Communication Complexity and Quantum Optimization Lower Bounds via Query Complexity

A Thesis

submitted to the

Tata Institute of Fundamental Research, Mumbai

for the degree of Doctor of Philosophy in

Computer Science

by

Suhail Sherif

Tata Institute of Fundamental Research, Mumbai

May, 2021

DECLARATION

This thesis, titled **"Communication Complexity and Quantum Optimization Lower Bounds via Query Complexity"** is a presentation of my original research work. Whatever contributions of others are involved, every effort is made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions.

The work was done under the guidance of **Professor Arkadev Chattopadhyay** at the Tata Institute of Fundamental Research, Mumbai.

Suhail Sherif

In my capacity as supervisor of the candidate's thesis, I certify that the above statements are true to the best of my knowledge.

Arkadev Chattopadhyay

Date: 11.02.2021

Acknowledgements

This thesis owes its existence to a multitude of people.

First and foremost I'd like to credit those members of society whose underappreciated hard work contributed to the easy access to essentials that I enjoyed throughout my PhD.

Secondly I'd like to thank the people throughout history who have written educational content and helped create the experience I had of getting interested in mathematics, computer science and other engaging topics. I consider myself fortunate for having a family that fostered in me a love of books that allowed me to engage with the knowledge of the world.

My research at TIFR was aided most by my advisor Arkadev Chattopadhyay. Nikhil Mande and Sagnik Mukhopadhyay have also had direct and meaningful impacts on my research. I also had a wonderful time working with Ankit Garg, Robin Kothari and Praneeth Netrapalli when I was an intern at Microsoft Research Lab India.

Less direct yet meaningful impacts came from a variety of sources. I benefited from a lot of inspiring courses taken by lecturers at TIFR. I have also found the theoretical computer science community, both at TIFR and beyond, to be an inspiring and welcoming community. My friendships with fellow students at TIFR and with people visiting TIFR were invaluable in creating a nice environment to live and work in.

I have been largely sheltered from bureaucratic concerns by a most helpful STCS office. TIFR's subsidized housing and food also made life a bit easier, perhaps, than it should have been. I apologize to the cleaner that I inadvertently scared with a toy snake that was meant to startle Arkadev (who never noticed it).

I also thank Nati Linial and Ronald de Wolf for serving on my thesis committee. I am very grateful to Ronald for enhancing the quality of this thesis with his suggestions and corrections.

Contents

Chapter 1

Introduction

A lot of computer science is dedicated to understanding the complexity of carrying out computational tasks. Quite often the task involves an input based on which the computation must give us a valid output. In the simple case of computing functions each input is associated with a single valid output, and the task is to find what output the given input corresponds to. When stated in this form, the task resembles a classification task wherein all the inputs are being classified by their outputs.

However, the actual computation done in order to solve this classification task need not resemble what we imagine when we think of classifying things. For instance, when the task is to classify a given whole number as either prime or composite, some of the most widely used classifiers do *weird* operations like tossing coins to get a random number and then repeatedly squaring that random number. A less eccentric classifier would perhaps use a lookup table, but such a lookup table would be prohibitively large as the numbers become larger. One can make this lookup table smaller by taking a more coarse approach. Indeed, one can look at the last digit. If it is 0, classify it as composite. If it is 2 or 5, then see whether it is a one digit number. If yes, classify it as prime. If no, classify it as composite. In this new and really short 'generalized lookup table' (we still are doing lookups to see which digit to look at and what to output) we have already classified large swathes of numbers. However, a specification of what to do if the last digit is not 0, 2 or 5 would be more complicated and so one would imagine that the specification of such a classifier will also become prohibitively large. But can we prove that any generalized lookup table that solves the above classification task *necessarily* has to be large? Such questions are going to play a pivotal role in this thesis.

1.1 Generalized Lookup Tables: Decision Trees

Ingo Wegener [Weg00] notes that the above notion of classification has been around for a long time, with the famous botanist Carl Linnaeus' systems of nomenclature being quite representative. Figure 1.1 shows how one can observe aspects of a member of the vegetable

