle = \begin{cases} |\psi_0\rangle |0\rangle & Mv = b \\ \gamma_{\text{fail}}^{vb} |\psi_0(v, b)\rangle |0\rangle + \gamma_{\text{succ}}^{vb} |\psi_1(v, b)\rangle |1\rangle & Mv \neq b, \end{cases}$$

where $|\gamma_{\text{succ}}^{vb}|^2 \geq 1 - \epsilon$. This unitary U' makes $q = O(\sqrt{n} \cdot \log \frac{1}{\epsilon})$ queries to U and $U^\dagger$, uses $O(q \log n)$ additional elementary gates, and a single ancilla. The only queries in U are those made by $U_{Mv \stackrel{?}{=} b}$ to U_M , U_v , U_b and their conjugates; since $U_{Mv \stackrel{?}{=} b}$ uses n queries to each of these oracles, the net query complexity of U' is $q' = O(nq) = O(n^{3/2})$. Finally since M is of size n^2 and $q' = O(n^{3/2})$ we can always first query v and b into an ancillary

register, since both are vectors of length n .

Effectively, for any desired success probability $\epsilon \in (0, 1)$ we are able to obtain a unitary $Q_{\text{verify}} := U'$ that computes $1_{Mv=b}$ with one-sided bounded error, measuring the output register of which yields the claimed guarantees. $\square$

Flagging the correct matrix-vector product in superposition

In order to establish our reduction, we need to go beyond verifying whether a certain vector, obtained by running **ALG** on an input $v \in \mathbb{F}^n$ and measuring its output register, is correct. In particular, we would like to compute the indicator function 1_X which marks the vectors on which **ALG** succeeds with high probability in superposition over all $v \in \mathbb{F}^n$, in order to learn its Fourier characters of high weight. This is complicated twofold — by the noise or part of the state associated with failure in the output of **ALG**, and the entanglement with workspace registers. Nevertheless, we can compute an approximate, noisy version of the indicator function on good input vectors. As a first step, we construct an algorithm using ideas from [Lemma 5.3.1](#) to ensure that whenever **ALG** outputs the correct answer, it also outputs a flag indicating success.

Lemma 5.3.2 (Noisy quantum MvPV). *Suppose we are given a quantum oracle U_M for a matrix $M \in \mathbb{F}^{n \times n}$*

$$U_M |j, k, z\rangle = |j, k, z \oplus M_{jk}\rangle$$

*for all indices $j, k \in [n]$ and $z \in \mathbb{F}$, and a noisy quantum algorithm **ALG** as described in [Eq. \(5.7\)](#), i.e.*

$$\text{ALG} |v\rangle |0\rangle = \beta_{\text{succ}}^v |v\rangle |Mv\rangle |w_0(v)\rangle + \beta_{\text{fail}}^v |v\rangle |\Psi(v)\rangle.$$

*Then there exists a gate-efficient quantum algorithm $\text{ALG}_{\text{verified}}$ that succeeds and outputs Mv with probability $|\beta_{\text{succ}}^v|^2$ along with a flag indicating success, and similarly outputs a flag indicating failure whenever it outputs a vector $z \neq Mv$, with probability at least $(1 - \epsilon)|\beta_{\text{fail}}^v|^2$. Furthermore, $\text{ALG}_{\text{verified}}$ uses $q = O(n^{3/2} \cdot \log \frac{1}{\epsilon})$ queries to U_M and $U_M^\dagger$, $O(1)$ queries to **ALG**, $O(q \log n \cdot \text{poly log } |\mathbb{F}|)$ additional one-qubit and two-qubit gates, and $O(n \log |\mathbb{F}|)$ ancillary qubits.*

Remark 5.3.3. *The input vectors v and b are taken to be given as states $|v\rangle$ and the right hand side of [Eq. \(5.7\)](#)—created using a single query to **ALG**— respectively. Using a hard-coded circuit of size $O(n \log n)$ we can implement entrywise oracles U_v and U_b that copy out the i^{th} entries of these vectors into a target register controlled on an index register.*

Proof. If **ALG** had the property that either $\beta_{\text{fail}}^v = 0$ or $\beta_{\text{succ}}^v = 0$ always, then we could have proceeded as in [Lemma 5.3.1](#). Nevertheless, if we first prepare the state $\text{ALG} |v\rangle |0\rangle$,

prepare a superposition over indices $i \in [n]$ in an ancillary register, and compute the indicator function $\mathbf{1}_{(Mv-b)}$ using fixed circuits of size $O(n \log n)$ gates to copy (controlled on an index register) the i^{th} entry from the output registers of **ALG**, we now have a unitary U that prepares the state

$$\begin{aligned}
& U \left(|v\rangle \otimes \left(\beta_{\text{succ}}^v |Mv\rangle |w_0(v)\rangle + \beta_{\text{fail}}^v \left(\sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v |z\rangle |w(v, z)\rangle \right) \right) \otimes |0^{k+1}\rangle \right) \\
&= |v\rangle \otimes \left(\beta_{\text{succ}}^v |Mv\rangle |w_0(v)\rangle |\psi_0\rangle |0\rangle + \right. \\
&\quad \left. \beta_{\text{fail}}^v \left(\sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v |z\rangle |w(v, z)\rangle \left[\sqrt{\frac{n - m_{vz}}{n}} |\psi_0(v, z)\rangle |0\rangle + \sqrt{\frac{m_{vz}}{n}} |\psi_1(v, z)\rangle |1\rangle \right] \right) \right), \tag{5.10}
\end{aligned}$$

where the states $|\psi_0\rangle, |\psi_0(v, z)\rangle, |\psi_1(v, z)\rangle$ are defined analogously to [Eq. \(5.8\)](#), m_{vz} is the number of indices at which a vector z disagrees with the correct answer Mv , and to avoid clutter we have not explicitly written the registers used for querying the entries M_{ij} , since they can be uncomputed as before.

As in [Lemma 5.3.1](#), we can apply fixed-point amplitude amplification to the unitary U , with the goal of amplifying the part of the superposition flagged by 1 in the last qubit. Incorrect answers $z \neq Mv$ may in general be wrong only at a single co-ordinate, so that in the worst case there may be a single $\gamma_z^v = 1$ with the corresponding $m_{vz} = 1$. Thus we can use the worst-case lower bound of $\frac{1}{\sqrt{n}}$ on the amplitude of the target state flagged by 1. Since U does not affect the state of the input, output, or workspace registers of **ALG**, the amplified operator $\mathbf{ALG}_{\text{verified}} := U'$ will also preserve the same superposition that is produced by **ALG**, with the component of the state flagged by 1 in the final register

amplified:

$$\begin{aligned}
& \text{ALG}_{\text{verified}} \left(|v\rangle \otimes \left(\beta_{\text{succ}}^v |Mv\rangle |w_0(v)\rangle + \beta_{\text{fail}}^v \left(\sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v |z\rangle |w(v, z)\rangle \right) \right) \otimes |0^{k+1}\rangle \right) \\
&= |v\rangle \otimes \left(\beta_{\text{succ}}^v |Mv\rangle |w_0(v)\rangle |\psi_0\rangle |0\rangle + \right. \\
&\quad \left. \beta_{\text{fail}}^v \left(\sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v |z\rangle |w(v, z)\rangle \left[\gamma_{\text{fail}}^{vz} |\psi_0(v, z)\rangle |0\rangle + \gamma_{\text{succ}}^{vz} |\psi_1(v, z)\rangle |1\rangle \right] \right) \right), \tag{5.11}
\end{aligned}$$

where $|\gamma_{\text{succ}}^{vz}|^2 > 1 - \epsilon$, and $\text{ALG}_{\text{verified}}$ uses $q = O(n^{3/2} \cdot \log \frac{1}{\epsilon})$ queries to U_M and $U_M^\dagger$, $O(1)$ queries to ALG , and $O(q \log n \cdot \text{poly} \log |\mathbb{F}|)$ additional elementary gates, and a single ancillary qubit.

Measuring the register containing the output of ALG , we see that $\text{ALG}_{\text{verified}}$ outputs Mv with probability at least $|\beta_{\text{succ}}^v|^2$ just as ALG does, but now the last register contains a flag qubit set to zero to indicate success. When $\text{ALG}_{\text{verified}}$ outputs an incorrect vector $z \neq Mv$, with probability at least $(1 - \epsilon)|\beta_{\text{succ}}^v|^2$ the flag qubit is set to 1 to indicate failure, giving the guarantees claimed in the lemma. $\square$

5.3.2 Noisy quantum oracles approximating the indicator function 1_X

To obtain our next two quantum procedures: an efficient sampler for the set X of good inputs to ALG and a learner for the Bogolyubov subspace, we first need to obtain a unitary implementation of an approximate version of the indicator 1_X , which is a Boolean valued function defined on $\mathbb{F}^n$. We start by observing that the subroutine $\text{ALG}_{\text{verified}}$ that we constructed in the previous section almost has the basic property that a unitary implementing 1_X should have: it attaches flags zero and one to vectors in X and its complement respectively. However, it is *noisy* in two ways: it errs with high probability on vectors outside X , and only succeeds with low probability on vectors in X . What we would like is (a noisy

version of) the oracle

$$U_X |v\rangle |0\rangle = |v\rangle |1_X(v)\rangle .$$

To obtain such a unitary, we use the machinery of quantum singular value threshold projection on top of $\text{ALG}_{\text{verified}}$. We now go into the details of our construction below, and give a brief technical statement of the quantum singular value threshold projection technique in [Section 5.1.3](#).

For an average-case algorithm ALG with average success probability α as defined in [Eq. \(5.4\)](#), recall that we are interested in the associated set of “good” input vectors defined by

$$X = \{v \in \mathbb{F}^n : |\beta_{\text{succ}}|^2 \geq \frac{\alpha}{2}\} ,$$

which we know has density at least $\alpha/2$ in $\mathbb{F}^n$, by the averaging argument in [Claim 5.4.2](#). The indicator function of this set takes the value $1_X(v) = 1$ when $v \in X$ and $1_X(v) = 0$ otherwise. Denote by U the algorithm $\text{ALG}_{\text{verified}}$ constructed in [Lemma 5.3.2](#). Bundling all the workspace registers together for brevity, we note that it has the following property: defining the projectors $\Pi = \mathbb{1}^v \otimes |0\rangle\langle 0|^{\text{work,flag}}$ and $\tilde{\Pi} = \mathbb{1} \otimes |0\rangle\langle 0|^{\text{flag}}$, where the identity term acts on all registers except the flag qubit, we have that

$$\tilde{\Pi} U \Pi = \sum_{v \in \mathbb{F}^n} |\beta_{\text{succ}}^v| |w_v\rangle\langle v, 0, 0| .$$

We interpret the right hand side above as the singular value decomposition of a matrix with right singular vectors $|v, 0, 0\rangle$, and left singular vectors $|w_v\rangle$ given by

$$|w_v\rangle = \frac{1}{|\beta_{\text{succ}}^v|} \left(\beta_{\text{succ}}^v |v, Mv, w_0(v), \psi_0\rangle + \beta_{\text{fail}}^v \sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} \gamma_z^v \gamma_{\text{fail}}^{vz} |v, z, w(v, z), \psi_0(v, z)\rangle \right) \otimes |0\rangle^{\text{flag}} .$$

The singular values $|\beta_{\text{succ}}^v|$ are defined by the relation

$$|\beta_{\text{succ}}^v|^2 = |\beta_{\text{succ}}^v|^2 + |\beta_{\text{fail}}^v|^2 \sum_{\substack{z \in \mathbb{F}^n \\ z \neq Mv}} |\gamma_z^v|^2 |\gamma_{\text{fail}}^{vz}|^2 .$$

Fixing a threshold parameter $t \in (0, 1)$, consider the partition of $\mathbb{F}^n$ into the following three sets:

1. a good set $X_t^g = \{v \in \mathbb{F}^n : |\beta_{\text{succ}}^v|^2 \geq t + t^2\}$
2. an intermediate set $W_t = \{v \in \mathbb{F}^n : |\beta_{\text{succ}}^v|^2 \in (t - 2t^2, t + t^2)\}$

3. a bad set $X_t^b = \{v \in \mathbb{F}^n : |\beta_{\text{succ}}^v|^2 \leq t - 2t^2\}$.

Choosing $\epsilon = t^2$ in [Lemma 5.3.2](#) we have that when $v \in X_t^g$, $|\beta_{\text{succ}}^v|^2 \geq t + t^2$, and when $v \in X_t^b$, $|\beta_{\text{succ}}^v|^2 \leq t - t^2$.

We see that $U = \text{ALG}_{\text{verified}}$ hence satisfies the conditions required in [Theorem 5.1.2](#), so that we can use it to select the vectors in X_t^g with high probability using the technique of singular value threshold projection. Since $\sqrt{t + t^2} \geq \sqrt{t} (1 + \frac{1}{2}t - \frac{1}{8}t^2)$, we can choose the thresholds in [Theorem 5.1.2](#) to be t as above, and $\delta = \frac{1}{2}t^{3/2} - \frac{1}{8}t^{5/2}$, and obtain a unitary U_q with the action

$$U_q |v, 0, 0\rangle = \tilde{\beta}_{\text{succ}}^v |w_v\rangle |0\rangle^{\text{flag}} + \tilde{\beta}_{\text{fail}}^v |\Psi_v\rangle |1\rangle^{\text{flag}},$$

where the amplitudes satisfy the following guarantees ($\Pr(\text{flag} = 0) := |\tilde{\beta}_{\text{succ}}^v|^2$ etc):

1. for inputs $v \in X_t^g$, $\Pr(\text{flag} = 0) \geq 1 - 2\epsilon$;
2. for inputs $v \in X_t^b$, $\Pr(\text{flag} = 1) \geq 1 - 2\epsilon$.

Importantly, we note that for inputs $v \in W_t$, we get no useful guarantee other than the consistency condition $\Pr(\text{flag} = 0) \leq 1$. The unitary U_q can be implemented using $q = O(\frac{1}{\delta} \log \frac{1}{\epsilon}) = O(\frac{1}{t^{3/2}} \log \frac{1}{\epsilon})$ queries to U , i.e. to $\text{ALG}_{\text{verified}}$, and $O(q)$ additional one-qubit and two-qubit gates.

U_q represents a noisy version of the indicator function on X_t^g — applying a Pauli-X gate to flip the flag qubit, U_q is a noisy quantum oracle for $\mathbf{1}_X$ as defined in [Eq. \(5.6\)](#), with one additional complication: there is a “wasteland” slice $W \subseteq \mathbb{F}^n$ on which U_q gives no guarantee as to the value of $\mathbf{1}_X$. Note that this can in principle be a very large set — by inspecting the averaging argument in [Claim 5.4.2](#) we see that it is possible to construct adversarial examples of ALG for which W can have density as large as $\frac{1-\alpha}{1-\alpha/2}$ for $t = \alpha/2$. Nevertheless, we will show how to overcome this difficulty in [Section 5.4.1](#).

5.3.3 Quantum sampling from the set of good inputs

Using the subroutine $\text{ALG}_{\text{verified}}$, it is also possible to sample from the set of vectors on which ALG succeeds with at least a desired probability. Suppose we are interested in the set $X_\tau = \{v \in \mathbb{F}^n : |\beta_{\text{succ}}^v|^2 > \tau\}$ for some $\tau \in (0, 1)$. We use the same ideas that we presented in the previous section — using singular value threshold projection from [Theorem 5.1.2](#) with the choice of threshold $t = \sqrt{\tau}$ and $\delta = \eta t$ for some $\eta > 0$ and so

$q = O(\frac{1}{\eta\sqrt{\tau}} \log \frac{1}{\epsilon})$, we get a unitary U_q that implements a noisy version of an approximation to $\mathbf{1}_X$. It is sufficient to choose constant η , e.g. $\eta = 0.01$, because unlike in the case of the indicator function in the previous section, we do not require high precision with respect to the wasteland slice W_t : instead of sampling from X_τ , we can easily work with $X_{\tau'} \subseteq X_\tau$ for $\tau' = (1 + \eta)\tau$ without any difficulties.

Suppose we prepare the uniform superposition over all $v \in \mathbb{F}^n$ and run U_q . If the density of X_τ is $\mu(X_\tau)$, we see that with the projector $\tilde{\Pi} = \mathbb{1} \otimes |0\rangle\langle 0|^{\text{flag}}$ we have

$$\left\| \left\langle X_\tau, 0^{\text{flag}} | \tilde{\Pi} U_q | 0 \right\rangle \right\|^2 \geq (1 - \epsilon) \mu(X_\tau) ,$$

where

$$|X_\tau\rangle := \frac{1}{\sqrt{|X_\tau|}} \sum_{v \in X_\tau} |v\rangle$$

is the uniform superposition over vectors in X_τ . In particular, we can apply fixed-point amplitude amplification by [Theorem 5.1.1](#) to U_q , to obtain a boosted unitary U'_q with $q' = O\left(\frac{1}{\sqrt{\mu(X_\tau)}} \log \frac{1}{\delta}\right)$ uses of U_q and $U_q^\dagger$, such that $\|\tilde{\Pi} U'_q | 0 \rangle - |X_\tau, 0^{\text{flag}}\rangle\| \leq \delta$. Note that this procedure gives us a way to sample approximately from the uniform distribution on X_τ that is quadratically faster than a classical algorithm that draws random samples and verifies whether the **ALG** computes correctly on the drawn sample. Applying this argument to the case where $\tau = \alpha/2$, we have the following corollary.

Corollary 5.3.4. *Given an average-case quantum algorithm **ALG** with oracle access to the matrix $M \in \mathbb{F}^{n \times n}$ via U_M , there is a quantum algorithm $\mathbf{Q}_{\text{samp}}$ that uses **ALG** $O\left(\frac{1}{\sqrt{\tau\mu(X_\tau)}}\right)$ times and with high probability outputs a vector $v \in X_\tau$ on which **ALG** succeeds with probability at least τ . $\mathbf{Q}_{\text{samp}}$ makes $q = O\left(n^{3/2} \cdot \frac{1}{\sqrt{\tau\mu(X_\tau)}}\right)$ queries to U_M and $U_M^\dagger$, $O\left(q \log n \cdot \text{poly log } |\mathbb{F}|\right)$ additional one-qubit and two-qubit gates, and $O(n \log |\mathbb{F}|)$ ancillary qubits.*

5.3.4 Learning Bogolyubov subspaces from noisy quantum oracles

We would next like to design a quantum procedure for efficiently learning the Bogolyubov subspace from the noisy quantum oracle we constructed above in [Section 5.3.2](#). Since the

Bogolyubov subspace is characterised by the Fourier coefficients of the indicator 1_X , our problem reduces to learning the Fourier spectrum of functions that are encoded in noisy quantum oracles of the aforementioned form.

[AC02] studied the problem of using noisy evaluations of a linear Boolean function $f(x) = a \cdot x$ to determine the underlying string $a \in \mathbb{F}_2^n$, given a guarantee that the evaluations have an average probability of at least $\frac{1}{2} + \epsilon$ of being correct over random $x \in \mathbb{F}_2^n$. Here we show that the circuit that they consider can be used more generally to sample characters from the Fourier spectrum of a function $f : \mathbb{F}^n \rightarrow \mathbb{F}_2$ that is accessed via a noisy quantum oracle of the type defined in Eq. (5.6). In particular, we have the following result.

Lemma 5.3.5. *Suppose we are given as input a noisy quantum oracle U_f to $f : \mathbb{F}^n \rightarrow \mathbb{F}_2$, such that*

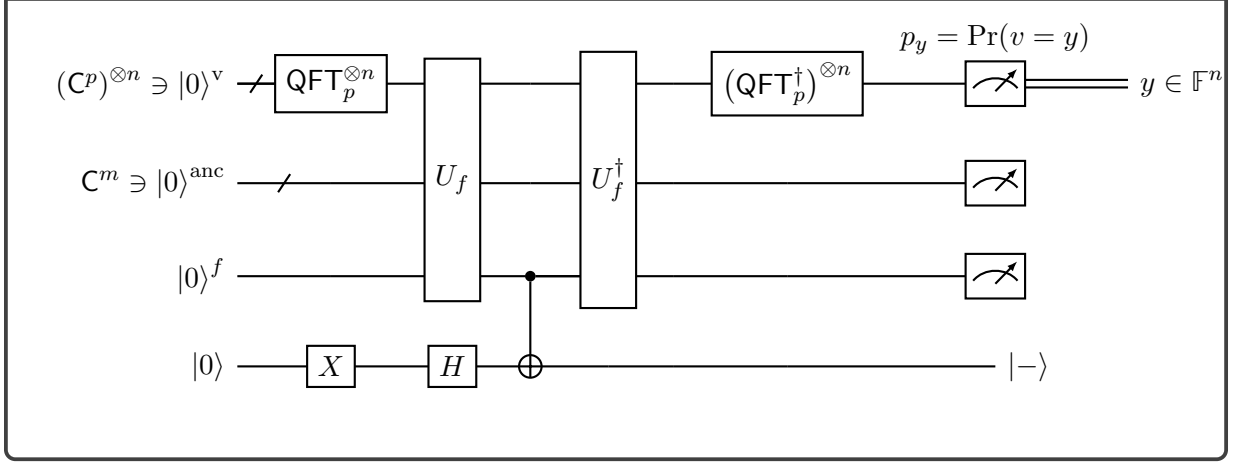
$$U_f |x\rangle |0^{m+1}\rangle = |x\rangle \left(\beta_{\text{succ}}^x |w_0(x)\rangle |f(x)\rangle + \beta_{\text{fail}}^x |w_1(x)\rangle |\overline{f(x)}\rangle \right),$$

where $\forall x \in \mathbb{F}^n$, $|\beta_{\text{succ}}^x|^2 + |\beta_{\text{fail}}^x|^2 = 1$ and $|\beta_{\text{succ}}^x|^2 \geq 1 - \epsilon$, $\overline{f(x)} = f(x) \oplus 1$ is NOT($f(x)$), and $|w_0(x)\rangle$ and $|w_1(x)\rangle$ are arbitrary m -qubit states of the workspace register. Then measuring the top three registers of the following circuit C_{GL} produces output $y \in \mathbb{F}^n$, 0^{m+1} with probability p_y such that $\forall y \in \mathbb{F}^n$,

$$\left| |\hat{f}(y)|^2 - p_y \right| \leq 4\epsilon,$$

where $\hat{f}(y)$ are the Fourier coefficients of f , i.e. $f(x) = \sum_{y \in \mathbb{F}^n} \hat{f}(y) \chi_y(x)$ with $\chi_y(x) := \omega^{-x \cdot y}$.

Circuit C_{GL} : Quantum Goldreich-Levin algorithm for Fourier sampling



Note that the number m of ancillary qubits in the workspace register need not have any relation to n .

Remark 5.3.6. In our application, we actually have a weaker assumption on the input U_f , which in fact is the indicator function of [Section 5.3.2](#). The guarantee we obtain on the Fourier sampler is then

$$\left| |\hat{f}(y)|^2 - p_y \right| \leq 4\epsilon + 4\rho_W,$$

where $\rho_W := |W|/|\mathbb{F}|^n$ is the density of the intermediate set W .

Proof. The input state undergoes the following transformations through the circuit C_{GL} :

$$\begin{aligned} |0\rangle |0\rangle |0\rangle |0\rangle &\xrightarrow{\text{QFT}_p^{\otimes n} \otimes \mathbb{1} \otimes \mathbb{1} \otimes X} \frac{1}{\sqrt{N}} \sum_{x \in \mathbb{F}^n} |x\rangle |0\rangle |0\rangle |1\rangle \\ &\xrightarrow{U_f \otimes H} \frac{1}{\sqrt{N}} \sum_{x \in \mathbb{F}^n} |x\rangle \left(\beta_{\text{succ}}^x |w_0(x)\rangle |f(x)\rangle + \beta_{\text{fail}}^x |w_1(x)\rangle |\overline{f(x)}\rangle \right) |-\rangle \\ &\xrightarrow{\mathbb{1} \otimes \mathbb{1} \otimes \text{CNOT}} \frac{1}{\sqrt{N}} \sum_{x \in \mathbb{F}^n} |x\rangle \left((-1)^{f(x)} \beta_{\text{succ}}^x |w_0(x)\rangle |f(x)\rangle + (-1)^{\overline{f(x)}} \beta_{\text{fail}}^x |w_1(x)\rangle |\overline{f(x)}\rangle \right) |-\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{x \in \mathbb{F}^n} (-1)^{f(x)} |x\rangle \left(\beta_{\text{succ}}^x |w_0(x)\rangle |f(x)\rangle - \beta_{\text{fail}}^x |w_1(x)\rangle |\overline{f(x)}\rangle \right) |-\rangle, \end{aligned} \tag{5.12}$$

where $N = p^n$. In order to establish that the output is $y \in \mathbb{F}^n$ with probability p_y such that $|p_y - \hat{f}(y)^2| < 4\epsilon$ on measuring the first register after executing the entire circuit, we will compute the inner product

$$\beta_y := \left\langle y, 0, 0, - \middle| C \middle| 0, 0, 0, 0 \right\rangle, \quad (5.13)$$

since $p_y = |\beta_y|^2$. Notice that we also have

$$\begin{aligned} |y, 0, 0, -\rangle &\xrightarrow{\text{QFT}_p^{\otimes n} \otimes \mathbb{1} \otimes \mathbb{1} \otimes \mathbb{1}} \frac{1}{\sqrt{N}} \sum_{v \in \mathbb{F}^n} \omega^{y \cdot v} |v, 0, 0, -\rangle \\ &\xrightarrow{U_f \otimes \mathbb{1}} \frac{1}{\sqrt{N}} \sum_{v \in \mathbb{F}^n} \omega^{y \cdot v} |v\rangle \left(\beta_{\text{succ}}^v |w_0(v)\rangle |f(v)\rangle + \beta_{\text{fail}}^v |w_1(v)\rangle |\overline{f(v)}\rangle \right) |-\rangle, \end{aligned} \quad (5.14)$$

and we can compute the amplitude β_y in Eq. (5.13) by taking the inner product between the states on the last lines of Eqs. (5.12) and (5.14). Since $f(x) \in \mathbb{F}_2$ we have that $\langle f(x) | \overline{f(x)} \rangle = 0$. Similarly, $\langle v | x \rangle = \delta_{vx}$, and while the states of the workspace register may not be orthogonal, they are normalized. Hence we have

$$\beta_y = \frac{1}{N} \sum_{v \in \mathbb{F}^n} \omega^{y \cdot v} (-1)^{f(v)} \left(|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2 \right).$$

Since for every $v \in \mathbb{F}^n$, $|\beta_{\text{succ}}^v|^2 \geq 1 - \epsilon$ and $|\beta_{\text{succ}}^v|^2 + |\beta_{\text{fail}}^v|^2 = 1$, it always holds that $|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2 \geq 1 - 2\epsilon$. Unlike the Boolean case (i.e. $\mathbb{F} = \mathbb{F}_2$), the Fourier coefficients of f are no longer real numbers, and so need some additional care. Nevertheless, using their definition from Eq. (5.1), and the Cauchy-Schwarz inequality, we have

$$\begin{aligned} |\hat{f}(y) - \beta_y| &= \frac{1}{N} \left| \sum_{v \in \mathbb{F}^n} \omega^{y \cdot v} (-1)^{f(v)} \left(1 - \left(|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2 \right) \right) \right| \\ &\leq \frac{1}{N} \left| \sum_{v \in \mathbb{F}^n} \omega^{2y \cdot v} (-1)^{2f(v)} \right|^{1/2} \left| \sum_{v \in \mathbb{F}^n} \left(1 - \left(|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2 \right) \right)^2 \right|^{1/2} \\ &\leq \frac{1}{N} \cdot \sqrt{N} \cdot 2\epsilon \sqrt{N} \\ &\leq 2\epsilon. \end{aligned} \quad (5.15)$$

As $|\hat{f}(y)|, |\beta_y| \leq 1$, and since $||x|^2 - |y|^2| \leq |x^* + y^*||x - y| \leq 2|x - y|$ when $0 \leq |x|, |y| \leq 1$ and x^* denotes complex conjugation, we finally have that

$$||\hat{f}(y)|^2 - p_y| \leq 4\epsilon,$$

showing that the circuit $\mathbf{C}_{\text{GL}}$ can sample approximately from the Fourier spectrum of f . $\square$

The quantum Fourier transform $\mathbf{QFT}_p$ over finite fields $\mathbb{F}_p$ can be implemented efficiently with $O(\log^2 |\mathbb{F}|)$ elementary gates and in depth $O(\log |\mathbb{F}|)$ [Bea97, Hoy97, MRR06].

Learning subspace coefficients of 1_X : Our interest is in the indicator function $1_X : \mathbb{F}^n \rightarrow \{0, 1\}$, which takes value 1 on the set of vectors on which **ALG** succeeds with appreciable probability $|\beta_{\text{succ}}^v|^2 \geq \alpha$. Using the unitary U_q of Section 5.3.2 almost satisfies the condition required for Lemma 5.3.5: $|\beta_{\text{succ}}|^2 \geq 1 - \epsilon$ and $|\beta_{\text{fail}}|^2 \leq \epsilon$, where ϵ can be chosen to be $o(1)$. To see the bound noted in Remark 5.3.6, suppose the densities of the sets X_t^g, X_t^b and W_t are ρ_g, ρ_b and ρ_W respectively and let $\pi_v := \left(1 - \left(|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2\right)\right)^2$.

Then in Eq. (5.15) we have

$$\begin{aligned} |\hat{f}(y) - \beta_y| &\leq \frac{1}{N} \left| \sum_{v \in \mathbb{F}^n} \omega^{2y \cdot v} (-1)^{2f(v)} \right|^{1/2} \left| \sum_{v \in \mathbb{F}^n} \left(1 - \left(|\beta_{\text{succ}}^v|^2 - |\beta_{\text{fail}}^v|^2\right)\right)^2 \right|^{1/2} \\ &\leq \frac{1}{N} \cdot \sqrt{N} \cdot \left| \sum_{v \in X_t^g} \pi_v + \sum_{v \in X_t^b} \pi_v + \sum_{v \in W_t} \pi_v \right|^{1/2} \\ &\leq \frac{1}{\sqrt{N}} \cdot |2\epsilon\rho_g N + 2\epsilon\rho_b N + 2\rho_W N|^{1/2} \\ &\leq 2\epsilon + 2\rho_W, \end{aligned}$$

where on the third line we used the definition of U_q to see that the quantity $\pi_v \leq 2\epsilon$ on both X_t^g and X_t^b , while we only have the guarantee that $\pi_v \leq 2$ on the set W_t , and that $\rho_g + \rho_b + \rho_W = 1$. Hence we have the following corollary.

Corollary 5.3.7. *Given an average-case quantum algorithm $\mathbf{ALG}^M$ as described in Eq. (5.7) and the oracle U_M for a matrix $M \in \mathbb{F}^{n \times n}$, we can learn heavy Fourier characters χ_y with $\hat{f}(y) \geq c$ of the indicator function on the set X of inputs on which **ALG** succeeds with high probability, using **ALG** $O(1/c)$ times, $q = O\left(\frac{1}{c} \cdot n^{3/2}\right)$ queries to U_M and $U_M^\dagger$,*

$O(q \cdot \log n \cdot \text{poly log } |\mathbb{F}|)$ additional one-qubit and two-qubit gates, and $O(n \log |\mathbb{F}|)$ ancillary qubits.

The procedure is to simply measure the state output by $\mathbf{C}_{\text{GL}}$ to obtain a character $y \in \mathbb{F}^n$, and then run standard quantum amplitude estimation to estimate the value of the corresponding Fourier coefficient to additive precision $c/100$, which will use $\mathbf{C}_{\text{GL}}$ a total of $O(1/c)$ times.

5.4 Reductions for linear problems

In this section, we prove [Lemma 5.2.1](#). Namely, we will present an algorithm $\mathbf{ALG}'$ whose complexity is as required by the lemma statement. In order to prove the correctness of the algorithm, we will consider $M \in \mathbb{F}^{n \times n}$ such that $\Pr_{v, \mathbf{ALG}}[\mathbf{ALG}^M(v) = Mv] \geq \alpha$, and we will prove for this M and every vector $v \in \mathbb{F}^n$, that $\Pr_{\mathbf{ALG}'}[(\mathbf{ALG}')^M(v) = Mv] \geq 1 - \delta$. To this end we fix a matrix M satisfying the premise:

$$\Pr_{v, \mathbf{ALG}}[\mathbf{ALG}^M(v) = Mv] \geq \alpha. \quad (5.16)$$

Before describing the proof of [Lemma 5.2.1](#) we introduce the following notation. For each $v \in \mathbb{F}^n$ let $p_v = \Pr_{\mathbf{ALG}}[\mathbf{ALG}(v) = Mv]$ be the probability that $\mathbf{ALG}$ computes correctly the output on input v , where the probability is taken only over the quantum randomness of $\mathbf{ALG}$.

Before proceeding with the proof, we need the following definition of threshold sets.

Definition 5.4.1. *For an algorithm $\mathbf{ALG}^M$ and a matrix M satisfying [Eq. \(5.16\)](#), we define the set of its good inputs, i.e., the inputs $v \in \mathbb{F}^n$ such that $\mathbf{ALG}^M(v) = M \cdot v$ with a non-negligible probability as follows.*

$$X_\kappa := \{v \in \mathbb{F}^n : p_v \geq \kappa\};.$$

5.4.1 Properties of threshold sets

Below we prove several claims regarding the threshold sets X_κ .

Claim 5.4.2. *For $\kappa \leq \alpha/2$, the density of X_κ is at least $\alpha/2$, i.e., $|X_\kappa| \geq \frac{\alpha}{2}|\mathbb{F}|^n$.*

Proof. By the assumption of the lemma we have $\mathbb{E}_{v \in \mathbb{F}^n}[p_v] \geq \alpha$, and hence

$$\alpha \leq \mathbb{E}_v[p_v] \leq 1 \cdot \Pr_v[p_v \geq \alpha/2] + \alpha/2 \cdot \Pr_v[p_v < \alpha/2] \leq \Pr_v[p_v \geq \kappa] + \alpha/2.$$

Therefore, for all $\kappa \leq \alpha/2$, we have that $\Pr_v[p_v \geq \kappa] \geq \alpha/2$, as required. $\square$

Next, we prove that if we choose a random $\tau \in [\alpha/4, \alpha/2]$ and $\tau' = \tau - 1/K$ for a sufficiently large K , then, with high probability over the random choices of τ the sets X_τ and $X_{\tau'}$ will have almost the same density.

Claim 5.4.3. *For a parameter K let $r \in \{1, \dots, K\}$ be chosen uniformly at random. Let $\tau = (1 + r/K)\frac{\alpha}{4} \in [\frac{\alpha}{4} + \frac{\alpha}{4K}, \frac{\alpha}{2}]$, and $\tau' = \tau - \frac{\alpha}{4K}$. Then, $\Pr\left[\left|\frac{|X_{\tau'}|}{|\mathbb{F}^n|} - \frac{|X_\tau|}{|\mathbb{F}^n|}\right| \leq \frac{2}{K}\right] > 1/2$.*

Proof. Note that the interval $[\tau, \tau']$ is sampled by dividing $[\alpha/4, \alpha/2]$ into K intervals, and taking one of them uniformly at random, it follows that $\mathbb{E}\left[\left|\frac{|X_{\tau'}|}{|\mathbb{F}^n|} - \frac{|X_\tau|}{|\mathbb{F}^n|}\right|\right] \leq 1/K$. The claim follows by Markov's inequality. $\square$

In particular [Claim 5.4.3](#) implies the following corollary.

Corollary 5.4.4. *For a parameter K let $r \in \{1, \dots, K\}$ be chosen uniformly at random. Let $\tau = (1 + r/K)\frac{\alpha}{4} \in [\frac{\alpha}{4} + \frac{\alpha}{4K}, \frac{\alpha}{2}]$, and $\tau' = \tau - \frac{\alpha}{4K}$.*

Suppose that $X^* \subseteq \mathbb{F}^n$ is an arbitrary set such that $X_\tau \subseteq X^* \subseteq X_{\tau'}$. Then with probability at least $1/2$ (over the choice of τ) it holds that

$$|\hat{1}_{X^*}(r) - \hat{1}_{X_\tau}(r)| \leq \frac{2}{K}$$

for all $r \in \mathbb{F}^n$.

Proof. By [Claim 5.4.3](#) we have $\Pr\left[\left|\frac{|X_{\tau'}|}{|\mathbb{F}^n|} - \frac{|X_\tau|}{|\mathbb{F}^n|}\right| \leq \frac{2}{K}\right] > 1/2$. Suppose this event happens and $\left|\frac{|X_{\tau'}|}{|\mathbb{F}^n|} - \frac{|X_\tau|}{|\mathbb{F}^n|}\right| \leq \frac{2}{K}$. Since $X_\tau \subseteq X^* \subseteq X_{\tau'}$, it follows that $\left|\frac{|X^*|}{|\mathbb{F}^n|} - \frac{|X_\tau|}{|\mathbb{F}^n|}\right| \leq \frac{2}{K}$. We observe that

$$|\hat{1}_{X^*}(r) - \hat{1}_{X_\tau}(r)| \leq \left|\frac{|X_\tau|}{|\mathbb{F}^n|} - \frac{|X^*|}{|\mathbb{F}^n|}\right| \leq \frac{2}{K},$$

which finishes the proof of the corollary. $\square$

5.4.2 Proof of Lemma 5.2.1

In this section, we prove Lemma 5.2.1, which we restate below for convenience.

Lemma 5.2.1. *Let $\mathbb{F} = \mathbb{F}_p$ be a prime field, $n \in \mathbb{N}$, and $\alpha := \alpha(n) \in (0, 1]$. Let ALG^M be a quantum query algorithm that has oracle access to a matrix $M \in \mathbb{F}^{n \times n}$, gets as input a vector $v \in \mathbb{F}^n$, makes $Q(n)$ queries, and attempts to compute Mv . Then, for every constant $\delta > 0$, there exists a quantum algorithm $(\text{ALG}')^M$ that makes $O(\alpha^{-7/2})$ uses of ALG^M , $O((Q(n) + n^{3/2}) \cdot \alpha^{-7/2})$ queries to U_M and $U_M^\dagger$, uses $O(n^{3/2} \alpha^{-7/2} \cdot \log n \cdot \text{poly}(\log |\mathbb{F}|))$ additional one-qubit and two-qubit gates, and $O(\alpha^{-2} \cdot n \log n)$ ancillary qubits such that the following holds.*

For every matrix $M \in \mathbb{F}^{n \times n}$ such that ALG^M computes Mv correctly with probability α :

$$\Pr_{v, \text{ALG}} [\text{ALG}^M(v) = Mv] = \mathbb{E}_{v \in \mathbb{F}^n} \left[\|\Pi_{Mv} \text{ALG}^M |v\rangle |0\rangle\|^2 \right] \geq \alpha ,$$

the algorithm $(\text{ALG}')^M$ computes Mv correctly for every $v \in \mathbb{F}^n$ with probability $1 - \delta$:

$$\forall v \in \mathbb{F}^n \quad \Pr_{\text{ALG}'} [(\text{ALG}')^M(v) = Mv] = \|\Pi_{Mv} (\text{ALG}')^M |v\rangle |0\rangle\|^2 \geq 1 - \delta .$$

Let ALG^M be the average-case quantum algorithm from the lemma statement that has query access to the matrix M . Note that given the algorithm ALG^M we can define an algorithm $\text{ALG}_{\text{boost}}^M$ that given an input v makes $O(1/\alpha)$ calls to $\text{ALG}^M(v)$ independently, and each time verifies the result using Lemma 5.3.2. Therefore, for every constant δ , we may assume that there is at least an $\alpha/2$ fraction of inputs on which $\text{ALG}_{\text{boost}}^M$ outputs the correct answer with probability at least $1 - \delta/8$.

We define $(\text{ALG}')^M$ as follows.

Algorithm 1: Reduction for linear problems

Input: ALG^M , $v \in \mathbb{F}^n$, $\alpha \in (0, 1)$

Output: $M \cdot v$

1. Let $K = \frac{4}{\alpha^{3/2}}$, and let $r \in \{1, \dots, K\}$ be chosen uniformly at random.
2. Setting a random threshold: Let $\tau = (1 + r/K) \frac{\alpha}{4} \in [\frac{\alpha}{4}, \frac{\alpha}{2} - \frac{1}{K}]$, and $\tau' = \tau - \frac{1}{K}$.
3. Membership Oracle for X^* : Construct a quantum noisy membership oracle $\tilde{O}_{X^*}$ using [Section 5.3.2](#) such that with high probability $X_\tau \subseteq X^* \subseteq X_{\tau'}$.
4. Learning X_τ : Having $\tilde{O}_{X^*}$ defined above, apply the quantum local correction lemma ([Lemma 3.5.1](#)) with error parameter $\delta = 1/6$ to compute a collection of $t = O(1/\alpha^2)$ vectors $B = \{b_1, \dots, b_t \in \mathbb{F}^n\}$ and indices $k_1, k_2, \dots, k_t$. Indeed, by [Corollary 5.4.4](#), we may apply [Lemma 3.5.1](#).
5. Efficient sampling from X_τ : Sample x_1, x_2, x_3 from X_τ using [Corollary 5.3.4](#) with $\kappa = \tau$.
6. Self-correcting v : Using the set B and indices k_i computed in Step 3, let s be the t -sparse vector defined as in [Lemma 3.5.1](#):

$$s = \sum_{j=1}^t \langle v, b_j \rangle \cdot \vec{e}_{k_j}, \text{ and}$$

$$x_4 = v - x_1 - x_2 - x_3 - s.$$

7. Computing $M \cdot v$: Let $b = \text{ALG}_{\text{boost}}^M(x_1) + \text{ALG}_{\text{boost}}^M(x_2) + \text{ALG}_{\text{boost}}^M(x_3) + \text{ALG}_{\text{boost}}^M(x_4) + M \cdot s$, where $M \cdot s$ is computed by multiplying M by all t non-zero entries of s .
8. Verification: Using [Lemma 5.3.1](#) with $\varepsilon = O(\alpha^2)$ verify whether $b = M \cdot v$. If the verification accepts, return b .
9. Probability amplification: Repeat the steps 5–8 $O(1/\alpha^2)$ times.

Correctness: By our choice of the parameters and the justifications in the algorithm, we see that with constant probability we simultaneously have that (i) the quantum oracle $\tilde{O}_{X^*}$ from [Lemma 5.3.2](#) for every $x \in \mathbb{F}^n$, outputs $\tilde{O}_{X^*}(x) = 1_{X^*}(x)$ in Step 3; (ii) The set

B and indices k_i are computed correctly by Lemma 3.5.1 in Step 4; (iii) Step 5 produces uniformly random samples $x_1, x_2, x_3 \in X_\tau$.

Assuming successful executions of the previous steps, by Lemma 3.5.1 for x_4 computed in Step 6 we have that

$$\Pr_{x_1, x_2, x_3 \in X_\tau} [x_4 \in X_\tau] \geq \alpha^2.$$

Observing that for $x_1, x_2, x_3, x_4 \in X^*$ it holds that $\Pr[\text{ALG}_{boost}^M(x_i) = Mx_i \ \forall i = 1, 2, 3, 4] \geq 1 - \delta/2$. Therefore, with probability $\Omega(\alpha^2)$, the result computed in Step 7 is the correct output $b = M \cdot v$.

Given the representation of X_τ from Step 4 we repeat steps 5–8 $O(1/\alpha^2)$ times to amplify the probability of success to $2/3$ for every input vector v .

Query complexity: First, we bound the number of queries made by our algorithm in each iteration (Steps 3–8). This amounts to bounding the number of queries required to learn the heavy Fourier coefficients of 1_{X^*} in Step 4 (using the oracle from Step 3), queries in Step 5 to sample from X_M , queries in the four executions of ALG_{boost}^M in Step 7 (each corresponding to $O(1/\alpha)$ calls to ALG^M and $O(1/\alpha)$ times verifying the computation), queries required to compute $M \cdot s$ in Step 7, and queries required for verification in Step 8.

By Lemma 3.5.1, in Step 4 we need at most $O(\alpha^{-7/2})$ queries to ALG^M , and $O(n^{3/2} \cdot \alpha^{-7/2})$ queries to U_M and $U_M^\dagger$ to find the set B and indices $k_1, k_2, \dots, k_t$. Note that the algorithm does not repeat Step 4 $O(\alpha^{-2})$ times.

In step 5, by Corollary 5.3.4, we make $O(1)$ uses of the noisy unitary implementation of the approximate indicator function in Section 5.3.2, which translates to $O(n^{3/2} \cdot \alpha^{-3/2})$ queries to U_M and $U_M^\dagger$ and $O(\alpha^{-3/2})$ queries to ALG^M . In Step 7, each application of ALG_{boost}^M uses $O(1/\alpha)$ calls to ALG^M and $O(1/\alpha)$ verification steps. In addition s is t -sparse for $t = 4/\alpha^2$, we compute $M \cdot s$ by making at most $O(n/\alpha^2)$ queries. Step 8 makes $O(n^{3/2})$ queries to U_M and $U_M^\dagger$. Repeating steps 5–8 for $O(1/\alpha^2)$ times leads to the bound of $O(n^{3/2} \cdot \alpha^{-7/2})$ on the query complexity of the constructed algorithm.