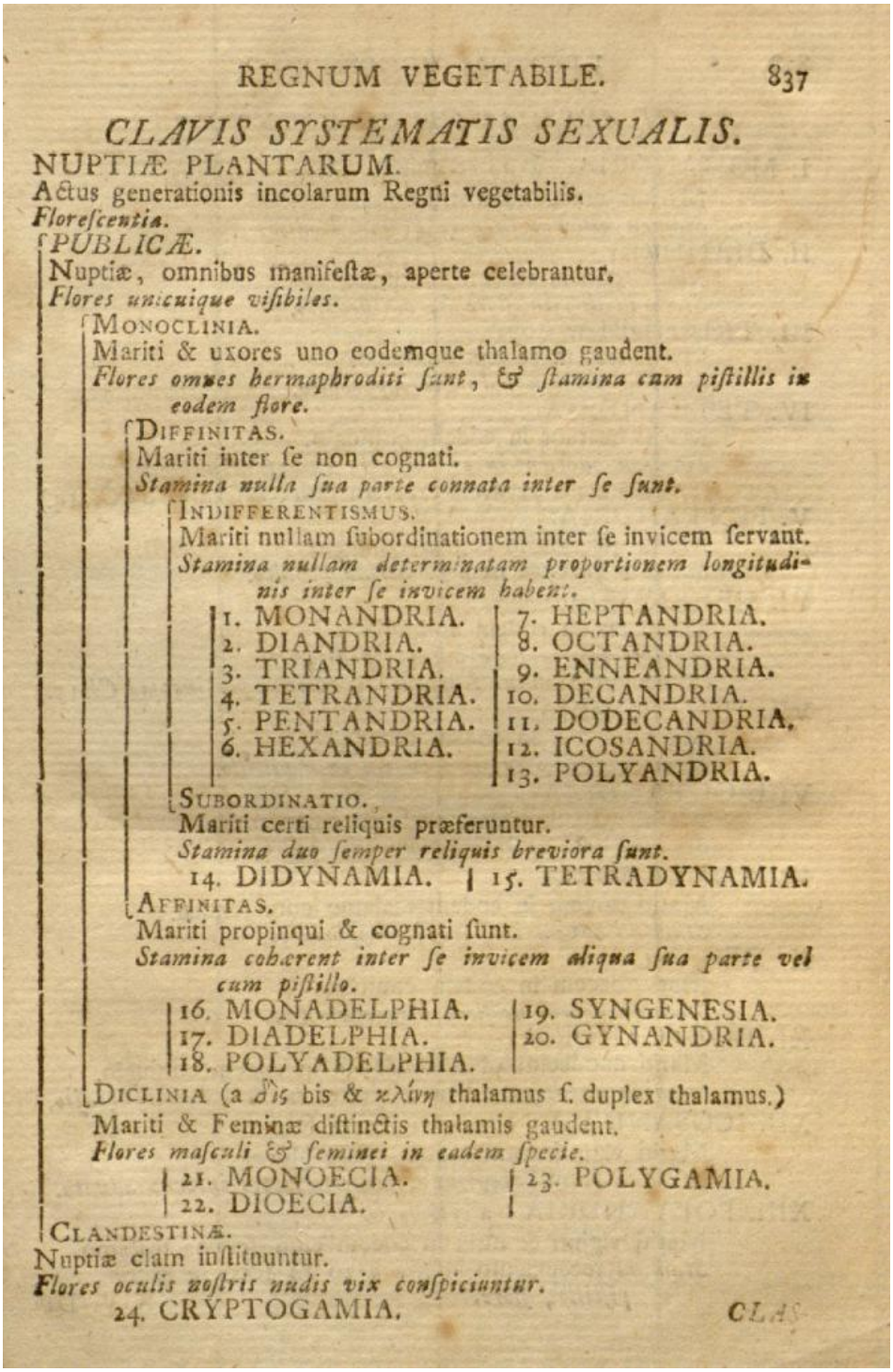
kingdom in order to compute its class.

In the context of Boolean functions which take n bits of input and output 1 bit, which will be the focus of a majority of this thesis, it was Lee [Lee59] who introduced the notion of a Branching Program which is a quite similar model to what we attempted above for classifying primes. Akers [Ake78] turned it into a graphical representation called a Binary Decision Diagram. Both of these are aimed at giving small descriptions of functions instead of specifying the values of the functions for each of the 2^n inputs. We will be dealing with a simpler model known as Decision Trees. Figure 1.2 shows a decision tree. To look up the classification of an input, you start at the root and look at what index is specified at the root. You then look at that bit of the input. If it is a 0, you go to the left child in the decision tree, otherwise you go to the right child. You then continue to look up bits and traverse the tree until you reach a leaf, where the classification of the input is specified.

Consider the model of computation wherein you want to compute a function but do not know the input. You can access the input by querying its bits. So you create an algorithm that will go about querying bits and computing the value of the function. Say you want to minimize the number of queries you need to make in the worst case in order to compute the function. Note that if the function has a decision tree T, then that gives us an obvious algorithm that would just follow the correct path in the tree and compute the function. If q is the maximum depth of a leaf of T, then the algorithm also makes q queries in the worst case. Conversely, given any algorithm that computes the function we can create a decision tree out of it by noting which bit it queries first, and then depending on the value of the queried bit, which bit it queries next and so on. Hence we use the term Decision Tree and Query Algorithm interchangeably. The natural measure of a query algorithm is the number of queries it makes. The corresponding measure of a decision tree is the height of the decision tree. The *decision tree complexity* (or *query complexity*) of a function f is the minimum, among all decision trees computing f, of the height of the decision tree.