Efficiency: Now we count the additional gates used by our algorithm. The membership oracle from Lemma 5.3.2 in Step 3, Fourier sampling circuit from Lemma 3.5.1 in Step 4, and efficient sampling circuit from Corollary 5.3.4 in Step 5 all use $O(n^{3/2} \log n \cdot \alpha^{-3/2} \cdot \text{poly}(\log |\mathbb{F}|))$ additional gates. Additionally, we need $O(n)$ gates to represent vectors s and v_i for $i \in \{1, 2, 3, 4\}$. Thus, in total our algorithm uses $O(n^{3/2} \log n \cdot \alpha^{-7/2} \cdot \text{poly}(\log |\mathbb{F}|))$ additional one-qubit and two-qubit gates, and use $O(\alpha^{-2} \cdot n \log n)$ ancillary qubits.

Large fields: For the case where the field size is large (say, $|\mathbb{F}| \geq 10/\alpha$), there is a simpler worst case to average case reduction. Note that since we can efficiently verify a candidate solution using [Lemma 5.3.1](#), we can sample a line ℓ passing through the input vector v and a uniformly random point $x \in \mathbb{F}^n$. It is easy to show that with probability at least $\Omega(\alpha)$, at least an $\Omega(\alpha)$ -fraction of the points on ℓ are in the good set X , i.e., the set of points on which ALG^M outputs the correct answer with probability at least $\Omega(\alpha)$. Thus, with probability $\Omega(\alpha^3)$, we sample two random points a, b on ℓ that both belong to X , and then we can interpolate the value of $M \cdot v$ from $\text{ALG}^M(a)$ and $\text{ALG}^M(b)$. We note that this interpolation technique inherently requires large fields, while the proof presented above works over all finite fields.

Making ALG' a unitary: Finally, we note that it is straightforward to turn the algorithm ALG' , which is produced by our reduction, into a unitary quantum algorithm. Yet, it is important to do this carefully so as to keep the overheads in gate complexity of the resulting unitary within the desired bound of $O(n^{3/2})$. We sketch how to do this below.

Suppose we have performed [Step 4](#) and obtained a classical description of the set of vectors B and the corresponding indices $k_1, \dots, k_t$. We now need to perform [Steps 5-9](#) in a unitary manner. In lieu of [Step 5](#), we maintain three copies of the output state of the unitary Q_{samp} of [Corollary 5.3.4](#) in three quantum registers, one each for the vectors x_1, x_2 and x_3 . [Step 6](#) only involves arithmetic operations over $\mathbb{F}$, and we perform the required computation by a unitary circuit that computes the sparse shift vector in the form of a state $|s\rangle$, using B and $k_1, \dots, k_t$. In an ancillary register, we then compute a superposition of vectors corresponding to $v - x_1 - x_2 - x_3 - s$, using the states representing the uniform samples x_1, x_2 and x_3 .

By using amplitude amplification, we can construct a unitary implementation corresponding to $\text{ALG}_{\text{boost}}^M$ with $O(\frac{1}{\sqrt{\alpha}} \log \frac{1}{\delta})$ uses of $\text{ALG}_{\text{verified}}$. Next, we apply this unitary to each of the four registers corresponding to $x_1, \dots, x_4$, and also compute $M \cdot s$ using the circuit described in [Section 5.3.1](#). Finally, we perform more arithmetic to compute a state corresponding to the vector b of [Step 7](#) in an ancillary register. Applying Q_{verify} from [Lemma 5.3.1](#) to the registers containing v and b , we obtain a version of ALG' which attaches a flag indicating success to its output register. This can then be boosted using $O(1/\alpha)$ rounds of fixed-point amplitude amplification to obtain the desired unitary quantum algorithm.

Chapter 6

Conclusion

We conclude the first part of this thesis by discussing some of the subsequent works that have followed up on these results since the preliminary versions of these works were published. In the end, we mention some of the interesting problems that are still open, and discuss some of the challenges and barriers in solving them.

6.1 Subsequent work

Following the publication of the preliminary version of this work, several subsequent results have extended and strengthened its results. We report on some of them below.

The main drawback of [Theorem 1](#), which presents worst-case to average-case reductions for the matrix multiplication problem, is that the running time of the worst-case algorithm has quasi-polynomial dependence on $1/\alpha$. To address this, Hirahara and Shimizu [\[HS23\]](#) employed direct product theorems to achieve a nearly-linear dependence on $1/\alpha$ for both the matrix multiplication and online matrix-vector multiplication problems.

The work of [\[GSS24\]](#) initiated the study of worst-case to *weak*-average-case reductions for matrix multiplication. It used the self-correction technique of [\[BLR90\]](#) to resolve the problem in the low-error regime, and applied the additive combinatorics framework developed in this work to address the high-error regime (over $\mathbb{F}_2$) for *one-sided* weak-average-case algorithms.

Subsequently, [\[HS25a\]](#) introduced a multi-output version of Yao’s XOR lemma and used it to construct *non-uniform* worst-case to weak-average-case reductions for matrix multiplication even in the high-error regime. In addition, they used coding-theoretic techniques

to develop new uniform reductions for matrix multiplication and optimal reductions for online matrix-vector multiplication.

In [HS25b], Hirahara and Shimizu combined the techniques from this work, [GSS24], and most notably [HS25a], with an improved coding scheme. This led to efficient and uniform worst-case reductions for matrix multiplication across all error regimes and over all prime fields.

Most recently, the authors of [SS25] simplified the proof of [HS25b] and improved the running time of the resulting worst-case algorithm for matrix multiplication as a function of α .

Another recent work [VZ25] studies a classical worst-case to average-case reduction for matrix multiplication in the presence of computational assumptions. In particular, assuming hardness of the learning parity with noise (LPN), they construct a distribution over a family of trapdoor matrices that are indistinguishable from random matrices. Having this, they were able to show a *uniform* worst-case to weak-average-case reduction with optimal parameters, albeit with a computational assumption, improving over [HS25a] where they showed a non-uniform reduction.

On the quantum side, very recently [AK25] extended the ideas of [HS23] to the quantum setting, and showed a new quantum worst-case to average-case reduction for the matrix vector multiplication problem. Their reduction avoids using the machinery of additive combinatorics, and shows that an equivalent of the hardness self-amplification technique shown in [HS23] exists in the quantum setting as well.

6.2 Open problems

Our work leaves many natural open problems, and raises several new directions of investigation. One example is obtaining reductions for various natural problems in other computational models (e.g., communication complexity, property testing, PAC learning, and beyond). However, for brevity, we would like to focus on and highlight three directions that we find particularly promising.

In Theorem 5, we have shown fine-grained *quantum* worst-case to average-case reductions, where the success rate is arbitrarily small, *for all linear problems*. These reductions make crucial use of quantum procedures that speed up classical computational tasks such as linear verification and learning of Bogolyubov subspaces from approximate indicators encoded in noisy quantum oracles.

In contrast to our general result for quantum algorithms, these computational tasks constitute a complexity bottleneck in the classical setting, and in turn such worst-case to average-case reductions are only known for a small number of specific problems or restricted computational models, e.g., linear problems in the setting of static data structures (Theorem 2). It remains open whether such general results for all linear problem can also be obtained in the classical setting.

Thus an immediate and alluring question is whether our local correction via additive combinatorics framework can also be used to show worst-case to average-case reductions for all linear problems for classical (uniform) algorithms. We observe that using similar techniques as in Theorem 2, our framework can be used to show that given an efficient average-case algorithm and an efficient *verifier* for the problem, one can indeed design an explicit efficient worst-case circuit or uniform algorithm. A natural open problem here is to eliminate the assumption about the verifier and answer the aforementioned question in the affirmative.

Open Problem 1. *Are there efficient transformations of classical algorithms for general linear problems, which are only correct on 1% of their inputs, into similarly efficient worst-case classical algorithms?*

Another fascinating open problem is to improve the running time (as a function of $\alpha = 1/2 + \varepsilon$) of the worst-case algorithms in the line of work on worst-case to *weak*-average-case reductions for matrix multiplication [GSS24, HS25a, HS25b, SS25].

Open Problem 2. *Assume there is an algorithm that runs in time T , and on average computes a $1/2 + \varepsilon$ fraction of entries of matrix multiplication over $\mathbb{F}_2$. Can it be transformed into a worst-case algorithm running in time $\tilde{O}(T/\varepsilon)$.*

Returning to the quantum setting, it is natural to ask whether the framework we constructed can be extended beyond the class of linear problems, to tasks such as matrix multiplication.

Open Problem 3. *Can efficient average-case quantum algorithms with success rate 1% for large classes of non-linear problems be transformed into efficient worst-case quantum algorithms?*

The last open problem we would like to raise refers to quantum algorithms that act on data of exponential size, encoded in the amplitudes of a quantum state, as in the celebrated HHL algorithm [HHL09]. It would be highly appealing to extend the framework presented in this work to this setting, where quantum algorithms are extremely powerful.

Open Problem 4. *Can we obtain fine-grained worst-case to average-case reductions for quantum algorithms such as the HHL algorithm, in which the data (and the output) are encoded in the amplitudes of a quantum state?*

Part II

Lower Bounds on Non-Local Quantum Computation

Chapter 7

Introduction

A non-local quantum computation (NLQC) replaces a local interaction with entanglement and a single simultaneous round of communication. See Fig. 7.1 for an illustration. In this figure, Part (a) shows a channel $\mathcal{T}_{AB \rightarrow AB}$ that is implemented by directly interacting the input systems and Part (b) shows a non-local quantum computation, where the goal is for the action of this circuit on the AB systems to approximate the channel $\mathcal{T}_{AB \rightarrow AB}$. NLQC has applications in a number of areas, including quantum position-verification [KMSB06, KMS11, Mal16], Hamiltonian simulation [ACH⁺24], the AdS/CFT correspondence [May19, MPS20, May21, May22], and information theoretic cryptography [ABM⁺23]. In all of these settings, we are interested in understanding the resources necessary to implement a given computation in the non-local form. Lower bounds on the entanglement necessary to complete a non-local computation are so far poorly understood but are central to these applications.

Two important classes of non-local computations are f -routing and f -BB84, both initially studied in the context of quantum position-verification [KMS11, BCS22]. These computations involve classical inputs of size n and $O(1)$ size quantum inputs. The local implementation of these computations involves computing a classical function f from the classical inputs, then doing $O(1)$ quantum operations conditioned on the outcome. A long-standing goal has been to prove that the non-local implementation of the same computation requires growing quantum resources, and in particular growing entanglement. Recently, a bound on the quantum system size was proven in [BCS22], but this assumes a purified view that absorbs classical processing into the quantum part of the computation.

Here in this thesis, we achieve the following two goals. First, we will prove a lower bound for entanglement, at least as quantified by the Schmidt rank of the shared system,

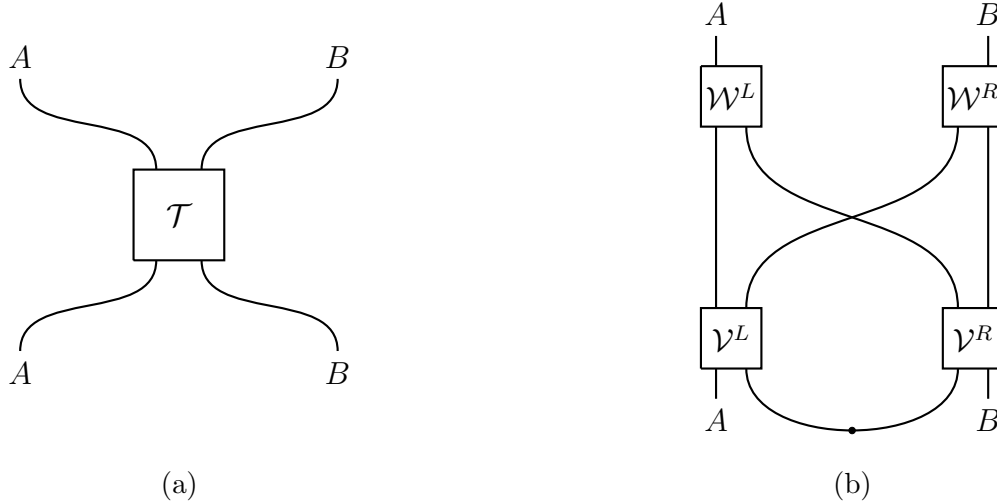


Figure 7.1: Local and non-local computations.

but at the cost of only considering protocols that complete f -routing or f -BB84 with only one-sided error. Second, we show a lower bound on the number of gates required to implement the non-local computation.

It is a striking feature of non-local quantum computation that these mostly classical local computations require large quantum entanglement to implement non-locally. Beyond this, establishing this quantum-classical separation is relevant for quantum position-verification (QPV) and for information theoretic cryptography. In a QPV scheme, we want to find operations that are as easy as possible for the honest player (who acts locally) to implement, but as hard as possible for a party of dishonest players, who implement an NLQC. We will discuss QVP in more detail shortly. Proving entanglement lower bounds for f -routing or f -BB84 gives a verification scheme where the honest player implements a nearly classical computation, while the dishonest player needs large quantum resources. This establishes the security of such QPV schemes under a bounded entanglement assumption. Our bound for entanglement only holds in the setting of one-sided error and hence not immediately relevant to practical schemes, but represents a step towards proving entanglement lower bounds in the robust setting.¹

Also relevant to our work is the connection between NLQC and information theoretic cryptography. In information theoretic cryptography, we wish to understand the

¹Many entanglement lower bounds apply in the robust setting, see e.g. [TFKW13, EFPS24, May22, BK11], but none grow with the classical input size. We achieve growth with classical input size at the expense of robustness.

communication and randomness cost to establish information theoretic privacy in various cryptographic settings. A primitive known as conditional disclosure of secrets (CDS) has been identified as one of the simplest settings where one can hope to understand the cost of privacy. CDS was initially studied in the context of private information retrieval [GIKM00], has applications in secret sharing [AA20], and shares a number of connections to communication complexity and interactive proofs [AV21]. In [ABM⁺23], f -routing was shown to be closely related to CDS: lower bounds on entanglement cost in f -routing give lower bounds on randomness in CDS. This opens a new avenue for quantum information techniques to make contributions to classical information theoretic cryptography.

A closely related topic to NLQC is position-verification. The subject of position-verification considers how to establish the spatial location of a party or object, by interacting with them remotely. Verifying position may be a cryptographic goal in itself, or a building block used for other cryptographic constructions. As well, position-verification has recently been understood to be closely connected with other primitives in information theoretic cryptography [ABM⁺23], topics in quantum gravity [May19, MPS20, May21, May22], to Hamiltonian simulation [ACH⁺24], and to uncloneable secret sharing [AGLL25].

In a position-verification scheme, the verifier sends the prover quantum and classical systems and asks for a reply at a set of designated spacetime locations. See Fig. 7.2 for a standard setup in a spacetime with one spatial dimension. In this figure, inputs are given at locations c_1, c_2 . The prover should apply a designated quantum operation to these inputs, then return the outputs to points r_1, r_2 . In Part (a), An honest prover enters the designated spacetime region (grey) to apply the needed quantum operation, and in Part (b), a dishonest prover attempts to reproduce the same operation while acting outside the spacetime region. This leads to the definition of a non-local quantum computation. The figure is reproduced from [May19].

It has been known that when the inputs and outputs are all classical, there is no unconditionally secure verification scheme [CGMO09]. This is because the prover can intercept the input signals, copy and forward them, and compute the expected replies without ever entering the designated spacetime region. Since the no-cloning theorem precludes this copy and forward attack with quantum information, using quantum inputs was suggested as a potential route to secure position-verification [KMSB06, KMS11, Mal16].

Even with quantum inputs, position-verification was proven insecure in the unconditional setting [BCF⁺14, BK11]. The attacks use entanglement distributed across a spacetime region to simulate operations that might otherwise need to be implemented inside of the region. Following this no-go result, focus has shifted to proving security under the assumption of bounded entanglement or communication, with a number of works

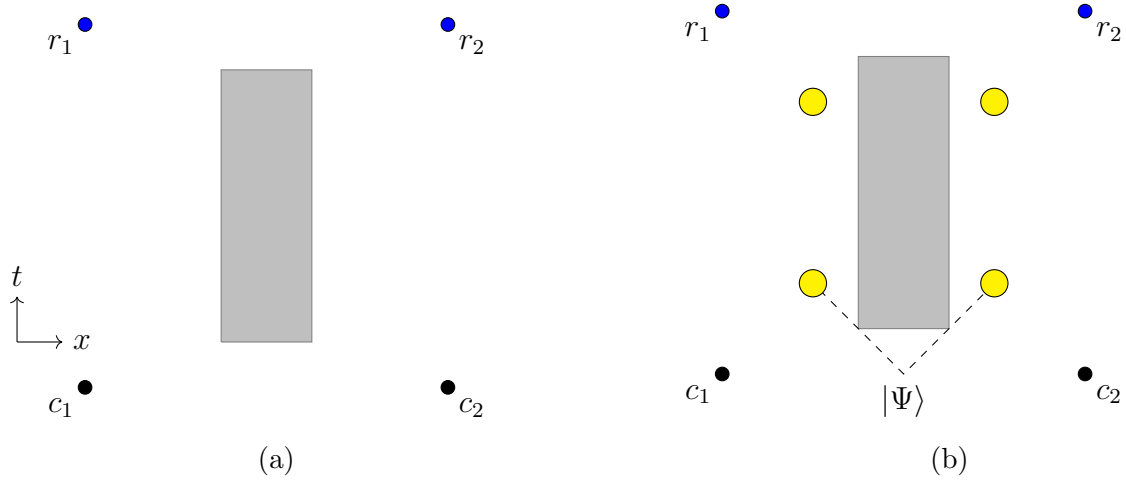


Figure 7.2: A position-verification scheme in $1 + 1$ dimensions.

establishing lower bounds [BK11, TFKW13, BCS22, May22, GC19] and upper bounds [BK11, BFSS13, CM22, GC19, Spe16, CL15] on entanglement requirements. Note that some of the existing lower bounds only bound the size of the resource system, rather than any measure of entanglement, or only apply in the setting where the communication must be classical. Another approach is to assume physical integrity of a device, which could contain a secret key [KMS11, CKPG23].

In the bounded entanglement setting, particular attention has been paid to classes of protocols where most of the input is classical, with just $O(1)$ quantum bits involved, and in particular to schemes where an honest prover need only compute a classical function and do $O(1)$ quantum operations. In this context, it has been hoped that the quantum resource requirements would grow with the classical input size, so that a dishonest prover would need large quantum resources. Security of these schemes would then be based on the assumption that quantum resources are more difficult to prepare and implement than classical ones.

In a recent work, this hope was partly realized [BCS22]. The authors study two schemes referred to as f -routing and f -BB84. In these schemes, an honest prover needs to compute a Boolean function f , then perform $O(1)$ quantum operations conditional on the value of f . For these tasks [BCS22] considers protocols that act unitarily on a shared resource system plus the inputs in the first round. In this setting, they prove that, with high probability over random choices of f , a dishonest prover needs to use a resource system composed of q qubits with q bounded below linearly in the number of classical input bits. Of the two variants for which this bound is proven, f -BB84 has the additional property that it can

be made fully loss tolerant [ABB⁺23]. This means that in experimental implementations, where most photons sent over long distances are lost, the f -BB84 protocol maintains a linear lower bound.

While attractive, two important caveats remain in the practicality of these verification schemes and the applicability of this proof. First, as we discuss in more detail later on, the unitary view of [BCS22] means some resources that in a physical implementation can be classical are included in the system size they lower bound. Ideally, one would bound the quantum resources in the physical protocol, not the quantum resources in a purified (unitary) view of the protocol.

As well, the perspective of [BCS22] focuses on the distributed resources of the honest and dishonest provers but ignores the local computational resources. Considering this, we note that a random function, with overwhelming probability, is of exponential complexity. Thus in both f -routing and f -BB84 with random f , the honest prover needs exponential classical resources and $O(1)$ quantum resources, while the dishonest prover needs at least linear quantum resources and (at minimum) exponential classical resources, since they also need to compute f . From this perspective, the honest prover's actions are not much easier than the dishonest one, and in any case, a protocol requiring the computation of an exponential complexity function is not practical².

An interesting alternative given in [BCS22] is to use a low complexity function with a large communication complexity. For the inner product function, the authors prove a lower bound of $q = \Omega(\log n)$ with n the classical input size. While the computation complexity of the honest prover is now linear, in the input size, it still requires exponential complexity in the size of the quantum resource manipulated by the dishonest prover.

7.1 Contributions

7.1.1 Entanglement lower bounds

An f -routing task is defined by choosing a Boolean function $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$. In this task, Alice, on the left, receives a quantum system Q of $O(1)$ size and a classical string $x \in \{0, 1\}^n$. Bob, on the right, receives $y \in \{0, 1\}^n$. Their goal is to bring Q to Alice

²We can also compare this to [LLQ21], which requires the honest prover have a polynomial time quantum computer and allows the use of only classical communication. From a computational perspective this is *more* feasible than f -BB84 or f -routing scheme with a random choice of function, since BQP is weaker than EXP.

if $f(x, y) = 0$, and to Bob if $f(x, y) = 1$. In the setting of non-local computation, Alice and Bob may only use shared entanglement and a single simultaneous round of communication to accomplish this. A difficulty in accomplishing this task arises since quantum mechanics prevents the system in register Q from being copied, and hence apparently Alice is forced to decide alone whether to keep or send Q . Furthermore, she doesn't know where Q is needed until learning y , at which point the single round of communication has already passed.

Despite the difficulty, the f -routing task can be completed for arbitrary functions f , with the most efficient known protocols using $2^{\Theta(\sqrt{n \log n})}$ entanglement and communication for arbitrary functions [ABM⁺23], and efficient protocols known for some low complexity classes of functions [CM23, BFSS13]. So far, only an $\Omega(1)$ lower bound is known on entanglement in f -routing. Here, we prove a new lower bound on f -routing with one-sided perfection.

We define a $(\varepsilon_0, \varepsilon_1)$ -correct f -routing scheme as one which brings Q to the left with fidelity at least $1 - \varepsilon_0$ when $(x, y) \in f^{-1}(0)$, and brings Q to the right with fidelity at least $1 - \varepsilon_1$ when $(x, y) \in f^{-1}(1)$.