The notion of a decision tree is also very versatile. The decision tree in Figure 1.2 has each node querying a single bit of the input. Depending on the use case, one can also study decision trees that allow more complicated queries. For instance, part of this thesis works with parity decision trees. At a node of a parity decision tree one can choose a subset of the bits and query the parity of those bits. In another part of this thesis we will find ourselves querying values and gradients of a real-valued function. This is hard to think of as a decision tree given the abundance of possible answers one can get from a query. It is more natural to think of it as a query algorithm. In the case of quantum query algorithms it is hard to fathom what a corresponding decision tree would be and so we will stick to the term quantum query algorithm. Despite it being such a simple model of computation, decision trees are *very* widely used in a variety of fields of theoretical computer science.

- The most direct application is in scenarios where one's computation involves a resource



REGNUM VEGETABILE. 837

CLAVIS SYSTEMATIS SEXUALIS.

NUPTIÆ PLANTARUM.

Actus generationis incolarum Regni vegetabilis.

Floreſcentia.

- *PUBLICÆ.*
  Nuptiæ, omnibus manifeſtæ, aperte celebrantur.
  Flores unicuique viſibiles.
  - MONOCLINIA.
    Mariti & uxores uno eodemque thalamo gaudent.
    Flores omnes hermaphroditi ſunt, & ſtamina cum piſtillis in eodem flore.
    - DIFFINITAS.
      Mariti inter ſe non cognati.
      Stamina nulla ſua parte connata inter ſe ſunt.
      - INDIFFERENTISMUS.
        Mariti nullam ſubordinationem inter ſe invicem ſervant.
        Stamina nullam determinatam proportionem longitudinis inter ſe invicem habent.
        1. MONANDRIA.
        2. DIANDRIA.
        3. TRIANDRIA.
        4. TETRANDRIA.
        5. PENTANDRIA.
        6. HEXANDRIA.
        7. HEPTANDRIA.
        8. OCTANDRIA.
        9. ENNEANDRIA.
        10. DECANDRIA.
        11. DODECANDRIA.
        12. ICOSANDRIA.
        13. POLYANDRIA.
      - SUBORDINATIO.
        Mariti certi reliquis præferuntur.
        Stamina duo ſemper reliquis breviora ſunt.
        14. DIDYNAMIA. | 15. TETRADYNAMIA.
    - AFFINITAS.
      Mariti propinqui & cognati ſunt.
      Stamina cohærent inter ſe invicem aliqua ſua parte vel cum piſtillo.
      16. MONADELPHIA.
      17. DIADELPHIA.
      18. POLYADELPHIA.
      19. SYNGENESIA.
      20. GYNANDRIA.
  - DICLINIA (a δὶς bis & κλίνη thalamus ſ. duplex thalamus.)
    Mariti & Feminæ diſtinctis thalamis gaudent.
    Flores maſculi & feminei in eadem ſpecie.
    21. MONOECIA.
    22. DIOECIA.
    23. POLYGAMIA.
- CLANDESTINÆ.
  Nuptiæ clam inſtituuntur.
  Flores oculis noſtris nudis vix conſpiciuntur.
  24. CRYPTOGAMIA.

CLAS

Figure 1.1: Carl Linnaeus' 'sexual system' of classifying plants, from his book [Lin59] wherein he tries to classify everything in nature (as per the book, nature is split into God and everything in the physical world). The page shown shows how to classify the members of the vegetable kingdom into classes. It first queries whether the flowers are visible. If they are not, the output is Cryptogamia. If they are, it queries whether the flowers are hermaphrodites (having both stamens and pistils in the same flowers). It continues in a similar vein.

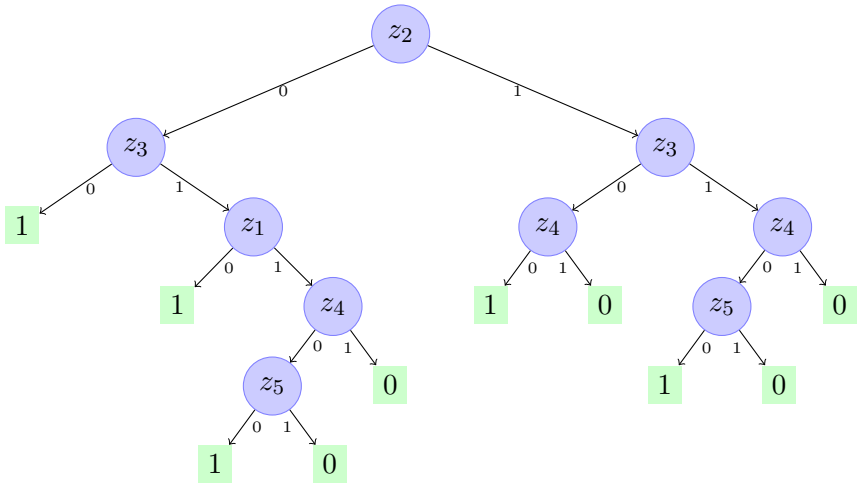


Figure 1.2: A decision tree computing the function that takes 5 bits $z_1z_2z_3z_4z_5$ as input and outputs 1 if there are two consecutive 0s and outputs 0 otherwise. A decision diagram would have merged the leftmost z_4 query and the rightmost z_4 query because the subtrees below them are the same. Decision trees are defined as trees, so they do not allow this merging. Since there is leaf at depth 5, the depth of this tree is 5.

that is expensive to access. In such a scenario it is very useful to know the minimum number of times one needs to access the resource. This is exactly what query complexity deals with. Some nice examples of this are as follows.

- The folklore lower bound of $\Omega(n \log n)$ for sorting with comparison queries is a nearly immediate consequence of framing the question in the decision tree model.
- The decision tree model is also useful in solving fun puzzles such as *"You have 12 balls that are all of the same weight, except for one ball that is either slightly heavier or slightly lighter. You wish to identify both the defective ball and whether it is heavier or lighter than the rest. What is the minimum number of weighings you need to do on a beam balance in order to accomplish this?"*

- We find it very challenging to compare the power of various Turing Machine models in order to prove complexity class separations. Not surprisingly it is much easier to compare the power of query complexity models and prove separations. Remarkably, it is very common to use these query complexity separations in order to show oracle separations in the Turing Machine models! See for instance [BGS75, Ver99, RT19].

- It is known that if a Boolean function f has small decision tree complexity, then f is computed by a low-degree polynomial. Conversely every (total) Boolean function f that is computed by a low-degree polynomial can be computed by a small-depth decision tree. Such analyses have led to very interesting results such as the fact that approximating a

total Boolean function with a polynomial can't lower the degree required much more than if you were to compute it exactly with a polynomial [NS92].

- Communication Complexity is a field with its own rich set of applications. Decision trees have played a major role in a number of major results in communication complexity. Lower bounds on the decision tree complexity of certain tasks are shown to hold for related tasks in communication complexity, a fact that has enabled a lot of progress in some fields. We will see more on this shortly.

- Quantum query complexity has served as a fruitful testing ground for analyzing the power of quantum computation. Famous algorithms such as Grover's unstructured search algorithm and Shor's period finding algorithm are actually quantum query algorithms that vastly outperform known classical algorithms. Another interesting speedup is that a single quantum query to a real-valued smooth function can give an arbitrarily close approximation to the gradient of the function at a point. This could have consequences for efficient algorithms for machine learning. We will see more on this as well.

We now take a closer look at communication complexity, the setting of the first result of this thesis.

1.2 Communication Complexity

Communication complexity was introduced by Yao [Yao79] in order to analyze how much information processors needed to exchange while computing a function whose inputs are distributed between them.[1] He noted that this model has a combinatorial flavour and showed a connection with combinatorial rectangles. Over the course of the next few decades, communication complexity has seen applications in various other fields including lower bounds in streaming algorithms, data structures, formulas and extension complexity, with some being the best known lower bounds to date. Let us take an informal and imprecise look at a few applications.

- Lower bounds on the memory requirement of streaming algorithms: One could think of the data being stored by an algorithm as a message being sent by the algorithm to its future self. Such an interpretation was used by the seminal work of Alon, Matias and Szegedy [AMS99] to show lower bounds on the memory needed by streaming algorithms using lower bounds on one-way communication complexity.

- Data structure lower bounds: Say you have a data structure which is very efficient (updates and queries can be performed with very little cost). Then a party who has

[1] He was inspired by a similar notion by Abelson [Abe78] which dealt only with smooth functions over real-valued inputs.

received all updates can send a *short* message to a party who has received most updates, enabling the latter to answer queries "better" than it previously could. If your function is complicated enough that it needs large communication in order to answer "better", then it cannot have such an efficient data structure. This method was introduced by [MNSW98] and has been used to achieve state-of-the-art lower bounds such as that of [LWY20].