Theorem 7.1.1 (Informally stated; see Theorem 9.1.5). *Define $\text{FR}_0(f)$ as the log of the minimal Schmidt rank of any entangled state that suffices to complete f -routing with $\varepsilon_0 = 0$, $\varepsilon_1 > 0$ for the function f , and $\text{FR}_1(f)$ as the analogous quantity with $\varepsilon_0 > 0$, $\varepsilon_1 = 0$. We have the following lower bounds.*

$$\begin{aligned} \text{FR}_0(f) &\geq \frac{1}{4} \log \text{rank}(g|_f) , \\ \text{FR}_1(f) &\geq \frac{1}{4} \log \text{rank}(g|_{-f}) , \end{aligned}$$

where $g|_f$ is a real-valued function that satisfies $f(x, y) = 0$ if and only if $g(x, y) = 0$, and $f(x, y) = 1$ if and only if $g(x, y) > 0$.

For some natural functions, this provides explicit strong lower bounds. For instance, taking f to be the equality function EQ , we are assured $g(x, y)$ is diagonal with non-zero entries on the diagonal and hence is of full rank, therefore $\text{FR}_0(EQ) \geq n/4$. Similarly, we obtain linear lower bounds on $\text{FR}_1(NEQ)$ where NEQ is the non-equality function.

As mentioned before, f -routing is related closely to the conditional disclosure of secrets (CDS) primitive. A CDS instance is defined by a choice of Boolean function $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$. In CDS, Alice and Bob receive inputs $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^n$ respectively, while Alice additionally holds a secret string $s \in \{0, 1\}^k$. Alice and Bob share

randomness, and send messages to a referee. Their goal is for the referee to be able to determine s if and only if $f(x, y) = 1$.

In [ABM⁺23], it was shown that the randomness cost of CDS for a function f upper bounds the entanglement cost of an f -routing scheme for the same function. Because of this, our lower bounds on entanglement in f -routing imply the same lower bounds on randomness in CDS. Since we only need one sided perfection for our bounds to apply, it is possible to bound CDS with imperfect correctness and perfect privacy (ppCDS), or perfect correctness but imperfect privacy (pcCDS). We obtain the following.

$$\begin{aligned} \text{ppCDS}(f) &\geq \Omega(\log \text{rank}(g|_f)) , \\ \text{pcCDS}(f) &\geq \Omega(\log \text{rank}(g|_{\neg f})) . \end{aligned}$$

The expressions on the left hand side denote randomness costs in the perfectly private and perfectly correct scenarios, respectively. This provides a new technique for bounding randomness in CDS, which is proven using the quantum information perspective, but can be phrased purely in terms of classical quantities.

We also prove lower bounds against the f -BB84 class of non-local quantum computations. An f -BB84 task is defined by choosing a Boolean function $f(x, y)$ with $x \in \{0, 1\}^n$ given to Alice and $y \in \{0, 1\}^n$ given to Bob. The system Q is in one of the BB84 states $\{H^{f(x,y)} |b\rangle\}$, with $b \in \{0, 1\}$ and $f(x, y) \in \{0, 1\}$. Alice and Bob should both output b .

The f -BB84 task can be completed for arbitrary functions f , with the most efficient known protocols using $2^{\Theta(n)}$ entanglement and communication for arbitrary functions [BFSS13].³ Again, we consider separate error rates in 0 and 1 instances; a $(\varepsilon_0, \varepsilon_1)$ -correct f -BB84 protocol produces b on both sides with probability $1 - \varepsilon_0$ when $f(x, y) = 0$, and produces b on both sides with probability $1 - \varepsilon_1$ when $f(x, y) = 1$. We prove the following lower bounds on the log of the Schmidt rank of any resource state that suffices to complete f -BB84 with one-sided error, so that one of ε_0 or ε_1 is zero.

Theorem 7.1.2 (Informally stated; see Theorem 9.2.6). *Define $\text{FBB84}_0(f)$ as the log of the minimal Schmidt rank of any entangled state that suffices to complete f -routing with $\varepsilon_0 = 0$, $\varepsilon_1 > 0$ for the function f , and $\text{FBB84}_1(f)$ as the analogous quantity with*

³Note that [BFSS13] deals with the f -routing task, but their protocol is easily adapted to f -BB84.

$\varepsilon_0 > 0, \varepsilon_1 = 0$. We have the following lower bounds.

$$\begin{aligned} \text{FBB84}_0(f) &\geq \frac{1}{4}(\log \text{rank}(g|_f) - 1) , \\ \text{FBB84}_1(f) &\geq \frac{1}{4}(\log \text{rank}(g|_{\neg f}) - 1) , \end{aligned}$$

where again $g|_f$ is a function with zeros exactly at the zeros of f and is positive otherwise.

In a practical setting, f -BB84 is of particular interest because it can be made loss-tolerant [ABB⁺23], and doesn't require quantum communication from the honest prover to the verifier. Unfortunately, our bound does not carry over to the noisy setting, so is not of immediate practical value. Nonetheless, our bound is the first non-trivial entanglement bound on f -BB84. Similar to the case for f -routing, for f -BB84 the bound above leads to linear lower bounds for the equality, non-equality, set-disjointness, and greater-than functions.

We can better understand our lower bounds by relating them to communication complexity. In a quantum communication complexity protocol, Alice holds $x \in \{0, 1\}^n$, Bob holds $y \in \{0, 1\}^n$, and Alice and Bob communicate qubits to determine $f(x, y)$. The number of qubits of communication needed for Alice and Bob to output 1 with non-zero probability when $f(x, y) = 1$, and never output 1 when $f(x, y) = 0$ is known as the quantum non-deterministic communication complexity, $\text{QNP}^{\text{cc}}(f)$. Using the characterization of QNP^{cc} in terms of the non-deterministic rank [dW00, dW03], we obtain

$$\begin{aligned} \log \text{rank } g|_f &\geq (\text{QNP}^{\text{cc}}(f) - 1) , \\ \log \text{rank } g|_{\neg f} &\geq (\text{coQNP}^{\text{cc}}(f) - 1) . \end{aligned}$$

This relates our lower bounds on CDS, f -routing, and f -BB84 to quantum communication complexity.

7.1.2 Gate lower bounds

In Chapter 10, we focus on the computational requirements of the honest and dishonest players and give a new bound against f -routing and f -BB84. Our bound follows from an extension of a lower bound strategy used in [BCS22]. We show that given a function f , a successful attack on f -BB84 or f -routing requires a dishonest player to implement a number of quantum gates linear in the simultaneous message passing (*SMP*) cost of f , where we allow shared entanglement in the *SMP* scenario. As a concrete example, this

places a linear lower bound in the input size against the number of quantum gates needed by a dishonest player to implement the inner product function. Meanwhile, the honest player can implement $O(1)$ quantum gates and only a linear number of classical gates. Our bound is also robust, applying whenever the honest player succeeds with fidelity at least 0.89 for f -BB84 or at least 0.90 for f -routing on a constant fraction of the inputs. Furthermore, our bound applies to the loss tolerant variant of f -BB84 studied in [ABB⁺23]. This allows us to combine the advantages of our scheme with full loss tolerance.

Our lower bound technique builds on a reduction from f -routing and f -BB84 to simultaneous message passing (SMP) introduced in [BCS22]. In simultaneous message passing, two players receive inputs x and y and send messages to a referee that should determine $f(x, y)$. In f -routing (and similarly f -BB84), a quantum system Q is brought left or right based on the value of $f(x, y)$. In relating these two settings, recall that a dishonest prover in f -routing has two agents, who intercept the input signals. These become the two players in the SMP scenario. The key idea to reduce f -routing to SMP is to show that any successful attack on f -routing has a state after the first round of operations that determines whether the input Q is brought left or brought right. This means that this state determines the value of $f(x, y)$. The reduction works by having the players communicate data in an SMP protocol that is sufficient to reproduce this state to a referee. The referee can then determine if this is a state with Q on the left or Q on the right, and hence determine $f(x, y)$.

Our main technical contribution is to adapt this reduction to lower bound the number of quantum gates and measurements applied by a dishonest prover, rather than to bound the dimensionality of their resource system as was done already in [BCS22]. Heuristically, a simple f -routing protocol would lead to a simple description of how to prepare this state, and hence to a good SMP , so lower bounds on SMP lead to lower bounds on the complexity of the f -routing protocol. Importantly, we are interested in bounding the number of quantum operations performed by the prover, while allowing free classical processing. Direct use of the reduction from [BCS22] would instead bound the total number of classical and quantum operations. To obtain a bound on quantum operations alone requires we modify the reduction to SMP , and in particular reduce to simultaneous message passing with shared entanglement allowed between the players.

The general form of the bound we obtain is stated below.

Theorem 7.1.3 (Informally stated; see Theorem 10.3.5). *In a non-local f -routing or f -BB84 protocol for a function $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, let q be the number of qubits held by each player, $C_M(f)$ be the number of measurements they jointly make, and $C_G(f)$ is the number of gates they jointly apply. Denote by $SMP_{(\epsilon, \delta)}^*(f)$ the communication cost in the*

simultaneous message passing model with shared entanglement allowed, where we require $1 - \epsilon'$ correctness on a $(1 - \delta)$ -fraction of the inputs. We then have

$$(\log(q) + 1)(2C_G(f) + C_M(f)) \geq SMP_{(\epsilon', \delta)}^*(f) .$$

This is stated assuming the gate set is Clifford + T, but is easy to adapt to any gate set.

In our bound, Importantly, the bound holds even when allowing free classical processing, including the use of mid-circuit measurements and classical computations that make use of those mid-circuit measurement outcomes.

We then exploit linear lower bounds on SMP^* for the inner product function IP given in [CVDNT98] to prove an explicit lower bound,

$$(\log q + 1)(2C_G(IP) + C_M(IP)) \geq \Omega(n) .$$

Here, n is the number of input bits to the inner product function. This bound on quantum gates and measurements holds when requiring the protocol succeed on only a fraction $1/2 + \epsilon(n)$ of the inputs, with $\epsilon(n)$ going to zero slower than $2^{-n/4}$.

Chapter 8

Preliminaries

In this chapter, we give the definitions and notations that we will use in the rest of the thesis. In particular, we need definitions from quantum information, communication complexity, and non-local quantum computation which we will present in the following sections.

8.1 Notation and definitions

We assume basic familiarity with quantum computation and information, and refer the interested reader to standard textbooks such as [NC10].

We let Ψ_{AB}^+ denote the maximally entangled state on AB , so that

$$|\Psi^+\rangle_{AB} = \frac{1}{\sqrt{d}} \sum_{i=1}^d |i\rangle_A |i\rangle_B \quad ,$$

and $\Psi_{AB}^+ = |\Psi^+\rangle\langle\Psi^+|_{AB}$.

Given two density matrices ρ, σ , we define the fidelity by

$$F(\rho, \sigma) = \text{tr} \left(\sqrt{\sqrt{\sigma} \rho \sqrt{\sigma}} \right) \quad ,$$

so that for pure states, with a slight abuse of notation, we get $F(|\psi\rangle, |\phi\rangle) = |\langle\psi|\phi\rangle|$.

Define the purified distance as

$$P(\rho, \sigma) = \sqrt{1 - (F(\rho, \sigma))^2} .$$

Note that this is a distance and satisfies the triangle inequality. We also use the trace norm,

$$\|\rho - \sigma\|_1 \equiv \text{tr} \sqrt{(\rho - \sigma)^\dagger (\rho - \sigma)} .$$

We will make use of the complementary information trade-off (CIT) inequality [RB09], which we state below.

Theorem 8.1.1 ([RB09]). *Let $|\psi\rangle_{REF}$ be an arbitrary tripartite state, with R a single qubit. We consider measurements on the R system that produce a measurement result we store in a register Z . We consider measurements in both the computational and Hadamard basis, and denote the post-measurement state when measuring in the computational basis by ρ_{ZEF} , and when measuring in the Hadamard basis by σ_{ZEF} . Then,*

$$H(Z|E)_\rho + H(Z|F)_\sigma \geq 1 .$$

The fidelity is related to the one-norm distance by the Fuchs–van de Graaf inequalities,

$$1 - F(\rho, \sigma) \leq \frac{1}{2} \|\rho - \sigma\|_1 \leq \sqrt{1 - (F(\rho, \sigma))^2} .$$

The von Neumann entropy is defined by

$$S(A) = -\text{tr}(\rho \log \rho) ,$$

and the quantum mutual information is defined by

$$I(A : B)_\rho = S(A)_\rho + S(B)_\rho - S(AB)_\rho ,$$

and the conditional entropy is defined by

$$S(A|B)_\rho = S(AB)_\rho - S(B)_\rho .$$

We will also make use of the following statement of continuity of the conditional entropy [Win16],

Theorem 8.1.2 ([Win16]). *Suppose that*

$$\frac{1}{2} \|\rho_{AB} - \sigma_{AB}\|_1 \leq \varepsilon ,$$

and let $h(x) = -x \log x - (1-x) \log(1-x)$ be the binary entropy function. Then

$$|S(A|B)_\rho - S(A|B)_\sigma| \leq 2\varepsilon \log d_A + (1+\varepsilon)h\left(\frac{\varepsilon}{1+\varepsilon}\right) .$$

We will also make use of the following observation.

Observation 8.1.3. *Let ρ_{AB} be a density matrix on the AB system, d be the dimension of subsystem A , and ρ_B be the marginal of ρ_{AB} on the subsystem B . Then we have*

$$\begin{aligned} \text{tr}\left(\rho_{AB} - \frac{\mathcal{I}_A}{d} \otimes \rho_B\right)^2 &= \text{tr}\left(\rho_{AB}^2 + \frac{\mathcal{I}_A}{d^2} \otimes \rho_B^2 - 2\rho_{AB}\left(\frac{\mathcal{I}_A}{d} \otimes \rho_B\right)\right) \\ &= \text{tr}(\rho_{AB}^2) + \frac{1}{d^2} \text{tr}(\mathcal{I}_A) \text{tr}(\rho_B^2) - \frac{2}{d} \text{tr}_B(\rho_{AB}^2) \\ &= \text{tr}(\rho_{AB}^2) - \frac{1}{d} \text{tr}(\rho_B^2) \\ &= \text{tr}(\rho_{AB}^2) - \text{tr}\left(\frac{\mathcal{I}_A}{d^2} \otimes \rho_B^2\right) . \end{aligned}$$

8.2 Communication complexity

Again, we assume the reader is familiar with the basics of communications complexity, and refer to the standard textbooks such as [KN96, RY20] for more background.

In the later chapters, we will make use of a reduction from f -routing and f -BB84 to communication complexity scenarios. Specifically, we will be interested in the *simultaneous message passing (SMP)* and *one-way communication scenarios*.

A simultaneous message passing scenario is defined by a choice of function $f : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$. The scenario involves three parties, Alice, Bob, and the referee. Alice receives $x \in \{0,1\}^n$ and Bob receives $y \in \{0,1\}^n$. Alice and Bob compute messages m_A, m_B from their local resources, including shared randomness, and the inputs they receive, and send their messages to the referee. Alice and Bob succeed if the referee can compute $f(x,y)$ from their messages. We define the *SMP* cost of f , denoted $SMP(f)$ to be $\min_P \max\{|m_A|, |m_B|\}$ where the minimization is over choices of protocols.

There are several variations of the basic *SMP* scenario. For example, we can allow the referee to only succeed with some probability taken over the shared randomness and selection of inputs. We will always assume the input distribution is uniform in this thesis. We denote the *SMP* cost in the case where they succeed with probability $1 - \varepsilon$ on at least $(1 - \delta)$ -fraction of inputs by $SMP_{(\varepsilon, \delta)}(f)$. We can also allow Alice and Bob to share entanglement as opposed to just classical randomness, and/or to send quantum messages. Our focus in this work is on the case where the messages are classical but they share entanglement. In this case, we denote the *SMP* cost by $SMP_{(\varepsilon, \delta)}^*(f)$.

A formal definition follows.

Definition 8.2.1 ((ε, δ) -*SMP* complexity). *Let $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be a function, and $\varepsilon, \delta \in [0, 1]$ be parameters. An *SMP* protocol P for f consists of three algorithms Alice, Bob, and a referee. Alice receives $x \in \{0, 1\}^n$ as input and outputs $m_A \in \{0, 1\}^*$, Bob receives $y \in \{0, 1\}^n$ as input and outputs $m_B \in \{0, 1\}^*$, and the referee receives m_A, m_B and outputs a bit $c = P(x, y)$. A protocol P is (ε, δ) -correct if there exists $S \subseteq \{0, 1\}^n \times \{0, 1\}^n$ such that $|S| \geq (1 - \delta) \cdot 2^{2n}$, and*

$$\forall (x, y) \in S : \Pr[P(x, y) = f(x, y)] \geq 1 - \varepsilon .$$

*The (ε, δ) -*SMP* complexity of f is defined as follows*

$$SMP_{(\varepsilon, \delta)}(f) = \min_{P: P \text{ is } (\varepsilon, \delta)\text{-correct}} \max\{|m_A|, |m_B|\} .$$

Similarly, we can define $SMP_{(\varepsilon, \delta)}^*(f)$ for the case where Alice and Bob share entanglement.

A second basic scenario is the one-way communication scenario. Here, there is no referee and Alice directly sends her message to Bob, who should succeed in computing $f(x, y)$ with probability at least $1 - \varepsilon$, on at least $(1 - \delta)$ -fraction of possible inputs.

Definition 8.2.2 ((ε, δ) -One-way communication complexity). *Let $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be a function, and $\varepsilon, \delta \in [0, 1]$ be parameters. A one-way communication protocol P for f consists of two algorithms Alice and Bob, where Alice receives $x \in \{0, 1\}^n$ as input and outputs $m_A \in \{0, 1\}^*$, and Bob receives $y \in \{0, 1\}^n$ and m_A as input and outputs a bit $c = P(x, y)$. As previously, the protocol is (ε, δ) -correct if there exists $S \subseteq \{0, 1\}^n \times \{0, 1\}^n$ with $|S| \geq (1 - \delta) \cdot 2^{2n}$ such that $\Pr[P(x, y) = f(x, y)] \geq 1 - \varepsilon$ for all $(x, y) \in S$. We focus on the case where Alice's message is classical but she shares entanglement with Bob.*

The (ε, δ) -one-way communication complexity of f is defined as follows

$$C_{\rightarrow, (\varepsilon, \delta)}(f) = \min_{P: P \text{ is } (\varepsilon, \delta)\text{-correct}} |m_A| .$$

Similarly, we can define $C_{\rightarrow,(\varepsilon,\delta)}^*(f)$ for the case where Alice and Bob share entanglement.

The two-way communication complexity allows back and forth messages between Alice and Bob. We can denote the minimal message size (we use the total number of bits sent by Alice and Bob) by $C_\varepsilon^*(f)$ when the success probability of protocol is at least $1 - \varepsilon$ (taken over the choice of inputs and internal randomness of the protocol).

An easy observation is that

$$SMP_{(\varepsilon,\delta)}^*(f) \geq C_{\rightarrow,(\varepsilon,\delta)}^*(f) \geq C_{\varepsilon'}^*(f) . \quad (8.1)$$

The first inequality follows because any SMP^* protocol can be turned into a one-way communication complexity scenario by having Alice send her message to Bob instead of the referee, and Bob to run the same computation as the referee would in the SMP^* . The second inequality follows because a one-way communication complexity protocol is immediately a two-way communication complexity protocol for the same function.

Remark 8.2.3. *Note that in the definition of $C_{\varepsilon'}^*(f)$ we are assuming average-case correctness, while our definitions of SMP and one-way complexity are more restricted. Nevertheless, one can set $\varepsilon' = \delta + (1 - \delta) \cdot \varepsilon$ to make sure the restricted case is as successful as the average-case scenario.*

A standard function studied in communication complexity is the mod 2 inner product function defined as

$$IP(x, y) = \sum_i x_i y_i \bmod 2 .$$

Intuitively, this is a difficult function to compute in communication complexity scenarios because the output depends sensitively on every bit of the input. More concretely, will make use of the following lower bound, proven in [CVDNT98].

$$C_\varepsilon^*(IP) \geq \max\left\{\frac{1}{2}(1 - 2\varepsilon)^2, (1 - 2\varepsilon)^4\right\}n - 1/2 .$$

We briefly comment on the proof of this theorem given in [CVDNT98]. While not explicitly stated there, an inspection of their proof reveals their lower bound applies in the average-case setting. In fact, it is only necessary for the protocol to work with probability $1 - \varepsilon$ over a uniform choice of input x for at least one fixed choice of input y , or over a uniform choice of input y at any fixed choice of x .

This lower bound is also improved in Corollary 4.3 of [NS02], showing that

$$C_\varepsilon^*(IP) \geq \max\left\{\frac{1}{2}n + 2\log(1 - 2\varepsilon), (1 - 2\varepsilon)^4 n - 1/2\right\} .$$

This has an important consequence in that even with $\varepsilon \rightarrow 1/2$ as $n \rightarrow \infty$, we obtain a $\Omega(n)$ lower bound, so long as ε goes to $1/2$ more slowly than $2^{-n/4}$.

Another measure of communication complexity which will be discussed later in this thesis and will be used to quantify out entanglement lower bounds is non-deterministic quantum communication, which is defined below.

Definition 8.2.4 (Non-deterministic quantum communication complexity). *Let $f : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be a function, and consider two algorithms, Alice and Bob, with Alice given input $x \in \{0, 1\}^n$ and Bob given input $y \in \{0, 1\}^n$. Alice and Bob exchange messages and Alice outputs a bit z . The non-deterministic quantum communication complexity $\text{QNP}^{\text{cc}}(f)$, introduced in [dW00], is the minimum number of qubits Alice and Bob need to exchange to both output 1 with probability strictly larger than zero when $f(x, y) = 1$, and both output 0 with probability 1 when $f(x, y) = 0$.*

A matrix M is called a non-deterministic communication complexity matrix for f exactly when $M(x, y) \neq 0$ when $f(x, y) = 1$. The non-zero entries are allowed to be any complex numbers. The minimal rank of any non-deterministic communication complexity matrix for f is denoted by $\text{nrnk}(f)$ and called the non-deterministic rank. We have the following result from [dW03].

Theorem 8.2.5 ([dW03]). *The non-deterministic rank and the non-deterministic quantum communication complexity are related by*

$$\text{QNP}^{\text{cc}}(f) = \lceil \log(\text{nrnk}(f)) \rceil + 1 .$$

8.3 f -BB84 and f -routing

We now move on to defining the two non-local quantum procedures that we study in this work, namely, f -BB84 and f -routing. The definitions of these tasks follows in the rest of this chapter.

8.3.1 f -BB84 definition and the strategy model

We give the following definition of a *qubit f -BB84* task. In this definition and the rest of the text, we refer to the two agents of the prover (who sit on the left and right of the grey region in Fig. 7.2b) as Alice and Bob. Because we are viewing cheating in the position-verification scenario as a form of a quantum game, which can be considered separately aside from the connection to position-verification, we also rename the role of the verifier as the referee.

Definition 8.3.1 (f -BB84). *A qubit f -BB84 task is defined by a choice of Boolean function $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, and a 2 dimensional Hilbert space $\mathcal{H}_Q$. Inputs $x \in \{0, 1\}^n$ and system Q are given to Alice, and input $y \in \{0, 1\}^n$ is given to Bob. The system Q is in the maximally entangled state with a reference system R .¹ Alice and Bob exchange one round of communication, with the combined systems received or kept by Alice labelled M and the systems received or kept by Bob labelled M' . Define projectors*

$$\Pi^{a,b} = H^a |b\rangle\langle b| H^a .$$

The referee will measure $\{\Pi^{f(x,y),0}, \Pi^{f(x,y),1}\}$ on the R system and find measurement outcome $b \in \{0, 1\}$. The qubit f -BB84 task is completed ε -correctly on input (x, y) if Alice and Bob both output b . More formally, Alice and Bob succeed if there exist POVM's $\{\Lambda_M^{x,y,0}, \Lambda_M^{x,y,1}\}, \{\Lambda_{M'}^{x,y,0}, \Lambda_{M'}^{x,y,1}\}$ such that,

$$\text{tr}\left(\Pi_R^{f(x,y),b} \otimes \Lambda_M^{x,y,b} \otimes \Lambda_{M'}^{x,y,b} \rho_{RMM'}\right) \geq 1 - \varepsilon .$$

Next, we give a fully general model capturing strategies that complete the f -BB84 task in the form of a non-local quantum computation.

1. Alice and Bob share a resource system $|\psi\rangle_{AB}$.
2. The referee prepares Ψ_{RQ}^+ and hands Q to Alice, preparing a joint state $|\psi\rangle_{RQAB}$.
3. At the same time as above, Alice receives $x \in \{0, 1\}^n$ and Bob receives $y \in \{0, 1\}^n$.

¹Notice that compared to the description of f -BB84 given in the introduction, we now have the referee give Alice and Bob one end of a maximally entangled state and then later measure the reference system. The referee's measurement is chosen such that the post-measurement state is one of the BB84 states, coinciding with the description in the introduction. The two formulations are equivalent.

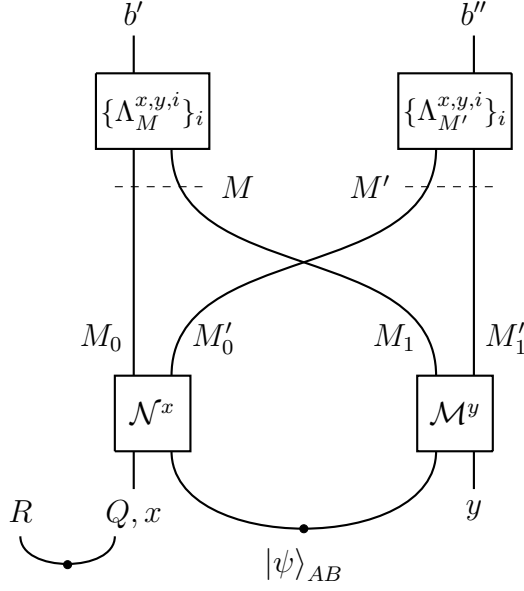


Figure 8.1: General NLQC strategy for f -BB84.

4. Alice applies $\mathcal{N}_{QA \rightarrow M_0 M'_0}^x$, Bob applies $\mathcal{M}_{B \rightarrow M_1 M'_1}^y$. Label $M = M_0 M_1$, $M' = M'_0 M'_1$ so that their joint state after the first round is

$$\rho_{RMM'} = \mathcal{N}_{QA \rightarrow M_0 M'_0}^x \otimes \mathcal{M}_{B \rightarrow M_1 M'_1}^y (|\psi\rangle\langle\psi|_{RQAB}) .$$

5. M_0 and M_1 are sent to Alice, so that she holds M . At the same time, M'_0 and M'_1 are sent to Bob so that he holds M' . The (classical) inputs x and y are copied and sent to both parties.
6. Alice, Bob and the referee all compute $f(x, y)$. The referee measures R in the $f(x, y)$ basis, obtaining measurement outcome b . Alice and Bob apply POVMs $\{\Lambda_M^{x,y,0}, \Lambda_M^{x,y,1}\}$ and $\{\Lambda_{M'}^{x,y,0}, \Lambda_{M'}^{x,y,1}\}$, then both output their measurement outcomes.

Recall that Alice and Bob succeed when they both obtain outcome b . See [Fig. 8.1](#) for an illustration of a general protocol for f -BB84. In this figure, Alice applies $\mathcal{N}^x$ in the first round, Bob applies $\mathcal{M}^y$. They communicate in one simultaneous exchange, and then apply measurements to their local systems. They succeed if $b = b' = b''$, with b determined by measuring a reference R maximally entangled with Q .

8.3.2 f -routing definition and the strategy model

We start by giving the following definition of a *qubit f -routing* task.

Definition 8.3.2 (f -routing). *A qubit f -routing task is defined by a choice of Boolean function $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, and a 2 dimensional Hilbert space $\mathcal{H}_Q$. Inputs $x \in \{0, 1\}^n$ and system Q are given to Alice, and input $y \in \{0, 1\}^n$ is given to Bob. Alice and Bob exchange one round of communication, with the combined systems received or kept by Alice labelled M and the systems received or kept by Bob labelled M' . Label the combined actions of Alice and Bob in the first round as $\mathcal{N}_{Q \rightarrow MM'}^{x,y}$. The qubit f -routing task is completed ε -correctly on an input (x, y) if Alice can recover Q when $f(x, y) = 0$ and Bob can recover Q when $f(x, y) = 1$. More formally, the protocol is ε -correct if there exists a channel $\mathcal{D}_{M \rightarrow Q}^{x,y}$ such that*

$$\text{when } f(x, y) = 0, \quad P(\mathcal{D}_{M \rightarrow Q}^{x,y} \circ \text{tr}_{M'} \circ \mathcal{N}_{Q \rightarrow MM'}^{x,y}(\Psi_{RQ}^+), \Psi_{RQ}^+) \leq \varepsilon ,$$

and there exists a channel $\mathcal{D}_{M' \rightarrow Q}^{x,y}$ such that

$$\text{when } f(x, y) = 1, \quad P(\mathcal{D}_{M' \rightarrow Q}^{x,y} \circ \text{tr}_M \circ \mathcal{N}_{Q \rightarrow MM'}^{x,y}(\Psi_{RQ}^+), \Psi_{RQ}^+) \leq \varepsilon .$$

Next, we give a fully general model capturing strategies that complete the f -routing task in the form of a non-local quantum computation.

1. Alice and Bob share a resource system $|\psi\rangle_{AB}$ with A held by Alice and B held by Bob.
2. The referee prepares Ψ_{RQ}^+ and hands Q to Alice.
3. At the same time as above, Alice receives $x \in \{0, 1\}^n$ and Bob receives $y \in \{0, 1\}^n$.
4. Alice applies $\mathcal{N}_{QA \rightarrow M_0 M'_0}^x$, Bob applies $\mathcal{M}_{B \rightarrow M_1 M'_1}^y$. Label $M = M_0 M_1$, $M' = M'_0 M'_1$ so that their joint state after the first round is

$$\rho_{RMM'} = \mathcal{N}_{QA \rightarrow M_0 M'_0}^x \otimes \mathcal{M}_{B \rightarrow M_1 M'_1}^y (|\psi\rangle\langle\psi|_{AB}) .$$

5. M_0 and M_1 are sent to Alice, so that she holds M . At the same time, M'_0 and M'_1 are sent to Bob so that he holds M' . The inputs x and y are copied and sent to both parties.

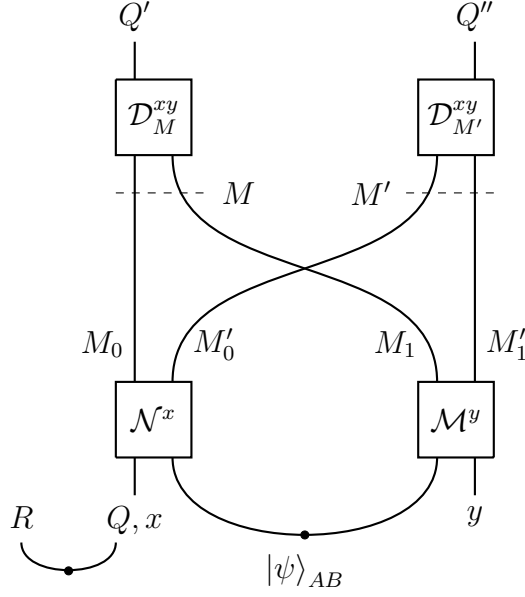


Figure 8.2: General NLQC strategy for f -routing.

6. Alice and Bob both compute $f(x, y)$. If $f(x, y) = 0$, Alice applies a channel $\mathcal{D}_{M \rightarrow Q}^{x, y}$ and returns Q to the referee. If $f(x, y) = 1$, Bob applies a channel $\mathcal{D}_{M' \rightarrow Q}^{x, y}$ and returns Q to the referee.

Fig. 8.2 shows an illustration of this general strategy. The first round operations are the same as f -BB84. In the second round, Alice and Bob apply channels mapping to qubit systems Q', Q'' . If $f(x, y) = 0$ Q' should be maximally entangled with the reference R . If $f(x, y) = 1$ then Q'' should be maximally entangled with R .

Chapter 9

Entanglement Lower Bounds

In this chapter, we give the proofs for our lower bounds on entanglement for f -BB84 and f -routing, where f is an arbitrary function. Our lower bounds are stated in terms of the function rank, and we also show connections to non-deterministic communication complexity of f . We close this chapter by showing how these techniques yield lower bounds for conditional disclosure of secrets, and we also apply our lower bounds to some explicit function to obtain quantitative bounds.

9.1 Lower bounds for f -routing

Let $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$ be an arbitrary Boolean function, and recall the definition of f -routing from [Definition 8.3.2](#). We will focus on f -routing with $\varepsilon_0 = 0$, $\varepsilon_1 \geq 0$ so that the protocol is perfectly correct on zero instances of $f(x, y)$, or with $\varepsilon_0 > 0$ and $\varepsilon_1 = 0$ so that the protocol is perfectly correct on 1 instances. We define the entanglement cost of an f -routing protocol to be the log of the minimal Schmidt rank of any resource system that can be used to perform the f -routing task. In notation, we define $\text{FR}_0(f)$ to be entanglement cost for f -routing with $\varepsilon_0 = 0$, $\varepsilon_1 = 0.05$, $\text{FR}_1(f)$ to be the entanglement cost when $\varepsilon_0 = 0.05$, $\varepsilon_1 = 0$.

Towards a lower bound in this setting, we first observe that $\rho_{RM'} = \rho_R \otimes \rho_{M'}$ iff $f(x, y) = 0$. We show this in the following lemma.

Lemma 9.1.1. *Suppose an f -routing protocol is perfectly correct on zero instances, and $\varepsilon_1 < \varepsilon_1^*$ correct on 1 instances, where ε_1^* depends on d_Q but is lower bounded by a constant.*

Then the mid-protocol state it produces, $\rho_{RM'}$, satisfies $\rho_{RM'} = \rho_R \otimes \rho_{M'}$ if and only if $f(x, y) = 0$.

Proof. By correctness on 0 instances, we have that

$$\text{when } f(x, y) = 0, \quad F(\mathcal{D}_{M \rightarrow Q}^{x, y} \circ \text{tr}_{M'} \circ \mathcal{N}_{Q \rightarrow MM'}^{x, y}(\Psi_{RQ}^+), \Psi_{RQ}^+) = 1 \quad ,$$

hence $f(x, y) = 0$ implies Alice can produce a Bell state $|\Psi^+\rangle_{RQ}$. By the data processing inequality, we have

$$2n = I(R : Q)_{\Psi_{RQ}^+} \leq I(R : M)_{\rho_{RM}} \quad ,$$

but we also have the inequality

$$I(R : M)_{\rho} + I(R : M')_{\rho} \leq 2S(R) \quad ,$$

so that $I(R : M') = 0$, which implies

$$\rho_{RM'} = \rho_R \otimes \rho_{M'} \quad ,$$

as needed. Conversely, if $f(x, y) = 1$ we cannot have $\rho_{RM'} = \rho_R \otimes \rho_{M'}$. To see why, first recall that f -routing which is ε_1 -correct on 1 instances has that there exists a family of channels $\{\mathcal{D}_{M' \rightarrow Q}^{x, y}\}_{x, y}$ such that

$$\text{when } f(x, y) = 1, \quad F(\mathcal{D}_{M' \rightarrow Q}^{x, y} \circ \text{tr}_M \circ \mathcal{N}_{Q \rightarrow MM'}^{x, y}(\Psi_{RQ}^+), \Psi_{RQ}^+) \geq 1 - \varepsilon_1 \quad .$$

Define

$$\sigma_{RQ} = \mathcal{D}_{M' \rightarrow Q}^{x, y}(\rho_{RM'}) \quad ,$$

and use that by data processing inequality we have,

$$I(R : M')_{\rho_{RM}} \geq I(R : Q)_{\sigma_{RQ}} \quad .$$

But then by the Fuchs–van de Graaf inequalities σ_{RQ} is close in trace distance to Ψ_{RQ}^+ ,

$$\frac{1}{2} \|\Psi_{RQ}^+ - \sigma_{RQ}\|_1 \leq \sqrt{2\varepsilon_1 - \varepsilon_1^2} \equiv \mu \quad ,$$

and by continuity of the conditional entropy,

$$\begin{aligned} I(R : Q)_{\Psi^+} - I(R : Q)_{\sigma_{RQ}} &= S(R|Q)_{\Psi^+} - S(R|Q)_{\sigma} \\ &\leq 2\mu \log d_R + (1 + \mu)h\left(\frac{\mu}{1 + \mu}\right) , \end{aligned}$$

so that

$$2 \log d_R - 2\mu \log d_R + (1 + \mu)h\left(\frac{\mu}{1 + \mu}\right) \leq I(R : Q)_{\sigma} .$$

We have that σ is not a tensor product whenever the left hand side of the equation above is strictly positive. For $d_Q = 2$, this occurs for $\varepsilon_1 > \varepsilon_1^* \approx 0.08$, and ε_1^* approaches 1 as $d_Q \rightarrow \infty$. $\square$

Next, we define what we call a *structure function* for a protocol. The structure function is computed from the mid-protocol density matrix $\rho_{RMM'}$, and as we will see, captures the zeros in $f(x, y)$.

Definition 9.1.2. *Given an f -routing protocol with mid-protocol density matrix $\rho_{RMM'}$, define the structure function $g(x, y)$ according to*

$$g(x, y) = \text{tr} \left(\rho_{RM'} - \frac{\mathcal{I}}{d_R} \otimes \rho_{M'} \right)^2 .$$

We claim that $g(x, y)$ captures some aspect of the structure in the function f which must be present in a correct f -routing protocol. More concretely we have the following.

Lemma 9.1.3. *In a perfectly correct f -routing protocol, the structure function $g(x, y)$ is zero if and only if $f(x, y) = 0$.*

Proof. This follows because $g(x, y) = 0$ is zero if and only if $\rho_{RM'} = \frac{\mathcal{I}}{d_R} \otimes \rho_{M'}$, and [Lemma 9.1.1](#) shows we have this tensor product form if and only if $f(x, y) = 0$. $\square$

Our next job is to relate the function $g(x, y)$ to the entanglement available to Alice and Bob. We prove the following lemma.

Lemma 9.1.4. *An f -routing protocol that uses a resource system with Schmidt rank d_E has a structure function of the form*

$$g(x, y) = \sum_{I=1}^{d_E^4} f_I(x) f'_I(y) .$$

Proof. From the general form of a non-local quantum computation protocol, the density matrix $\rho_{RM'_0M'_1}(x, y)$ can be expressed as

$$\rho_{RM'_0M'_1} = \mathcal{N}_{QA \rightarrow M'_0}^x \otimes \mathcal{M}_{B \rightarrow M'_1}^y (\Psi_{RQ}^+ \otimes |\psi\rangle\langle\psi|_{AB}) .$$

We will write $|\psi\rangle_{AB}$ in the Schmidt basis,

$$|\psi\rangle_{AB} = \sum_{i=1}^{d_E} |i\rangle_A |i\rangle_B ,$$

with un-normalized vectors $|i\rangle_A, |i\rangle_B$. Then we get

$$\begin{aligned} \rho_{RM'_0M'_1} &= \sum_{i,j=1}^{d_E} \mathcal{N}_{QA \rightarrow M'_0}^x (\Psi_{RQ}^+ \otimes |i\rangle\langle j|_A) \otimes \mathcal{M}_{B \rightarrow M'_1}^y (|i\rangle\langle j|_B) \\ &= \sum_{i,j=1}^{d_E} A_{RM'_0}^{x,i,j} \otimes B_{M'_1}^{y,i,j} . \end{aligned}$$

We can also compute the trace over R where we define A and B in the second line of the previous equation and get

$$\rho_{M'_0M'_1} = \sum_{i,j=1}^{d_E} A_{M'_0}^{x,i,j} \otimes B_{M'_1}^{y,i,j} .$$

Next, we compute $g(x, y)$. It is convenient to make use of [Observation 8.1.3](#) to write

$$g(x, y) = \text{tr} \left(\rho_{RM'}^2 - \frac{\mathcal{I}}{d_R^2} \otimes \rho_{M'}^2 \right) .$$

Inserting the forms of $\rho_{RM'}$ and $\rho_{M'}$ into this, we obtain

$$\begin{aligned} g(x, y) &= \sum_{i,j,i',j'=1}^{d_E} \text{tr} \left(\left(A_{QM'_0}^{x,i,j} A_{QM'_0}^{x,i',j'} - \frac{\mathcal{I}}{d_Q^2} \otimes A_{M'_0}^{x,i,j} A_{M'_0}^{x,i',j'} \right) \otimes B_{M'_1}^{y,i,j} B_{M'_1}^{y,i',j'} \right) \\ &= \sum_{I=1}^{d_E^4} f_I(x) f'_I(y) \ , \end{aligned}$$

as needed. □

We can view $g(x, y)$ as a matrix, and $f_I(x)$, $f'_I(y)$ as vectors so that the minimal number of terms appearing in this sum is the rank of the matrix $g(x, y)$. Thus, we obtain the main results of this section, stated as the following theorem.

Theorem 9.1.5. *For any f -routing protocol, where $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, that is $(\varepsilon_0, \varepsilon_1) = (0, 1/20)$ -correct, we have*

$$\text{FR}_0(f) \geq \frac{1}{4} \log \text{rank } g|_f \ ,$$

where g is zero and non-zero for the same inputs as f , which is denoted by $g|_f$.

Furthermore, reversing the role of M and M' , and assuming $(\varepsilon_0, \varepsilon_1) = (1/20, 0)$ -correctness, we can obtain

$$\text{FR}_1(f) \geq \frac{1}{4} \log \text{rank } g|_{\neg f} \ ,$$

where $\neg f$ is the negation of f .

We can also rephrase the above bounds in terms of the non-deterministic quantum communication complexity. Observe that

$$\text{rank } g|_f \geq \text{nrank}(f) \ . \tag{9.1}$$

Because the non-deterministic rank considers matrices with arbitrary complex values where $f(x, y) = 1$, while $g|_f$ has the additional constraint that it has positive real values when $f(x, y) = 1$, it is possible that there are functions for which $\text{rank } g|_f > \text{nrank}(f)$. We have not looked for such examples.

Using [Theorem 9.1.5](#), [Eq. \(9.1\)](#) and [Theorem 8.2.5](#), we obtain the following corollary.

Corollary 9.1.6. *For any f -routing protocol, where $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, that is $(\varepsilon_0, \varepsilon_1) = (0, 1/20)$ -correct, we have*

$$\text{FR}_0(f) \geq \frac{1}{4}(\text{QNP}^{\text{cc}}(f) - 1) ,$$

and, for any such protocol that is $(\varepsilon_0, \varepsilon_1) = (1/20, 0)$ -correct, we have

$$\text{FR}_1(f) \geq \frac{1}{4}(\text{coQNP}^{\text{cc}}(f) - 1) .$$

Furthermore, for f -routing which has two sided perfection, we have

$$\text{pFR}(f) \geq \frac{1}{4} \max\{\text{QNP}^{\text{cc}}(f), \text{coQNP}^{\text{cc}}(f)\} - 1 ,$$

Where $\text{pFR}(f)$ is the log of Schmidt rank of minimal resource state that allows us to complete the f -routing task with perfect completeness.

Abusing the notation, by labelling the set of function families which can be f -routed with polylogarithmic entanglement by pFR , we have

$$\text{pFR} \subseteq \text{QNP}^{\text{cc}} \cap \text{coQNP}^{\text{cc}} .$$

In the classical setting, $\text{NP}^{\text{cc}} \cap \text{coNP}^{\text{cc}} = \text{P}^{\text{cc}}$ [BFS86], but to our knowledge, it is not known if the quantum analogue holds.

9.2 Lower bounds for f -BB84

Similarly to before, let $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$ be an arbitrary Boolean function, and recall the definition of f -BB84 from Definition 8.3.1. Again, we will focus on f -BB84 with $\varepsilon_0 = 0, \varepsilon_1 \geq 0$ so that the protocol is perfectly correct on zero instances of $f(x, y)$, or with $\varepsilon_0 > 0$ and $\varepsilon_1 = 0$ so that the protocol is perfectly correct on 1 instances. We define the entanglement cost of an f -BB84 protocol to be the log of the minimal Schmidt rank of any resource system that can be used to perform the f -BB84 task. In notation, we define $\text{FBB84}_0(f)$ to be entanglement cost for f -routing with $\varepsilon_0 = 0, \varepsilon_1 = 0.05$, $\text{FBB84}_1(f)$ to be the entanglement cost when $\varepsilon_0 = 0.05, \varepsilon_1 = 0$.

As in the analysis of f -routing, we look for a function of the density matrix prepared by the protocol which captures the structure of the function $f(x, y)$. We define this next.

Definition 9.2.1. *Given a f -BB84 protocol with mid-protocol density matrix $\rho_{RMM'}$, define the structure function by first defining the density matrices*

$$\begin{aligned}\rho_M^0 &= \langle 0 |_R \rho_{RM} | 0 \rangle_R , \\ \rho_M^1 &= \langle 1 |_R \rho_{RM} | 1 \rangle_R .\end{aligned}$$

These are the post measurement states on M when the referee measures R in the computational basis, and finds outcome $b \in \{0, 1\}$. We similarly define $\rho_{M'}^0$ and $\rho_{M'}^1$. Then define

$$g(x, y) = \text{tr}(\rho_M^0 \rho_M^1) + \text{tr}(\rho_{M'}^0 \rho_{M'}^1) .$$

Our goal is to show that $g(x, y)$ and $f(x, y)$ are zero on exactly the same inputs. To do this, we need some results from earlier analysis of f -BB84. In particular, we recall the following definition from [BCS22], albeit with a slightly different notation.