- Formula size lower bounds: Consider a communication protocol between two parties that distinguishes a set $A \subseteq \{0,1\}^n$ from its complement $\bar{A}$ in the sense that if the former party has an input $x \in A$ and the latter an input $y \in \bar{A}$, the protocol outputs an $i \in [n]$ such that $x_i \neq y_i$. Such protocols can be shown to be *equivalent* to formulas that compute the function $\mathbb{1}_A$ that outputs 1 if and only if its input x belongs to A [KW88]. This has led to quite strong formula lower bounds via communication complexity (see for instance [DM18]).

- Extended formulation size of a polytope is known to be equivalent to nonnegative rank of an associated matrix and the nonnegative rank can be lower bound using communication complexity [Yan91]. Subsequent works have resulted in a tight relation between nonnegative rank and communication cost [FFGT15] and recently this was used in a breakthrough extension complexity lower bound [FMP$^+$15].

- The optimal number of bits of communication required to compute a function with worst-case advantage greater than half is, up to an additive constant, the logarithm of the sign rank of a matrix associated with the function [PS86]. This link has contributed to a deeper understanding of sign rank (see for instance [BMT19, HHL20]).

- The deterministic communication complexity of F is known to be quadratically related to the logarithm of the nonnegative rank of an associated matrix [Lov90]. As a consequence for every 0/1 matrix M, the logarithms of the nonnegative ranks of M and $J - M$ cannot be vastly different.

The last three applications above are of particular interest to us as we will be dealing with variants of them. Let us take a closer look at communication complexity to enable ourselves to talk about their variants. We formally define the model in more detail in Chapter 2.

In the simple case of two parties communicating, we default to the names Alice and Bob to represent the two. We wish to study how much they need to interact in order to solve a computational task. A natural task would be to compute a function that takes two inputs, where one input is held by Alice and one input held by Bob, and outputs 1 bit. We represent such 'functions with two inputs' by capital letters. So let F be a function taking two inputs, one held by Alice and from the set $\mathcal{X}$, the other held by Bob and from the set $\mathcal{Y}$. Since they both know the function F they will be required to compute, they agree on a protocol to follow beforehand. The cost of the protocol is the maximum number of bits exchanged in the worst

case during its execution. The *deterministic communication complexity* of F is the minimum cost among protocols that compute F, and is denoted $\mathsf{D}^{\mathsf{cc}}(F)$. Note that the protocol can also be specified as a tree. Each internal node is associated with either Alice or Bob, and is labeled with a corresponding function that the relevant party should use in order to figure out what bit to communicate and how the protocol should then proceed. The output of the protocol will then be specified at the leaves of the tree.

> A communication protocol as defined above is the same as a decision tree in which at each node one is allowed to query arbitrary functions of Alice's inputs (making the node an Alice node) or arbitrary functions of Bob's inputs (making the node a Bob node).

The function F can be specified by its outputs on the $|\mathcal{X}| \times |\mathcal{Y}|$ inputs. This is best represented in the form of a matrix M_F. The rows of M_F are indexed by $\mathcal{X}$, and the columns are indexed by $\mathcal{Y}$. The matrix entry $M_F[x, y]$ is $F(x, y)$. (See Figure 2.1b for an example.)

It is known that a cost-c deterministic protocol computing F implies that the rank of M_F is at most 2^c (see Lower bound 2). It is still unknown whether the rank of M_F being 2^c implies that there is a communication protocol of cost $c^{O(1)}$. It is conjectured to be so, and this conjecture is called the Log-Rank Conjecture (LRC, see Conjecture 3.2.3). It is worth noting that similar statements are known to be true, three of which we have mentioned in the applications above. The last of them is akin to saying that the Log-Nonnegative-Rank Conjecture is true.[2]

Akin to the LRC, the analogous conjecture for bounded error communication complexity was still open. A cost-c private-coin randomized protocol [3] computing F to within error ϵ is proof that the ϵ-approximate rank of M_F is at most 2^c. (The ϵ-approximate rank of a matrix M, $\mathsf{rank}_\epsilon(M)$, is defined as the minimum rank of a matrix M', where $\|M' - M\|_\infty \leq \epsilon$.) It was unknown whether the 1/3-approximate rank of M_F being 2^c implies that there is a private coin randomized communication protocol of cost $c^{O(1)}$. The Log-Approximate-Rank Conjecture, posed by Lee and Shraibman [LS09b] (see Conjecture 3.2.14), conjectures that it does. Interestingly, Gavinsky and Lovett [GL14] showed that the Log-Approximate-Rank Conjecture implies the Log-Rank Conjecture. Another interesting consequence of the conjecture would be that the randomized communication complexity of any total function would be within a polynomial of its quantum communication complexity. Having shown that the deterministic communication complexity is at most $\tilde{O}(\sqrt{\mathsf{rank}(M_F)})$, Lovett [Lov14] asked whether it could be generalized to saying that the randomized communication complexity is at most the square root of the approximate rank. This is still unknown and the best upper bound is just the approximate rank itself [GS19] (see Theorem 3.2.15). It is worth noting that both the LRC and the LARC are known to hold for some structured classes of functions. We will elaborate