Definition 9.2.2. *For $\varepsilon \in [0, 1]$, let S_0^ε be the set of states $\rho_{RMM'}$ such that there exists a measurement on subsystem M and a measurement on subsystem M' that each allow us to guess the outcome of a measurement in the computational basis on R with probability at least $1 - \varepsilon$. Similarly, let S_1^ε be the set of states $\rho_{RMM'}$ such that there exists a measurement on subsystem M and a measurement on subsystem M' that allows us to guess the outcome of a measurement in the Hadamard basis on R with probability at least $1 - \varepsilon$.*

With this definition we can state the following result from [BCS22, ACCM25].

Lemma 9.2.3 (Informal, see Lemma 10.1.3). *Suppose that $\varepsilon < 0.11$. Then $S_0^\varepsilon \cap S_1^\varepsilon = \emptyset$.*

We will skip the proof of this lemma for now, and give a proof later in Chapter 10. Informally, this result states that a state that works well in 1 instances cannot also be a state that works well in 0 instances, and vice versa. We use this idea to prove the next lemma.

Lemma 9.2.4. *In a f -BB84 protocol with $\varepsilon_0 = 0$ and $0 \leq \varepsilon_1 < 0.11$, the structure function $g(x, y)$ is zero if and only if $f(x, y)$ is zero.*

Proof. We first establish the following fact: $g(x, y) = 0$ if and only if both the following are true: ρ_M^0 and ρ_M^1 have non-overlapping support, and $\rho_{M'}^0$ and $\rho_{M'}^1$ have non-overlapping

support. To see this, first assume $g(x, y) = 0$. Then diagonalize ρ_M^0 and ρ_M^1 so that

$$\begin{aligned}\rho_M^0 &= \sum_i \lambda_i |\psi_i\rangle\langle\psi_i| \ , \\ \rho_M^1 &= \sum_j \mu_j |\phi_j\rangle\langle\phi_j| \ , \\ \rho_{M'}^0 &= \sum_i \lambda'_i |\psi'_i\rangle\langle\psi'_i| \ , \\ \rho_{M'}^1 &= \sum_j \mu'_j |\phi'_j\rangle\langle\phi'_j| \ ,\end{aligned}$$

with $\lambda_i, \mu_j, \lambda'_i, \mu'_j > 0$. Then $0 = g(x, y)$ gives that

$$0 = \text{tr}(\rho_M^0 \rho_M^1) + \text{tr}(\rho_{M'}^0 \rho_{M'}^1) = \sum_{i,j} \lambda_i \mu_j |\langle\phi_j|\psi_i\rangle|^2 + \sum_{i,j} \lambda'_i \mu'_j |\langle\phi'_j|\psi'_i\rangle|^2 \ .$$

But this occurs if and only if

$$\forall i, j, \quad \langle\phi_j|\psi_i\rangle = 0 \ ,$$

and

$$\forall i, j, \quad \langle\phi'_j|\psi'_i\rangle = 0 \ ,$$

so that ρ_M^0 and ρ_M^1 have non-overlapping support, and $\rho_{M'}^0$ and $\rho_{M'}^1$ do, as needed. For the converse, a direct calculation shows both $\text{tr}(\rho_M^0 \rho_M^1)$ and $\text{tr}(\rho_{M'}^0 \rho_{M'}^1)$ are 0.

Now, suppose that $f(x, y) = 0$. By assumption the protocol has $\varepsilon_0 = 0$. Considering Alice, this gives that Alice can obtain b from system M with probability 1. But this says that ρ_M^0 and ρ_M^1 must have orthogonal support. Considering Bob, perfect correctness also gives that Bob can determine b from M' with probability 1, so that $\rho_{M'}^0$ and $\rho_{M'}^1$ have orthogonal support. Then, from the comment above, we have that $g(x, y) = 0$.

Next suppose that $f(x, y) = 1$. By assumption, this gives that $\rho_{RMM'} \in S_1^\varepsilon$, so by [Lemma 10.1.3](#) that $\rho_{RMM'} \notin S_0^\varepsilon$. This means that either ρ_M^0 and ρ_M^1 have overlapping support, $\rho_{M'}^0$ and $\rho_{M'}^1$ have overlapping support, or they both do. This is because otherwise if they both have non-overlapping support there would be a measurement to perfectly determine b from both M and M' , and the state would be in S_0^ε . But then by our initial comment that $g(x, y) = 0$ if and only if these density matrices have orthogonal support, we must have $g(x, y) > 0$. $\square$

Lemma 9.2.5. *An f -BB84 protocol that uses a resource system with Schmidt rank d_E has a structure function of the form*

$$g(x, y) = \sum_{I=1}^{2d_E^4} f_I(x) f'_I(y) .$$

Proof. From the general form of a non-local quantum computation protocol, the density matrix $\rho_{RM'_0 M'_1}(x, y)$ can be expressed as

$$\rho_{RM'_0 M'_1} = \mathcal{N}_{QA \rightarrow M'_0}^x \otimes \mathcal{M}_{B \rightarrow M'_1}^y (\Psi_{RQ} \otimes |\psi\rangle\langle\psi|_{AB}) .$$

We will write $|\psi\rangle_{AB}$ in the Schmidt basis,

$$|\psi\rangle_{AB} = \sum_{i=1}^{d_E} |i\rangle_A |i\rangle_B ,$$

with un-normalized vectors $|i\rangle_A, |i\rangle_B$. Then we get

$$\begin{aligned} \rho_{RM'_0 M'_1} &= \sum_{i,j=1}^{d_E} \mathcal{N}_{QA \rightarrow M'_0}^x (\Psi_{RQ}^+ \otimes |i\rangle\langle j|_A) \otimes \mathcal{M}_{B \rightarrow M'_1}^y (|i\rangle\langle j|_B) \\ &= \sum_{i,j=1}^{d_E} A_{RM'_0}^{x,i,j} \otimes B_{M'_1}^{y,i,j} . \end{aligned}$$

We can also project R into $|b\rangle_R$, $b \in \{0, 1\}$ to get $\rho_{M'}^i$,

$$\rho_{M'_0 M'_1}^b = \sum_{i,j=1}^{d_E} A_{M'_0}^{b,x,i,j} \otimes B_{M'_1}^{y,i,j} .$$

A similar calculation gives ρ_M^b . Inserting these expressions into $g(x, y)$, we obtain an expression of the form

$$g(x, y) = \sum_{I=1}^{2d_E^4} f_I(x) f'_I(y) .$$

where the d_E^4 appears because each of the two trace terms in $g(x, y)$ comes with a product

of two density matrices, each of which involves two sums running from 1 to d_E . We collect all of these sums into the index i appearing here, which now runs 1 to $2d_E^4$. $\square$

Similarly to the case of f -routing, we can conclude our main lower bound from the above lemma.

Theorem 9.2.6. *For any f -routing protocol, where $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, that is $(\varepsilon_0, \varepsilon_1) = (0, 1/20)$ -correct, we have*

$$\text{FBB84}_0(f) \geq \frac{1}{4}(\log \text{rank } g|_f - 1) ,$$

where the notation $g|_f$ indicates as before that g is zero and non-zero on the same inputs as $f(x, y)$. We can also obtain a lower bound on $\text{FBB84}_1(f)$ by considering the negated function,

$$\text{FBB84}_1(f) \geq \frac{1}{4}(\log \text{rank } g|_{\neg f} - 1) .$$

Also, similarly to the case of f -routing, we can use the previous theorem, [Eq. \(9.1\)](#) and [Theorem 8.2.5](#) to obtain the following corollary.

Corollary 9.2.7. *For any f -BB84 protocol, where $f : \{0, 1\}^{2n} \rightarrow \{0, 1\}$, that is $(\varepsilon_0, \varepsilon_1) = (0, 1/20)$ -correct, we have*

$$\text{FBB84}_0(f) \geq \frac{1}{4}(\text{QNP}^{\text{cc}}(f) - 2) ,$$

and, for any such protocol that is $(\varepsilon_0, \varepsilon_1) = (1/20, 0)$ -correct, we have

$$\text{FBB84}_1(f) \geq \frac{1}{4}(\text{coQNP}^{\text{cc}}(f) - 2) .$$

9.3 A new lower bound technique for CDS

It was recently understood [[ABM⁺23](#)] that f -routing is closely related to a classical information theoretic primitive known as conditional disclosure of secrets (CDS). As a consequence, our new lower bounds on f -routing imply lower bounds on CDS. In this work,

we will informally describe these lower bounds, but see [AKL⁺24] for a more complete treatment.

The *conditional disclosure of secrets* primitive involves three parties, Alice, Bob, and the referee. Alice receives input $x \in \{0, 1\}^n$, and Bob receives $y \in \{0, 1\}^n$. Alice additionally holds a secret string s , and Alice and Bob share a random string r . Alice and Bob do not communicate, but each sends a message to the referee. The goal is for the referee to be able to learn s if and only if a function $f(x, y) = 1$, where the choice of the function f is known to all parties, and the referee knows both inputs x and y .

The quantum generalization of CDS, *conditional disclosure of quantum secrets* or CDQS, was recently defined [ABM⁺23]. In the quantum setting, Alice and Bob share a quantum state (rather than a random string) and can send quantum messages to the referee. The secret can now also be taken to be a quantum system Q rather than a classical string. The security and privacy conditions are similar to the classical case: correctness means that the referee can invert the map on Q applied by Alice and Bob when $f(x, y) = 1$, and security means the map applied by Alice and Bob is close to completely depolarizing when $f(x, y) = 0$. In general, we allow small correctness and privacy errors. See [ABM⁺23] for a formal definition.

We define the minimal number of qubits in Alice and Bob's shared resource system in a CDQS scheme for function f by $\text{CDQS}(f)$. If we require the protocol to be perfectly correct, we write $\text{pcCDQS}(f)$, and when perfectly private, we write $\text{ppCDQS}(f)$. Similarly, for classical CDS, we use $\text{CDS}(f)$ to denote the randomness cost of classical CDS. Again we modify this to $\text{pcCDS}(f)$, $\text{ppCDS}(f)$ for perfectly correct and perfectly private CDS randomness cost.¹

In [ABM⁺23], it was shown that a good classical CDS protocol for a function f implies a good quantum CDS protocol for the same function. This leads to²

$$\text{CDS}(f) = \Omega(\text{CDQS}(f)) ,$$

and the same relationships hold for the perfectly correct and perfectly private variants. Also in [ABM⁺23], it was shown that a good CDQS leads to a good f -routing protocol, and vice versa. This leads to

$$\text{CDQS}(f) = \Theta(\text{FR}(f)) ,$$

¹The reader should be warned that the same notation is used to denote the *communication* cost in other papers on CDS.

²In fact, to obtain this we also need an amplification result which appears in [AKL⁺24]. These bounds are proved formally there.

Also, perfectly private and perfectly correct variants are related to f -routing with one-sided errors,

$$\begin{aligned}\text{ppCDQS}(f) &= \Theta(\text{FR}_0(f)) \ , \\ \text{pcCDQS}(f) &= \Theta(\text{FR}_1(f)) \ .\end{aligned}$$

This implies that our lower bounds from [Theorem 9.1.5](#) apply to CDS. We have the following corollary.

Corollary 9.3.1. *Let $f : \{0,1\}^{2n} \rightarrow \{0,1\}$ and g be as in [Theorem 9.2.6](#). Then for perfectly private CDQS and perfectly correct CDQS we have*

$$\begin{aligned}\text{ppCDQS}(f) &= \Omega(\log \text{rank } g|_f) \ , \\ \text{pcCDQS}(f) &= \Omega(\log \text{rank } g|_{\neg f}) \ .\end{aligned}$$

This also gives lower bounds in terms of $\text{QNP}^{\text{cc}}(f)$ and $\text{coQNP}^{\text{cc}}(f)$, and the upper bound can be written in terms of classical CDS. This way, we can obtain the lower bound

$$\text{pcCDS}(f) \geq \Omega(\text{coQNP}^{\text{cc}}(f)) \ .$$

This is similar and comparable to the following lower bound obtained in [\[AV21\]](#),

$$\text{pcCDS}(f) \geq \frac{1}{2} \text{coNP}^{\text{cc}}(f) \ .$$

The lower bound on perfectly private CDS randomness cost seems to be new,

$$\text{ppCDS}(f) \geq \Omega(\text{QNP}^{\text{cc}}(f)) \ .$$

For perfect CDS, we can bound the randomness cost by the maximum of the QNP^{cc} and coNP^{cc} cost. If by abusing notation, we let pCDS denote the set of families of functions for which CDS can be implemented with perfect correctness and perfect privacy using polylogarithmic randomness, we obtain

$$\text{pCDS} \subseteq \text{QNP}^{\text{cc}} \cap \text{coNP}^{\text{cc}} \ .$$

9.4 Application to concrete functions

We detail lower bounds for equality, non-equality, greater than and less than, set-disjointness, and set-intersection below. Our results are summarized succinctly in [Table 9.1](#).

Function	pFR	FR ₁	FR ₀	FR
Equality	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$
Non-Equality	$\Theta(n)$	$\Theta(n)$	$\Theta(1)$	$\Theta(1)$
Inner-Product	$O(n), \Omega(1)$	$O(n), \Omega(1)$	$O(n), \Omega(1)$	$O(n), \Omega(1)$
Greater-Than	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$O(n), \Omega(1)$
Set-Intersection	$\Theta(n)$	$\Theta(n)$	$O(n), \Omega(1)$	$O(n), \Omega(1)$
Set-Disjointness	$\Theta(n)$	$O(n), \Omega(1)$	$\Theta(n)$	$O(n), \Omega(1)$

Table 9.1: Known upper and lower bounds on entanglement cost (quantified by the log of Schmidt rank) in f -routing for natural functions.

Equality and non-equality

Choose $f(x, y)$ to be the equality function,

$$EQ(x, y) = \begin{cases} 0, & x \neq y \\ 1, & x = y \end{cases}.$$

Then $g(x, y)$ is zero except on the diagonal, forcing it to have full rank. Therefore, from [Theorems 9.1.5](#) and [9.2.6](#) we have

$$\text{FR}_0(EQ) \geq \frac{n}{4}, \quad \text{pFBB84}(EQ) \geq \frac{1}{4}(n-1).$$

For the not-equals function, we can use the bound from [Theorem 9.1.5](#) to obtain

$$\text{FR}_1(NEQ) \geq \frac{n}{4}.$$

Greater and less than

Similarly, the ‘greater than’ function,

$$GT(x, y) = \begin{cases} 0, & x < y \\ 1, & x \geq y \end{cases} .$$

is upper triangular with non-zero elements on the diagonal, so it also has full rank, and we obtain a linear lower bound.

$$\text{FR}_0(GT) \geq \frac{n}{4}, \quad \text{pFBB84}(GT) \geq \frac{1}{4}(n-1) .$$

Further, because the negation of Greater-Than is also full rank, we can also bound fR_1 ,

$$\text{FR}_1(GT) \geq \frac{n}{4} .$$

The same bounds hold for the ‘less than’ function.

Set-Intersection and set-disjointness

Set disjointness is upper left triangular.³ This implies it is full rank, therefore

$$\text{FR}_0(DISJ) \geq \frac{n}{4}, \quad \text{pFBB84}(DISJ) \geq \frac{1}{4}(n-1) .$$

Since the negation of set intersection is set disjointness and hence of full rank, we also obtain

$$\text{FR}_1(INT) \geq \frac{n}{4} .$$

³To see why, consider that on the diagonal of the truth table running top right to bottom left, we have that $x + y = 11\dots 11$, the all 1’s string. This means x and y must have non-zero values in non-overlapping locations, e.g. for two bits this diagonal consists of $(x = 00, y = 11)$, $(x = 01, y = 10)$, $(x = 10, y = 01)$ and $(x = 11, y = 00)$. Moving downward from any entry on that diagonal y becomes larger, so must now have an overlapping entry with the x string.

Chapter 10

Gate Lower Bounds

In this chapter, we lower bound the number of gates and single qubit measurement that parties need to apply to successfully implement f -routing or f -BB84 for an arbitrary Boolean function f . Our lower bounds are stated in terms of a measure of the communication complexity of f , namely that of the simultaneous message passing with shared entanglement. Using this, we obtain explicit lower bounds for inner product and disjointness functions.

10.1 Analysis of f -BB84

We show that for a very general class of f -BB84 protocols, the state after the first round of operations can only successfully complete the task for either $f(x, y) = 0$ or $f(x, y) = 1$, but not both. Thus, the state after the first round determines the basis of the measurement b performed by Alice and Bob.

We begin by defining the sets of states for which the protocol can succeed in 0 instances and a second set of states for which it can succeed in 1 instances. A similar definition was given in the previous chapter. Here, we give a more formal one.

Definition 10.1.1. For $\varepsilon \in [0, 1]$, let S_0^ε be the set of states $|\phi\rangle_{RMM'}$ such that there exist a measurement on subsystem M and a measurement on subsystem M' that each allow us to guess the outcome of a measurement in the computational basis on R with probability at least $1 - \varepsilon$. Similarly, let S_1^ε be the set of states $|\phi\rangle_{RMM'}$ such that there exist a measurement on subsystem M and a measurement on subsystem M' that allows us to guess the outcome of a measurement in the Hadamard basis on R with probability at least $1 - \varepsilon$.

Next, we work towards proving that for small enough ε , the sets S_0^ε and S_1^ε are disjoint. We begin with the following lemma.

Lemma 10.1.2. *Let $h(x) = -x \log x - (1-x) \log(1-x)$ be the binary entropy function. Suppose $|\psi^0\rangle_{RMM'} \in S_0^\varepsilon$, and $\rho_{ZMM'}^0$ is obtained by measuring R in the computational basis. Then*

$$\begin{aligned} H(Z|M)_{\rho_{ZMM'}^0} &\leq h(\varepsilon) , \\ H(Z|M')_{\rho_{ZMM'}^0} &\leq h(\varepsilon) . \end{aligned}$$

Further, suppose $|\psi^1\rangle_{RMM'} \in S_1^\varepsilon$, and $\sigma_{ZMM'}^1$ is obtained by measuring R in the Hadamard basis. Then

$$\begin{aligned} H(Z|M)_{\sigma_{ZMM'}^1} &\leq h(\varepsilon) , \\ H(Z|M')_{\sigma_{ZMM'}^1} &\leq h(\varepsilon) . \end{aligned}$$

Proof. We will prove that

$$H(Z|M)_{\rho_{ZMM'}^0} \leq h(\varepsilon) .$$

The remaining statements are similar. Because $|\psi\rangle_{RMM'} \in S_0^\varepsilon$, there exists a measurement on M that determines the measurement outcome from measuring R with probability $1 - \varepsilon$. Let W denote this measurement outcome from measuring M . Then, Fano's inequality gives that

$$\Pr(Z \neq W) \leq \varepsilon \rightarrow H(Z|W)_{\rho^0} \leq h(\varepsilon) .$$

By data processing inequality, we further obtain that

$$H(Z|M)_{\rho^0} \leq H(Z|W)_{\rho^0} \leq h(\varepsilon) ,$$

as needed. □

Finally, we prove that the sets S_0^ε and S_1^ε are disjoint, as needed.¹

Lemma 10.1.3. *Suppose that $|\psi^0\rangle_{RMM'} \in S_0^\varepsilon$, and $|\psi^1\rangle_{RMM'} \in S_1^\varepsilon$ with $\varepsilon < 0.11$. Then we have $S_0^\varepsilon \cap S_1^\varepsilon = \emptyset$.*

¹Note that this lemma and [Lemma 10.2.3](#) are similar to the lemmas appearing in [\[BCS22\]](#), which in turn are similar to the statements in [\[BFSS13\]](#).

Proof. Because $|\psi^0\rangle_{RMM'} \in S_0^\varepsilon$, we get from [Lemma 10.1.2](#) that

$$H(Z|M)_{\rho^0} \leq h(\varepsilon) ,$$

where $\rho_{ZMM'}^0$ is obtained from $|\psi^0\rangle_{RMM'}$ by measuring R in the computational basis. By CIT, this means

$$H(Z|M')_{\sigma^0} \geq 1 - h(\varepsilon) .$$

where $\sigma_{ZMM'}^0$ is obtained by measuring $|\psi^0\rangle_{RMM'}$ in the Hadamard basis. Now consider $\psi_{RMM'}^1 \in S_1^\varepsilon$. By [Lemma 10.1.2](#) this has

$$H(Z|M')_{\sigma_{ZMM'}^1} \leq h(\varepsilon) ,$$

with σ^1 obtained from $\psi_{RMM'}^1$ by measuring R in the Hadamard basis. But then

$$H(Z|M')_{\sigma^0} - H(Z|M')_{\sigma^1} \geq 1 - 2h(\varepsilon) .$$

Now we apply the continuity of the conditional relative entropy ([Theorem 8.1.2](#)) to upper bound this entropy difference in terms of the trace distance between the states $\sigma_{ZMM'}^0, \sigma_{ZMM'}^1$. Defining $\Delta = \frac{1}{2} \|\sigma_{ZMM'}^0 - \sigma_{ZMM'}^1\|_1$ and noting that $\|\sigma_{ZMM'}^0 - \sigma_{ZMM'}^1\|_1 \geq \|\sigma_{ZM'}^0 - \sigma_{ZM'}^1\|_1$,

$$1 - 2h(\varepsilon) \leq H(Z|M')_{\sigma^0} - H(Z|M')_{\sigma^1} \leq 2\Delta + (1 + \Delta)h\left(\frac{\Delta}{1 + \Delta}\right) .$$

When $\varepsilon < 0.11$, $1 - 2h(\varepsilon) > 0$ and this places a non-trivial lower bound on Δ . But then

$$2\Delta = \|\sigma_{ZMM'}^0 - \sigma_{ZMM'}^1\|_1 \leq \|\psi_{RMM'}^0 - \psi_{RMM'}^1\|_1$$

by monotonicity of the trace distance, so that states in S_0^ε and S_1^ε are separated by a non-zero distance, which proves the lemma. $\square$

The fact that $S_0^\varepsilon \cap S_1^\varepsilon = \emptyset$ will be used in our reduction from f -BB84 to SMP^* . We give that reduction in [Section 10.3](#). Next, we prove a similar set separation for f -routing. This will allow us to use the same reduction for f -routing as well.

10.2 Analysis of f -routing

In this section, we review and generalize the results from [BFSS13, BCS22] showing that in an f -routing protocol, the side on which a qubit maximally entangled with the reference can be returned to the referee is already determined after Alice and Bob apply their operations $\mathcal{N}^x, \mathcal{M}^y$. In other words, the state $\rho_{RMM'}$ determines where the qubit will be routed. The analysis will be fairly similar to what we did in the previous section, but here we need to adopt the details to f -routing.

To see this, we begin by defining sets of states for which the qubit can be produced on the left or right, respectively.

Definition 10.2.1. *We define the 0-set $\tilde{S}_0^\varepsilon$ and 1-set $\tilde{S}_1^\varepsilon$ as*

$$\begin{aligned}\tilde{S}_0^\varepsilon &= \{\rho_{RMM'} : \exists \mathcal{N}_{M \rightarrow Q} \text{ s.t. } P(\mathcal{N}_{M \rightarrow Q} \circ \text{tr}_{M'}(\rho_{RMM'}), \Psi_{RQ}^+) \leq \varepsilon\} , \\ \tilde{S}_1^\varepsilon &= \{\rho_{RMM'} : \exists \mathcal{N}_{M' \rightarrow Q} \text{ s.t. } P(\mathcal{N}_{M' \rightarrow Q} \circ \text{tr}_M(\rho_{RMM'}), \Psi_{RQ}^+) \leq \varepsilon\} .\end{aligned}$$

We would like to show that the sets $\tilde{S}_0^\varepsilon$ and $\tilde{S}_1^\varepsilon$ do not overlap when ε is sufficiently small. Intuitively, the non-overlap of these sets indicates that the entanglement with R has been brought to either Alice or Bob after the first round of operations – if there is a way to recover the entanglement on the left, then there is not one on the right, and vice versa. We first record the following lemma.

Lemma 10.2.2. *Let $\rho_{RMM'}^0 \in \tilde{S}_0^{\varepsilon=0}$, $\rho_{RMM'}^1 \in \tilde{S}_1^{\varepsilon=0}$. Then*

$$P(\rho^0, \rho^1) \geq \frac{\sqrt{3}}{2} .$$

Proof. Consider purifications of $\tilde{\rho}^0, \tilde{\rho}^1$. We call these states $|\psi^0\rangle_{RMM'X}$ and $|\psi^1\rangle_{RMM'X}$. By purifying the channels appearing in the definitions of these sets, we have that there exist isometries $V_{M \rightarrow QE}$ and $W_{M' \rightarrow QE}$ such that

$$\begin{aligned}\text{tr}_{M'EX} \left(V_{M \rightarrow QE} |\psi^0\rangle\langle\psi^0|_{RMM'X} V_{M \rightarrow QE}^\dagger \right) &= \Psi_{RQ}^+ , \\ \text{tr}_{M'EX} \left(W_{M' \rightarrow QE} |\psi^1\rangle\langle\psi^1|_{RMM'X} W_{M' \rightarrow QE}^\dagger \right) &= \Psi_{RQ}^+ .\end{aligned}$$

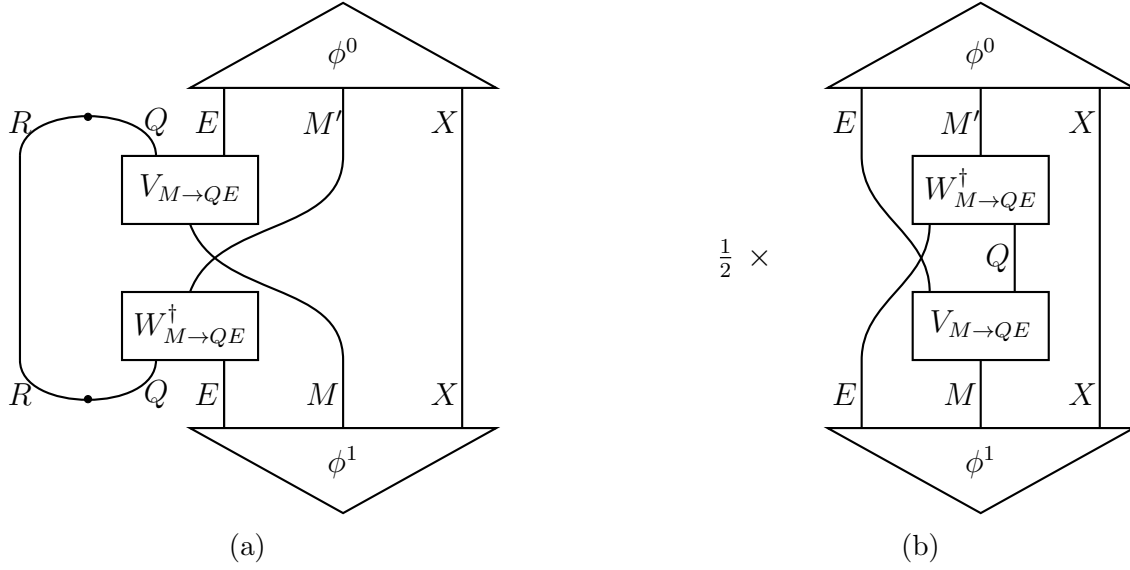


Figure 10.1: The inner product of ϕ^0 and ϕ^1 .

This implies the existence of pure states $|\phi^0\rangle_{M'EX}$, $|\phi^1\rangle_{MEX}$ such that

$$\begin{aligned} |\psi^0\rangle_{RMM'X} &= V_{M \rightarrow QE}^\dagger |\Psi^+\rangle_{RQ} \otimes |\phi^0\rangle_{M'EX} \quad , \\ |\psi^1\rangle_{RMM'X} &= W_{M' \rightarrow QE}^\dagger |\Psi^+\rangle_{RQ} \otimes |\phi^1\rangle_{MEX} \quad . \end{aligned}$$

All other purifications must be related by isometries $T_{X \rightarrow X'}^0$, $T_{X \rightarrow X'}^1$. In Fig. 10.1, we give a simple tensor calculation showing that the inner product of all such purifications is always smaller than $1/2$. In this figure, the curved lines are maximally entangled qubit pairs. See, e.g. [Mel23] for more details on this tensor notation. A rearrangement of the same inner product is given in Part (b) of the same figure. The maximally entangled pairs have been straightened to a wire, and the normalization of $1/2$ appears as an overall factor. The remaining object is again an inner product of two normalized states, now in a smaller Hilbert space, and is upper bounded by 1. Given this, we have

$$F(\tilde{\rho}^0, \tilde{\rho}^1) = \max_{|\tilde{\psi}^0\rangle, |\tilde{\psi}^1\rangle} \left| \langle \tilde{\psi}^0 | \tilde{\psi}^1 \rangle \right| \leq 1/2 \quad .$$

This implies $P(\tilde{\rho}^0, \tilde{\rho}^1) \geq \sqrt{3}/2$ as needed. □

We can now prove the following.

Lemma 10.2.3. *If $\varepsilon < \frac{\sqrt{3}}{4} \approx 0.43$, then $\tilde{S}_0^\varepsilon \cap \tilde{S}_1^\varepsilon = \emptyset$.*

Proof. If $|\psi_i\rangle$ is in S_i^ε it must be ε close in purified distance to a state in $S_i^{\varepsilon=0}$. Using this and Lemma 10.2.2 we find that if $|\psi_0\rangle \in \tilde{S}_0^\varepsilon$ and $|\psi_1\rangle \in \tilde{S}_1^\varepsilon$, then

$$P(|\psi_0\rangle, |\psi_1\rangle) \geq \frac{\sqrt{3}}{2} - 2\varepsilon .$$

Assuming $\varepsilon < \frac{\sqrt{3}}{4}$ we find that the purified distance is strictly positive and therefore the sets do not overlap. $\square$

10.3 Reduction to SMP^* and lower bounds

10.3.1 Reduction to SMP^*

In this section, we consider lower bounds on the number of quantum gates Alice and Bob need to apply in order to successfully complete a f -BB84 or f -routing task. We show that for certain functions, such as the inner product function, this is linear in the number of classical input bits n .

In more detail, we consider decomposing Alice and Bob's operations $\mathcal{N}^x$ and $\mathcal{M}^y$ into two qubit gates drawn from $\{T, X, Z, CNOT\}$ and single qubit measurements in the computational basis. Since we want to bound Alice and Bob's quantum operations, we will allow them free classical processing. This classical processing could take as inputs x, y and the outcomes of any mid-circuit measurements performed by Alice and Bob. In particular, the choice of gates later in the circuit can be conditioned on the outputs of classical processing involving earlier measurement outcomes. Notice that if we naively purify such a protocol, the classical processing that takes mid-circuit measurement outcomes as inputs will become a quantum operation. Thus, bounding quantum operations in the purified view does not suffice to bound the quantum operations in the un-purified view, and hence does not bound the operations Alice and Bob are required to implement physically. Instead, we must directly bound the quantum operations in the un-purified view.

To do this, we first prove a reduction from f -BB84 or f -routing to SMP^* .

Theorem 10.3.1. *Suppose P is an f -BB84 protocol that is $\varepsilon < \varepsilon_0 = 0.11$ correct, or an f -routing protocol that is $\varepsilon < \tilde{\varepsilon}_0 = \sqrt{3}/4$ correct, on a $(1 - \delta)$ -fraction of the inputs, uses*

$C_G(f)$ gates drawn from a gate set of size 4 and also uses $C_M(f)$ single qubit measurements in the computational basis. Then,

$$(\log(q) + 1)(2C_G(f) + C_M(f)) \geq SMP_{\delta, \varepsilon'}^*(f) , \quad (10.1)$$

where q is the number of qubits held by Alice and Bob, and $SMP_{\delta, \varepsilon'}^*(f)$ denotes the minimal message size needed to compute $f(x, y)$ in the SMP^* model with correctness $\varepsilon' = \varepsilon/\varepsilon_0$ for f -BB84 and $\varepsilon' = \varepsilon/\tilde{\varepsilon}_0$ for f -routing on at least $1 - \delta$ fraction of possible inputs.

Proof. We consider an f -routing protocol and show that it defines an SMP^* protocol. The referee holds a classical description of the initial resource state. Alice and Bob share the resource system. Alice and Bob's strategy will be to send the referee a description of their local operations. We consider a decomposition of Alice and Bob's operations into gates and measurements. Alice and Bob apply their operations to their shared resource state and the input system. As they do so, they keep a record of the gates they apply (which may be computed using mid-circuit measurement outcomes) and their measurement outcomes m , then send this to the referee. The referee will then compute a classical description of the state $\rho_{RMM'}(m)$ and determine if it is inside of $\tilde{S}_0^\varepsilon$ or $\tilde{S}_1^\varepsilon$. We show in the following that, as a consequence of the correctness of the f -routing protocol, with high probability $\rho_{RMM'}(m)$ is inside the set $\tilde{S}_{f(x,y)}$. For each gate, they specify the gate choice, requiring 2 bits, and the location of the gate, which requires $2 \log q$ bits for a contribution of $(2 \log q + 2)C_G(f)$ bits. Furthermore, specifying each measurement requires $\log q$ bits to show where the measurement occurs plus 1 bit to specify the measurement outcome, for a contribution of $(\log q + 1)C_M(f)$. The total message size sent by Alice and Bob then is the left hand side of Eq. (10.1).

It remains to show that $\rho_{RMM'}(m)$ is inside of $\tilde{S}_{f(x,y)}$ with high probability over the measurement outcomes m . We first establish this for a pair of inputs $(x, y) \in f^{-1}(0)$ which is ε -correct, and a $(x, y) \in f^{-1}(1)$ is similar. By correctness of the f -routing protocol, we have that there exists a decoder $\mathcal{D}_{MX_M \rightarrow Q}^{x,y}$ such that

$$P \left(\mathcal{D}_{MX_M \rightarrow Q}^{x,y} \left(\sum_m p_m \rho_{RM}(m) \otimes |m\rangle\langle m|_{X_M} \right), \Psi_{RQ}^+ \right) \leq \varepsilon ,$$

so that the decoders $\mathcal{D}_{M \rightarrow Q}^{m,x,y}(\cdot) = \mathcal{D}_{MX_M \rightarrow Q}(\cdot_M \otimes |m\rangle\langle m|_{X_M})$ have

$$\sum_m p_m P(\mathcal{D}_{MX_M \rightarrow Q}^{x,y}(\rho_{RM}(m)), \Psi_{RQ}^+) \leq \varepsilon .$$

Define the random variable $P_m = P(\mathcal{D}_{MX_M \rightarrow Q}^{x,y}(\rho_{RM}(m)), \Psi_{RQ}^+)$, so that the above reads $\langle P_m \rangle \leq \varepsilon$. So long as $P_m \leq \tilde{\varepsilon}_0$ we will have that $\rho_{RMM'}(m) \in \tilde{S}_0^{\tilde{\varepsilon}_0}$, so the referee fails only when $P_m > \tilde{\varepsilon}_0$. By Markov's inequality, this occurs with probability

$$\Pr[P_m > \tilde{\varepsilon}_0] \leq \frac{\varepsilon}{\tilde{\varepsilon}_0}.$$

Thus, the referee succeeds with probability $p \geq 1 - \varepsilon/\tilde{\varepsilon}_0$, so the SMP^* protocol is $\varepsilon' = \varepsilon/\tilde{\varepsilon}_0$ correct, as needed. A similar argument establishes ε' -correctness of the SMP^* protocol on inputs $(x, y) \in f^{-1}(1)$ that are ε -correct in the f -routing protocol. Because this argument shows f -routing correctness on a given input implies SMP^* correctness on the same input, if the f -routing is ε -correct on a fraction $1 - \delta$ of inputs the SMP^* protocol is ε' correct on that fraction of inputs as well.

The proof for f -BB84 is the same as the above, but now we replace [Lemma 10.2.3](#) with [Lemma 10.1.3](#). In this setting, the referee now looks at the state $\rho_{RMM'}$ and determines if this is in $S_0^{\varepsilon_0}$ or $S_1^{\varepsilon_0}$. Otherwise, the proof is the same. $\square$

It is worth commenting on why the reduction from f -routing is to SMP^* rather than just SMP . To understand this, notice that Alice and Bob cannot necessarily compute their gate choices directly from their inputs x and y . Instead, they may use the outcomes of mid-circuit measurements to choose gates. To determine these measurement outcomes, Alice and Bob need to share the same entangled state in their SMP protocol as is held in the f -routing protocol. A natural thought to avoid this is to have Alice and Bob purify their protocols, and apply only unitaries. In this case, however, classical processing used in the original protocol leads to additional quantum gates in the purified protocol. Thus, this would lower bound not the quantum gate complexity, but instead the total complexity including any classical part, and hence give a weaker bound.

10.3.2 Explicit lower bounds for f -BB84 and f -routing

In this section, we recall explicit lower bounds on SMP^* complexity of certain functions such as inner product (IP) and disjointness ($DISJ$) functions.

We first start by recalling the following result from [\[CVDNT98\]](#) on C_ε^* complexity of IP .

Lemma 10.3.2 ([\[CVDNT98\]](#)). *Let $IP : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be the mod 2 inner*

product function, and let $C_\varepsilon^*(IP)$ be the two-way communication complexity. Then,

$$C_\varepsilon^*(IP) \geq \max\left\{\frac{1}{2}(1-2\varepsilon)^2, (1-2\varepsilon)^4\right\}n - 1/2 .$$

This result is later improved in [NS02], which we state below.

Lemma 10.3.3 ([NS02]). *Let $IP : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be the mod 2 inner product function, and let $C_\varepsilon^*(IP)$ be the two-way communication complexity. Then,*

$$C_\varepsilon^*(IP) \geq \max\left\{\frac{1}{2}n + 2\log(1-2\varepsilon), (1-2\varepsilon)^4n - 1/2\right\} .$$

As a consequence of Eq. (8.1), we have the following corollary.

Corollary 10.3.4. *For IP , we have that*

$$SMP_\delta^*(IP) \geq \max\left\{\frac{1}{2}n + 2\log(1-2\delta), (1-2\delta)^4n - 1/2\right\} .$$

As a direct consequence of Theorem 10.3.1 and Corollary 10.3.4, we can state the following linear lower bound on the number of gates required to perform f -routing and f -BB84 tasks for IP .

Theorem 10.3.5. *Suppose $IP : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be the mod 2 inner product function, and P is an f -BB84 protocol for IP that is $\varepsilon < 0.11$ correct, or an f -routing protocol for IP that is $\varepsilon < \sqrt{3}/4$ correct, on a $1 - \delta$ fraction of the inputs, uses $C_G(IP)$ gates drawn from a gate set of size 4 and also uses $C_M(IP)$ single qubit measurements in the computational basis. Then,*

$$(\log(q) + 1)(2C_G(IP) + C_M(IP)) \geq \frac{1}{2}n + 2\log(1 - 2\varepsilon') = \Omega(n) ,$$

where q is the number of qubits held by Alice and Bob, and $\varepsilon' = \delta + (1 - \delta) \cdot \varepsilon$.

The reduction in Theorem 10.3.1 can easily be applied to other known SMP^* lower bounds for explicit functions. As an another example, it is shown in [Raz03] that letting $DISJ : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be the disjointness function, then we have $C_\varepsilon^*(DISJ) \geq \Omega(\sqrt{n})$. Having this and Theorem 10.3.1, we can conclude the following corollary.

Corollary 10.3.6. *Suppose $DISJ : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}$ be the disjointness function, and P is an f -BB84 protocol for $DISJ$ that is $\varepsilon < 0.11$ correct, or an f -routing protocol*

for IP that is $\varepsilon < \sqrt{3}/4$ correct, on a $1 - \delta$ fraction of the inputs, uses $C_G(DISJ)$ gates drawn from a gate set of size 4 and also uses $C_M(DISJ)$ single qubit measurements in the computational basis. Then,

$$(\log(q) + 1)(2C_G(DISJ) + C_M(DISJ)) \geq \Omega(\sqrt{n}) ,$$

where q is the number of qubits held by Alice and Bob, and $\varepsilon' = \delta + (1 - \delta) \cdot \varepsilon$.

10.3.3 Nearly matching upper bound for inner product

For the inner product function, our lower bound on quantum operations is tight up to logarithmic factors. To see this, we use the garden-hose strategy [BFSS13] to give an upper bound. This strategy uses only Bell basis measurements and classical processing to attack any f -routing scheme. Adapted to the garden-hose setting, where we only have a single element in our gate set (the two qubit Bell basis measurement), our bound becomes

$$2 \log q C_M \geq n .$$

We construct a scheme using $q = 12n$ EPR pairs and $24n$ measurements in the garden-hose model. This scheme then saturates the above bound up to logarithmic factors.²

To construct the protocol, recall that in the garden-hose attack Alice and Bob share N EPR pairs, and make Bell basis measurements connecting either the input system Q to an EPR pair, or connecting two EPR pairs. A useful analogy is to a set of N pipes, the ends of which can be connected together in pairs by hoses, or connected to the tap, which plays the role of the input system. The water flowing through the pipes tracks where the system Q ends up. A garden-hose attack for the inner product is as follows. Consider splitting N into sets of 6 pipes. Alice will make measurements connecting her first set of 6 to her second set of 6. By wiring her connections appropriately, she can apply any permutation to the pipes. Define the permutations,

$$\begin{aligned} A &= (13)(24) , \\ B &= (35)(46) , \\ F &= (56) . \end{aligned}$$

²Note that we have not tried to optimize the constants appearing in our construction, which likely can be improved.

We then have Alice and Bob implement the permutations

$$S_i = A^{x_i} B^{y_i} F B^{y_i} A^{x_i} .$$

Furthermore, Alice connects the tap to the 1st hose from her first set and sends the 2nd hose from her final set to Bob. Notice that S_i swaps the water from the first to the second hose iff $x_i y_i = 1$, so that the water ends up on the first hose (and so the state ends up with Alice) if $\sum_i x_i y_i = 0 \bmod 2$, and with Bob otherwise, as needed. Because Alice and Bob each do two sets of 6 measurements for each value of i , this uses $24n$ measurements. In addition, they used 2 sets of hoses connecting for each i so $12n$ hoses, completing the upper bound.

Chapter 11

Conclusion

We conclude the second part of this thesis by summarizing our results, and naming a few interesting problems that our work leaves open.

11.1 Summary of results

In this work, we have given the first non-trivial lower bounds on entanglement in the f -routing and f -BB84 tasks. For some natural functions, including equality, greater than, and set-disjointness, our lower bounds are linear in the classical input size. This realizes a long standing aspiration for f -routing and f -BB84 based position-verification schemes: that an honest party can compute a simple, classical function and perform $O(1)$ quantum operations, while a dishonest party must use entanglement that grows with the size of the classical input strings. However, this result comes at the unfortunate cost of requiring (one-sided) perfect correctness for the lower bounds to apply. Nonetheless, even proving bounds in this perfect case has so far been elusive, and we hope that our progress may open future avenues toward robust linear lower bounds.

An interesting observation is that our bounds lead to relations among operational quantities, e.g.

$$\text{FR}_0(f) \geq \text{QNP}^{cc}(f)$$

but are derived by separately relating FR_0 and QNP^{cc} to the non-deterministic rank. Given this result, it would be natural to expect a reduction at an operational level from

f -routing to quantum non-deterministic communication complexity, but we know of no such reduction. Another comment related to the non-deterministic rank is that our lower bounds are in terms of the rank of matrices with positive entries wherever $f(x, y) = 1$, and 0 entries otherwise. Minimizing over all such matrices produces a number lower bounded by the non-deterministic rank (which allows arbitrary complex non-zero entries). It would be interesting to understand if the minimization over positive entries is ever larger than over complex entries.

As a consequence of our lower bounds, we also obtain a new lower bound technique for randomness complexity in CDS with either perfect privacy or perfect correctness. We highlight the bound

$$\text{ppCDS}(f) \geq \Omega(\text{QNP}^{\text{cc}}(f)) \ .$$

which appears to be new and incomparable to earlier bounds on ppCDS. It would be interesting to construct explicit functions for which this lower bound is tighter than earlier ones, for instance, bounds from the PP^{cc} cost of f [AA20].

We have also given a new linear lower bound on the gate complexity of performing f -routing and f -BB84 against the inner product function. Our bound differs from earlier work in that it applies to a natural, low complexity function (inner product) and bounds the number of quantum operations necessary for an attack rather than the amount of shared entanglement. Assuming quantum gates are more difficult to implement than the same number of classical gates, our bound provides a separation in difficulty between an honest and dishonest player. Furthermore, it does so for a scheme that is computationally feasible for an honest prover, and which is loss tolerant.

A key strategy in this work compared to earlier ones has been to focus on the computational resources of the dishonest player. We can compare our bounds on quantum operations with bounds on system size, which are more common in the literature. To do so, consider restricting the dishonest player to circuits of depth d . Then, their maximum number of gates is $C \sim dq$ for q the number of qubits they control. From our bound we obtain the following.

$$q \gtrsim \frac{n}{d} \ .$$

A natural restriction on the depth of the dishonest player's circuit is to take $d = O(\log(n))$, the same as (for the inner product function) the classical circuit depth of the honest player.¹

¹If the dishonest player has agents sitting at constant positions this is enforced by relativity.