[2] The nonnegative rank of a matrix M, denoted $\mathsf{rank}^+(M)$, is defined as the minimum number of nonnegative rank-1 matrices needed to sum to M.

[3] Both Alice and Bob have access to private sources of randomness that the protocol can use to aid its computation. There is a probability that the protocol will output the wrong answer, but we want that for every input (x, y) the protocol outputs the correct answer with probability at least $1 - \epsilon$.

on this shortly.

On the other hand, the Set Disjointness function on $2n$ bits is known to need $\Omega(n)$ bits of communication even with randomness, and the logarithm of its approximate rank is merely $O(\sqrt{n})$ [KS92, Raz03]. This shows that the exponent in the LARC conjecture is at least 2. More recently, Göös, Jayram, Pitassi and Watson [GJPW17] came up with a 4th power separation between randomized communication complexity and the logarithm of the approximate rank.

There are other conjectures along similar lines, notably the Log-Approximate-Nonnegative Rank Conjecture [KMSY14] (see Conjecture 3.2.17) and Grolmusz's Conjecture [Gro97] (see Conjecture 3.2.28). We leave their details out of the introduction.

Although approximate ranks can be used to lower bound bounded error randomized communication complexity, many other lower bound measures are also known. There is still a lot left to be answered about the relative strengths of the lower bounds. We discuss them in much more detail in Chapter 3. A few observations and expositions are included there that have not been observed and exposited before in the literature, to the best of our knowledge.

There is a very important class of functions called composed functions. This class of functions has been widely used in order to prove numerous separations between complexity classes. These functions have a lot of structure that makes them easier to analyze. The structure is also expected to reduce the study of their communication complexities to that of query complexities (and in some cases it is known that this does happen). However, the analog of the LARC (and even the LRC) in the standard query world is known to be true. Namely, the deterministic and randomized query complexities are polynomially related to the polynomial degree. This automatically makes the LRC and LARC true for those composed functions wherein the communication complexity does reduce to the query complexity. However there is one important class of composed functions where the communication complexity provably does *not* reduce to query complexity. Rather it is expected to reduce to the model of query complexity wherein "parity queries" are allowed, and where the analogues of the LRC and LARC are not as well understood. We now take a look at such composed functions and that query model.

1.2.1 XOR Functions and Parity Decision Trees

There is a vast body of literature on communication complexity that involves composed functions. Given a function $f : \{0,1\}^n \to \{0,1\}$ and a function $G : \{0,1\}^b \times \{0,1\}^b \to \{0,1\}$, the function $f \circ G^n$ is defined as the composition of f with G^n. That is, $f \circ G^n : (\{0,1\}^b)^n \times (\{0,1\}^b)^n \to \{0,1\}$ maps

$$((x_1, x_2, \ldots, x_n), (y_1, y_2, \ldots, y_n)) \mapsto f(G(x_1, y_1), G(x_2, y_2), \ldots, G(x_n, y_n)).$$

We use $f \circ G$ to denote $f \circ G^{\mathrm{arity}(f)}$. When trying to compute $f \circ G$, Alice and Bob could follow the naïve protocol which uses a decision tree for f. When the decision tree asks for the

ith bit of the input to f, Alice and Bob compute G on their inputs (x_i, y_i), find the ith bit of input to f and then continue down the decision tree. If the decision tree makes q queries in the worst case and the communication cost of G is c, the cost of the protocol is at most qc.

For various choices of G, it has been shown that this protocol is optimal [RM97, CKLM19]. Hence in such cases a lower bound on the communication complexity of $f \circ G$ follows from a lower bound on the query complexity of f. This phenomenon is referred to as a deterministic lifting theorem with gadget G. Similar lifting theorems are known in various models (randomized, non-deterministic) and with various gadgets [GPW17, GLM$^+$16]. Such connections between query and communication lower bounds have led to a variety of lower bounds in communication complexity with very wide-ranging applications (for example [RM97, GJPW17, GJW18]).

Here we will consider the case when the gadget G is the function $\mathsf{XOR} : \{0,1\} \times \{0,1\} \to \{0,1\}$ defined as $\mathsf{XOR}(x, y) = 0$ if and only if $x = y$. Many natural functions are actually functions that are composed with XOR. For instance, $\mathsf{EQ} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, which asks whether two n-bits strings are equal, is the n-bit NOR composed with XOR. We refer to these functions as XOR functions.

There is still a lot of work left in proving optimal lifting theorems for the XOR gadget. One source of difficulty here is that $f \circ \mathsf{XOR}$ can require far fewer bits of communication than the number of queries f requires. In particular, if f is the parity function on n bits, then $f \circ \mathsf{XOR}$ is simply the parity function on $2n$ bits. This can be solved with 2 bits of communication, although the query complexity of f is n. However, we can change our query model so that we may ask it parity queries as well. For any subset S of the n input bits, we can find out with one query whether the number of 1s present in those input bits is even or odd. This query model is referred to as Parity Decision Trees (PDTs). The complexity of f in this query model is denoted $\mathsf{D}^{\oplus}(f)$. The question then is whether $\mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR})$ can be far less than $\mathsf{D}^{\oplus}(f)$. This question was answered by Hatami, Hossein and Lovett [HHL18] who showed that the two are polynomially equivalent: $\mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR}) \geq \mathsf{D}^{\oplus}(f)^{1/6}$. It is still open whether $\mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR}) \geq \Omega(\mathsf{D}^{\oplus}(f))$. In the case of randomized algorithms, even a polynomial equivalence is unknown.

On the other hand, f and $f \circ \mathsf{XOR}$ can be seen to share a lot of mathematical structure. (The quantities we speak of here are introduced in Section 3.3.) For instance, the Fourier sparsity of f is equal to the rank of $M_{f \circ \mathsf{XOR}}$. Given this fact, Tsang, Wong, Xie and Zhang [TWXZ13] tried to prove the LRC for XOR functions by trying to prove that $\mathsf{D}^{\oplus}(f)$ is at most polynomially more than the logarithm of the Fourier sparsity of f. Along with the above stated result of [HHL18] stating that $\mathsf{D}^{\oplus}(f)$ and $\mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR})$ are polynomially related, we can see that the conjecture that $\mathsf{D}^{\oplus}(f)$ is at most polynomially more than the logarithm of the Fourier sparsity of f is equivalent to the LRC for XOR functions. The LARC for the class of XOR functions is not known to be equivalent to an analogous conjecture concerning parity decision trees since there are no randomized lifting theorems for PDTs. The natural analogue would be that $\mathsf{R}^{\oplus}_{\epsilon}(f)$ is bounded above by a polynomial in the ϵ-approximate sparsity, where $\mathsf{R}^{\oplus}_{\epsilon}(f)$ is

the ϵ-error randomized PDT complexity of f and ϵ-approximate sparsity is defined akin to ϵ-approximate rank. If one expects the randomized XOR lifting theorem to be true (as many do), one must also expect the above conjecture and the LARC to share the same fate.

Apart from their relevance to communication complexity, PDTs have been studied for their insights into the structure of Boolean functions. Green and Sanders [GS08] proved that a Boolean function with small spectral norm can be written as a sum of a "few" indicator functions of affine spaces, where "few" here is doubly exponential in the spectral norm. Shpilka, Tal and Volk [STV17] improved this to singly exponential by showing that any function $f : \{0,1\}^n \to \{0,1\}$ with $\left\|\widehat{f}\right\|_1 = A$ has a parity decision tree with at most $2^A n^{2A}$ leaves. They hoped that this could be improved further to $\mathrm{poly}(n, A)$ leaves. This improvement, along with the LRC, would follow should one prove the following conjecture of Tsang, Wong, Xie and Zhang [TWXZ13]. The *parity kill number* of f is defined as

$$\mathsf{C}^{\oplus}_{\mathsf{min}}(f) := \min\{\text{co-dim}(S) | S \text{ is an affine subspace on which } f \text{ is constant}\}.$$

The conjecture (see Conjecture 3.3.9) is that the parity kill number of any Boolean function f is bounded above by a polynomial in the logarithm of $\left\|\widehat{f}\right\|_1$. O'Donnell, Wright, Zhao, Sun and Tan[OWZ+14] came up with a function showing that the degree of the polynomial in the conjecture has to be at least 1.58.

We now move on to state the results from this part of the thesis.

1.2.2 Our Contributions: Separating Log-Approximate-Rank and Communication

Our major contribution is the conception of SINK as a counterexample to many conjectures (see Figure 1.3). The details of this work are in Chapter 4. We give a quick run-through here.

SINK takes n bits of input, where $n = \binom{m}{2}$ for $m \in \mathbb{N}$, and it outputs a single bit. We show the following properties for SINK.