With this restriction, we obtain an almost linear lower bound on q , $q \geq n/\log(n)$. This compares favorably to the $q \gtrsim \log(n)$ lower bound obtained by [BCS22], and furthermore avoids the issue of the purified view of [BCS22] counting what can in practice be classical systems towards the quantum system size. Thus, our bound, plus the assumption that the attacker’s circuit depth should be similar to the honest players, gives a stronger bound on quantum resource system size than has been proven previously. We can also note that applying deeper circuits to small systems is plausibly harder than shallow circuits on larger systems, so even relaxing this assumption and relying only on the gate lower bound seems a stronger bound than a linear bound on system size.

A note on a subsequent work: Recently, an equivalence between f -routing and f -BB84 was shown in [BHM⁺25]. In this work, the authors study reductions among f -routing, f -BB84, CDQS, and other problems in non-local quantum computations, and in particular, they show that f -BB84 and f -routing are both reducible to each other with small overhead. In terms of connections to our results, this allows our rank lower bounds on f -BB84 to be derived from the rank lower bounds on f -routing.

11.2 Open problems

Our work leaves many natural open problems and raises several new directions of investigation. However, for brevity, we would like to focus on and highlight three directions that we find particularly promising.

The first and most immediate open problem that comes to mind is about extending the entanglement lower bounds to the non-perfect protocols. It seems like doing this might require completely new ideas, as it is not clear whether we can say much about the rank of a matrix that is non-zero when $f(x, y) = 1$, but is *close* to zero when $f(x, y) = 0$. Therefore, it might be the case that this approach fails completely when one tries to make it robust against noise.

Open Problem 5. *Is it possible to obtain comparable entanglement lower bounds that are robust against noise, and hold true even when the protocols do not have one-sided perfect correctness?*

Another intriguing open problem is to understand the connection between entanglement lower bounds and non-deterministic communication complexity more carefully. It seems that although we do not show a direct reduction from one to another, the techniques of

showing lower bounds on both are quite similar. Therefore, it seems very plausible that such a reduction does in fact exist and might simplify the lower bounds techniques.

Open Problem 6. *Is it possible to obtain entanglement lower bounds by reducing it directly to non-deterministic quantum communication complexity (or another similar measure) to simplify the proof techniques of this work?*

The last open problem we would like to raise is about the connection to the randomness complexity of conditional disclosure of secrets. It would be quite interesting to see if one can obtain explicit lower bounds that are comparable to the ones known previously, by using lower bounds on f -routing and f -BB84.

Open Problem 7. *Is it possible to improve known lower bounds on CDS (and its variants) which is a classical notion, using ideas from quantum information and in particular through reductions from f -routing and f -BB84?*

References

- [AA20] Benny Applebaum and Barak Arkis. On the power of amortization in secret sharing: d-uniform secret sharing and cds with constant information rate. *ACM Transactions on Computation Theory (TOCT)*, 2020.
- [AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- [ABB⁺23] Rene Allerstorfer, Andreas Bluhm, Harry Buhrman, Matthias Christandl, Llorenç Escolà-Farràs, Florian Speelman, and Philip Verduyn Lunel. Making existing quantum position verification protocols secure against arbitrary transmission loss. *arXiv preprint arXiv:2312.12614*, 2023.
- [ABM⁺23] Rene Allerstorfer, Harry Buhrman, Alex May, Florian Speelman, and Philip Verduyn Lunel. Relating non-local quantum computation to information theoretic cryptography. *arXiv preprint arXiv:2306.16462*, 2023.
- [AC02] Mark Adcock and Richard Cleve. A quantum Goldreich-Levin theorem with cryptographic applications. In *STACS 2002*, pages 323–334. Springer, 2002.
- [ACCM25] Vahid Asadi, Richard Cleve, Eric Culf, and Alex May. Linear gate bounds against natural functions for position-verification. *Quantum*, 9:1604, January 2025.
- [ACH⁺24] Harriet Apel, Toby Cubitt, Patrick Hayden, Tamara Kohler, and David Pérez-García. Security of position-based quantum cryptography limits hamiltonian simulation via holography. *arXiv preprint arXiv:2401.09058*, 2024.
- [ACM25] Vahid R. Asadi, Eric Culf, and Alex May. Rank Lower Bounds on Non-Local Quantum Computation. In Raghu Meka, editor, *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, volume 325 of *Leibniz In-*

- ternational Proceedings in Informatics (LIPIcs)*, pages 11:1–11:18, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
- [AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC 1997*, pages 284–293, 1997.
 - [ADV⁺25] Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. More asymmetry yields faster matrix multiplication. In *SODA*, 2025.
 - [AGG⁺24] Vahid R. Asadi, Alexander Golovnev, Tom Gur, Igor Shinkar, and Sathyawageeswar Subramanian. Quantum worst-case to average-case reductions for all linear problems. In *SODA*, pages 2535–2567. SIAM, 2024.
 - [AGGS22] Vahid R. Asadi, Alexander Golovnev, Tom Gur, and Igor Shinkar. Worst-case to average-case reductions via additive combinatorics. In *STOC 2022*, pages 1566–1574. ACM, 2022.
 - [AGLL25] Prabhanjan Ananth, Vipul Goyal, Jiahui Liu, and Qipeng Liu. Unclonable secret sharing. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 129–157. Springer, 2025.
 - [Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *STOC 1996*, pages 99–108. ACM, 1996.
 - [AK25] Divesh Aggarwal and Dexter Kwan. Quantum worst-case to average-case reduction for matrix-vector multiplication, 2025.
 - [Aka08] Adi Akavia. Learning significant fourier coefficients over finite abelian groups. In Ming-Yang Kao, editor, *Encyclopedia of Algorithms - 2008 Edition*. Springer, 2008.
 - [AKL⁺24] Vahid R Asadi, Kohdai Kuroiwa, Debbie Leung, Alex May, Sabrina Pasterski, and Chris Waddell. Conditional disclosure of secrets with quantum resources. *arXiv preprint arXiv:2404.14491*, 2024.
 - [AMN98] Yossi Azar, Rajeev Motwani, and Joseph Seffi Naor. Approximating probability distributions using small sample spaces. *Combinatorica*, 18(2):151–171, 1998.

- [AV21] Benny Applebaum and Prashant Nalini Vasudevan. Placing conditional disclosure of secrets in the communication complexity universe. *Journal of Cryptology*, 2021.
- [BABBB21] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. The average-case complexity of counting cliques in erdős-rényi hypergraphs. *SIAM Journal on Computing*, pages FOCS19–39, 2021.
- [BBB19] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. The average-case complexity of counting cliques in Erdős–Rényi hypergraphs. In *FOCS 2019*, pages 1256–1280. IEEE, 2019.
- [BBF03] Stephane Beauregard, Gilles Brassard, and Jose M. Fernandez. Quantum arithmetic on Galois fields, 2003.
- [BCF⁺14] Harry Buhrman, Nishanth Chandran, Serge Fehr, Ran Gelles, Vipul Goyal, Rafail Ostrovsky, and Christian Schaffner. Position-based quantum cryptography: Impossibility and constructions. *SIAM Journal on Computing*, 2014.
- [BCGL92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michel Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.
- [BCS22] Andreas Bluhm, Matthias Christandl, and Florian Speelman. A single-qubit position verification protocol that is secure against multi-qubit attacks. *Nature Physics*, 2022.
- [Bea97] Robert Beals. Quantum computation of Fourier transforms over symmetric groups. In *STOC 1997*, page 48–53. ACM, 1997.
- [BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.
- [BFS86] László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *27th Annual Symposium on Foundations of Computer Science (SFCS 1986)*. IEEE, 1986.
- [BFSS13] Harry Buhrman, Serge Fehr, Christian Schaffner, and Florian Speelman. The garden-hose model. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, 2013.

- [BHM⁺25] Andreas Bluhm, Simon Höfer, Alex May, Mikka Stasiuk, Philip Verduyn Lunel, and Henry Yuen. A complexity theory for non-local quantum computation. *arXiv preprint arXiv:2505.23893*, 2025.
- [BK11] Salman Beigi and Robert König. Simplified instantaneous non-local quantum computation with applications to position-based cryptography. *New Journal of Physics*, 2011.
- [BLR90] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. In *STOC 1990*, pages 73–83. ACM, 1990.
- [BRSV17] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-case fine-grained hardness. In *STOC 2017*, pages 483–496. ACM, 2017.
- [BRSV18] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Proofs of work from worst-case assumptions. In *CRYPTO 2018*, pages 789–819. Springer, 2018.
- [BRTW14] Eli Ben-Sasson, Noga Ron-Zewi, Madhur Tulsiani, and Julia Wolf. Sampling-based proofs of almost-periodicity results and algorithmic applications. In *ICALP 2014*, pages 955–966. Springer, 2014.
- [BT06] Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1):1–106, 2006.
- [Bv06] Harry Buhrman and Robert Špalek. Quantum verification of matrix products. In *SODA 2006*, page 880–889. SIAM, 2006.
- [CGL15] Raphael Clifford, Allan Grønlund, and Kasper Green Larsen. New unconditional hardness results for dynamic and online problems. In *FOCS 2015*, pages 1089–1107. IEEE, 2015.
- [CGMO09] Nishanth Chandran, Vipul Goyal, Ryan Moriarty, and Rafail Ostrovsky. Position based cryptography. In *Annual International Cryptology Conference*, pages 391–407. Springer, 2009.
- [Cha02] Mei-Chu Chang. A polynomial bound in Freiman’s theorem. *Duke Mathematical Journal*, 113(3):399 – 419, 2002.

- [CKK12] Andrew M. Childs, Shelby Kimmel, and Robin Kothari. The quantum query complexity of read-many formulas. In *ESA 2012*, pages 337–348. Springer, 2012.
- [CKL18] Diptarka Chakraborty, Lior Kamma, and Kasper Green Larsen. Tight cell probe bounds for succinct boolean matrix-vector multiplication. In *STOC 2018*, pages 1297–1306, 2018.
- [CKLM18] Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation beats richness: New data-structure lower bounds. In *STOC 2018*, pages 1013–1020, 2018.
- [CKPG23] George Cowperthwaite, Adrian Kent, and Damian Pitalua-Garcia. Towards a proof-of-principle experimental demonstration of quantum position verification: working notes. *arXiv preprint arXiv:2309.10070*, 2023.
- [CL15] Kaushik Chakraborty and Anthony Leverrier. Practical position-based quantum cryptography. *Physical Review A*, 92(5):052304, 2015.
- [CM22] Sam Cree and Alex May. Code-routing: a new attack on position-verification. *arXiv preprint arXiv:2202.07812*, 2022.
- [CM23] Joy Cree and Alex May. Code-routing: a new attack on position verification. *Quantum*, 2023.
- [CS10] Ernie Croot and Olof Sisask. A probabilistic technique for finding almost-periods of convolutions. *Geometric and Functional Analysis*, 20(6):1367–1396, 2010.
- [CVDNT98] Richard Cleve, Wim Van Dam, Michael Nielsen, and Alain Tapp. Quantum entanglement and the communication complexity of the inner product function. In *NASA International Conference on Quantum Computing and Quantum Communications*, pages 61–74. Springer, 1998.
- [DKKS21] Pavel Dvořák, Michal Koucký, Karel Král, and Veronika Slívová. Data structures lower bounds and popular conjectures. In *ESA 2021*. LIPIcs, 2021.
- [DLV20] Mina Dalirrooyfard, Andrea Lincoln, and Virginia Vassilevska Williams. New techniques for proving fine-grained average-case hardness. In *FOCS 2020*, pages 774–785. IEEE, 2020.

- [dW00] Ronald de Wolf. Characterization of non-deterministic quantum query and quantum communication complexity. In *Proceedings of the 15th Annual IEEE Conference on Computational Complexity*, 2000.
- [dW03] Ronald de Wolf. Nondeterministic quantum query and communication complexities. *SIAM Journal on Computing*, 2003.
- [EFPS24] Llorenç Escolà-Farràs, Léo Colisson Palais, and Florian Speelman. A quantum cloning game with applications to quantum position verification. *arXiv preprint arXiv:2410.22157*, 2024.
- [FF93] Joan Feigenbaum and Lance Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.
- [FHM01] Gudmund Skovbjerg Frandsen, Johan P. Hansen, and Peter Bro Miltersen. Lower bounds for dynamic algebraic problems. *Information and Computation*, 171(2):333–349, 2001.
- [Fre77] Rusins Freivalds. Probabilistic machines can use less running time. In *IFIP 1977*, pages 839–842, 1977.
- [GC19] Alvin Gonzales and Eric Chitambar. Bounds on instantaneous nonlocal quantum computation. *IEEE Transactions on Information Theory*, 66(5):2951–2963, 2019.
- [Gen10] Craig Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *CRYPTO 2010*, pages 116–137. Springer, 2010.
- [Gid18] Craig Gidney. Halving the cost of quantum addition. *Quantum*, 2:74, June 2018.
- [GIKM00] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. *Journal of Computer and System Sciences*, 2000.
- [Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- [Gol11] Oded Goldreich. Notes on Levin’s theory of average-case complexity. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 233–247. Springer, 2011.

- [GR18] Oded Goldreich and Guy Rothblum. Counting t -cliques: Worst-case to average-case reductions and direct interactive proof systems. In *FOCS 2018*, pages 77–88. IEEE, 2018.
- [Gro05] Lov K. Grover. Fixed-point quantum search. *Physical Review Letters*, 95(15):1–4, 2005.
- [GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. In *STOC 2019*, pages 193–204. ACM, 2019.
- [GSS24] Ashish Gola, Igor Shinkar, and Harsimran Singh. Matrix multiplication reductions. In *RANDOM*, pages 34–1. Schloss Dagstuhl, 2024.
- [Gue19] Gian Giacomo Guerreschi. Repeat-until-success circuits with fixed-point oblivious amplitude amplification. *Physical Review A*, 99:022306 1– 022306 10, February 2019.
- [Haa19] Jeongwan Haah. Product decomposition of periodic functions in quantum signal processing. *Quantum*, 3:190, October 2019.
- [HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009.
- [HKNS15] Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *STOC 2015*, pages 21–30, 2015.
- [HLS21] Monika Henzinger, Andrea Lincoln, and Barna Saha. The complexity of average-case dynamic subgraph counting. *Electronic Colloquium on Computational Complexity*, 2021.
- [HLS22] Monika Henzinger, Andrea Lincoln, and Barna Saha. The complexity of average-case dynamic subgraph counting. In *SODA 2022*, pages 459–498. SIAM, 2022.
- [Hoy97] Peter Hoyer. Efficient quantum transforms. *arXiv:quant-ph/9702028*, 1997.
- [HS23] Shuichi Hirahara and Nobutaka Shimizu. Hardness self-amplification: Simplified, optimized, and unified. In *STOC 2023*, pages 70–83. ACM, 2023.

- [HS25a] Shuichi Hirahara and Nobutaka Shimizu. Error-correction of matrix multiplication algorithms. In *STOC*, pages 785–794, 2025.
- [HS25b] Shuichi Hirahara and Nobutaka Shimizu. An optimal error-correcting reduction for matrix multiplication. In *ICALP*, pages 97–1. Schloss Dagstuhl, 2025.
- [IL90] Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *FOCS 1990*. IEEE, 1990.
- [Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *CCC 1995*, pages 134–147. IEEE, 1995.
- [Imp11] Russell Impagliazzo. Relativized separations of worst-case and average-case complexities for NP. In *CCC 2011*, pages 104–114. IEEE, 2011.
- [JKM13] Stacey Jeffery, Robin Kothari, and Frederic Magniez. Nested quantum walks with quantum data structures. In *Proceedings of 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1474–1485. SIAM, 2013.
- [Juk12] Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.
- [KK11] Daniel M. Kane and Samuel A. Kutin. Quantum interpolation of polynomials. *Quantum Information & Computation*, 11(1):95–103, 2011.
- [KMS11] Adrian Kent, William J Munro, and Timothy P Spiller. Quantum tagging: Authenticating location via quantum information and relativistic signaling constraints. *Physical Review A*, 2011.
- [KMSB06] Adrian P Kent, William J Munro, Timothy P Spiller, and Raymond G Beausoleil. Tagging systems, 2006. US Patent 7,075,438.
- [KN96] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1996.
- [Kot14] Robin Kothari. *Efficient algorithms in quantum query complexity*. PhD thesis, University of Waterloo, 2014.
- [KU08] Kiran S. Kedlaya and Christopher Umans. Fast modular composition in any characteristic. In *FOCS 2008*, pages 146–155. IEEE, 2008.

- [Lar12] Kasper Green Larsen. Higher cell probe lower bounds for evaluating polynomials. In *FOCS 2012*, pages 293–301. IEEE, 2012.
- [LdW22] Noah Linden and Ronald de Wolf. Average-case verification of the quantum fourier transform enables worst-case phase estimation. *Quantum*, 6:872, December 2022.
- [Le 12] François Le Gall. Improved output-sensitive quantum algorithms for boolean matrix multiplication. In *Proceedings of 23rd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SODA 2012, page 1464–1476. SIAM, 2012.
- [Lev86] Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.
- [Lip91] Richard Lipton. New directions in testing. *Distributed computing and cryptography*, 2:191–202, 1991.
- [LLQ21] Jiahui Liu, Qipeng Liu, and Luowen Qian. Beating classical impossibility of position verification. *arXiv preprint arXiv:2109.07517*, 2021.
- [LLV19] Rio LaVigne, Andrea Lincoln, and Virginia Vassilevska Williams. Public-key cryptography in the fine-grained setting. In *CRYPTO 2019*, pages 605–635. Springer, 2019.
- [Lov15] Shachar Lovett. An exposition of Sanders’ quasi-polynomial Freiman-Ruzsa theorem. *Theory of Computing*, pages 1–14, 2015.
- [Lov17] Shachar Lovett. Additive combinatorics and its applications in theoretical computer science. *Theory of Computing*, pages 1–55, 2017.
- [LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM*, 60(6):1–35, 2013.
- [LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.
- [LW17] Kasper Green Larsen and Ryan Williams. Faster online matrix-vector multiplication. In *SODA 2017*, pages 2182–2189. SIAM, 2017.
- [Mal16] Robert Malaney. The quantum car. *IEEE Wireless Communications Letters*, 2016.

- [May19] Alex May. Quantum tasks in holography. *Journal of High Energy Physics*, 2019.
- [May21] Alex May. Holographic quantum tasks with input and output regions. *Journal of High Energy Physics*, 2021.
- [May22] Alex May. Complexity and entanglement in non-local computation and holography. *Quantum*, 2022.
- [Mel23] Antonio Anna Mele. Introduction to Haar measure tools in quantum information: A beginner’s tutorial. *arXiv preprint arXiv:2307.08956*, 2023.
- [MNRS07] Frederic Magniez, Ashwin Nayak, Jeremie Roland, and Miklos Santha. Search via quantum walk. In *STOC 2007*, page 575–584. ACM, 2007.
- [MP11] David A. Meyer and James Pommersheim. On the uselessness of quantum queries. *Theoretical Computer Science*, 412(51):7068–7074, December 2011.
- [MPS20] Alex May, Geoff Penington, and Jonathan Sorce. Holographic scattering requires a connected entanglement wedge. *Journal of High Energy Physics*, 2020.
- [MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *FOCS 2004*, pages 372–381. IEEE, 2004.
- [MR06] Dana Moshkovitz and Ran Raz. Sub-constant error low degree test of almost-linear size. In *STOC 2006*, pages 21–30. ACM, 2006.
- [MRR06] Cristopher Moore, Daniel Rockmore, and Alexander Russell. Generic quantum fourier transforms. *ACM Transactions on Algorithms*, 2(4):707–723, October 2006.
- [MSS07] Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–424, 2007.
- [NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 2010.
- [NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

- [NRW22] Parker Newton, Silas Richelson, and Chase Wilson. A high dimensional Goldreich-Levin theorem. In *STOC 2022*, pages 1463–1474. ACM, 2022.
- [NS02] Ashwin Nayak and Julia Salzman. On communication over an entanglement-assisted quantum channel. In *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, pages 698–704, 2002.
- [Raz03] Alexander A Razborov. Quantum communication complexity of symmetric predicates. *Izvestiya: Mathematics*, 67(1):145, 2003.
- [RB09] Joseph M Renes and Jean-Christian Boileau. Conjectured strong complementary information tradeoff. *Physical review letters*, 103(2):020402, 2009.
- [Reg04] Oded Regev. New lattice-based cryptographic constructions. *Journal of the ACM*, 51(6):899–942, 2004.
- [Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):1–40, 2009.
- [RY20] Anup Rao and Amir Yehudayoff. *Communication Complexity: and Applications*. Cambridge University Press, 2020.
- [San12] Tom Sanders. On the Bogolyubov–Ruzsa lemma. *Analysis & PDE*, 5(3):627–655, 2012.
- [Sho94] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *FOCS 1994*, pages 124–134. IEEE, 1994.
- [Sho09] Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge, 2009.
- [Spe16] Florian Speelman. Instantaneous Non-Local Computation of Low T-Depth Quantum Circuits. In *11th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2016)*, volume 61 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:24, Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [SS25] Igor Shinkar and Harsimran Singh. A simplified reduction for error correcting matrix multiplication algorithms. In *RANDOM*, pages 34–1. Schloss Dagstuhl, 2025.

- [STV01] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.
- [TFKW13] Marco Tomamichel, Serge Fehr, Jędrzej Kaniewski, and Stephanie Wehner. A monogamy-of-entanglement game with applications to device-independent quantum cryptography. *New Journal of Physics*, 2013.
- [TS17] Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *STOC 2017*, pages 238–251, 2017.
- [Vas18] Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *ICM 2018*. World Scientific, 2018.
- [VW10] Virginia Vassilevska Williams and Ryan Williams. Subcubic equivalences between path, matrix and triangle problems. In *FOCS 2010*, pages 645–654. IEEE, 2010.
- [VZ25] Vinod Vaikuntanathan and Or Zamir. Improving algorithmic efficiency using cryptography, 2025.
- [Win16] Andreas Winter. Tight uniform continuity bounds for quantum entropies: conditional entropy, relative entropy distance and energy constraints. *Communications in Mathematical Physics*, 2016.
- [YLC14] Theodore J. Yoder, Guang Hao Low, and Isaac L. Chuang. Fixed-point quantum search with an optimal number of queries. *Physical Review Letters*, 113:210501, November 2014.
- [Yue14] Henry Yuen. A quantum lower bound for distinguishing random functions from random permutations. *Quantum Information & Computation*, 14(13–14):1089–1097, oct 2014.
- [Zha15] Mark Zhandry. A note on the quantum collision and set equality problems. *Quantum Information & Computation*, 15(7–8):557–567, may 2015.