Theorem 1.2.1. *Let $F := \mathsf{SINK} \circ \mathsf{XOR}$ and $\bar{F}$ be the negation of F. We have the following properties.*

1. $\left\|\widehat{\mathsf{SINK}}\right\|_1 = \left\|\widehat{F}\right\|_1 \leq m$.
2. $\forall \epsilon > 0$, $\mathsf{rank}_\epsilon(F) \leq O_\epsilon(m^4)$.
3. $\forall \epsilon > 0$, $\mathsf{rank}^+_\epsilon(F) \leq O_\epsilon(m^5)$.
4. $\forall m > 2$, $\mathsf{C}^{\oplus}_{\mathsf{min}}(\mathsf{SINK}) = \lceil 2m/3 \rceil$.
5. $\mathsf{R}^{\oplus}_{1/3}(\mathsf{SINK}) = \Theta(m)$.
6. *For some* $\epsilon > 0$, $\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \geq \Omega(\mathsf{rank}^+_\epsilon(\bar{F})) \geq \Omega(m)$.

The above theorem has the following consequences.

Refuting the Log-Approximate-Rank Conjecture (Conjecture 3.2.14): The function F gives an exponential separation between $\mathsf{R}^{\mathsf{cc}}_{1/3}(F)$ and $\log \mathsf{rank}_{1/3}(F)$. In fact, $\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \geq \Omega(\mathsf{rank}_{1/3}(F)^{1/4})$.

Refuting the Strong Log-Approximate-Nonnegative-Rank Conjecture (Conjecture 3.2.18): The function F gives an exponential separation between $\mathsf{R}^{\mathsf{cc}}_{1/3}(F)$ and $\log \mathsf{rank}^{+}_{1/3}(F)$. In fact, $\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \geq \Omega(\mathsf{rank}^{+}_{1/3}(F)^{1/5})$.

Understanding Rank and Nonnegative Rank: The function F exhibits, for some constant $\epsilon > 0$, an exponential gap between $\log \mathsf{rank}_{\epsilon}(\bar{F})$ and $\log \mathsf{rank}^{+}_{\epsilon}(\bar{F})$. Lee [Lee12] had asked whether such a separation was possible. Furthermore, let M be the matrix that is close to M_F that witnesses its small approximate nonnegative rank. We thus get a matrix M whose entries are from $[0, \epsilon] \cup [1-\epsilon, 1]$ such that $\log \mathsf{rank}^{+}(M)$ is exponentially separated from $\log \mathsf{rank}^{+}(J - M)$.

- We can slightly modify F to get a function F' such that $\log \mathsf{rank}_{\epsilon}(F')$ is exponentially separated from the minimum of $\log \mathsf{rank}^{+}_{\epsilon}(F')$ and $\log \mathsf{rank}^{+}_{\epsilon}(\bar{F'})$.

Refuting Grolmusz's Conjecture (Conjecture 3.2.28): The function F gives an exponential separation between $\mathsf{R}^{\mathsf{cc}}_{1/3}(F)$ and $\log \left\|\widehat{F}\right\|_1$. In fact, $\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \geq \Omega(\left\|\widehat{F}\right\|_1)$.

Refuting the Parity Kill Number Conjecture (Conjecture 3.3.9): The function SINK gives an exponential separation between $\mathsf{C}^{\oplus}_{\mathsf{min}}(\mathsf{SINK})$ and $\log \left\|\widehat{F}\right\|_1$. In fact, $\mathsf{C}^{\oplus}_{\mathsf{min}}(F) \geq \Omega(\left\|\widehat{F}\right\|_1)$.

- Note that this implies that the number of leaves needed by a parity decision tree computing F is at least $2^{2m/3}$. Hence [STV17] cannot improve their structure theorem regarding functions with small spectral norm via their method. (See the end of the previous subsection, on page 10.)

Implications for Quantum Communication Complexity: Since quantum communication complexity is also known to have logarithm of the approximate rank as a lower bound, the above theorem leaves one of two possibilities. Either F has large quantum communication complexity, and so the quantum version of the Log-Approximate-Rank Conjecture is false (previously only a quadratic separation was known), or F has small quantum communication complexity and would be the first example where quantum communication is exponentially better than classical communication. Shortly after we published our work two teams of researchers, Anshu, Boddu and Touchette [ABT19] and Sinha and de Wolf [SdW19], independently showed that F has large quantum communication complexity by giving a lower bound of $\Omega(m^{1/3})$.

The idea behind the SINK function was that we wanted a non-trivial function with small spectral norm. We noted that subcubes have small spectral norm, and by taking the union

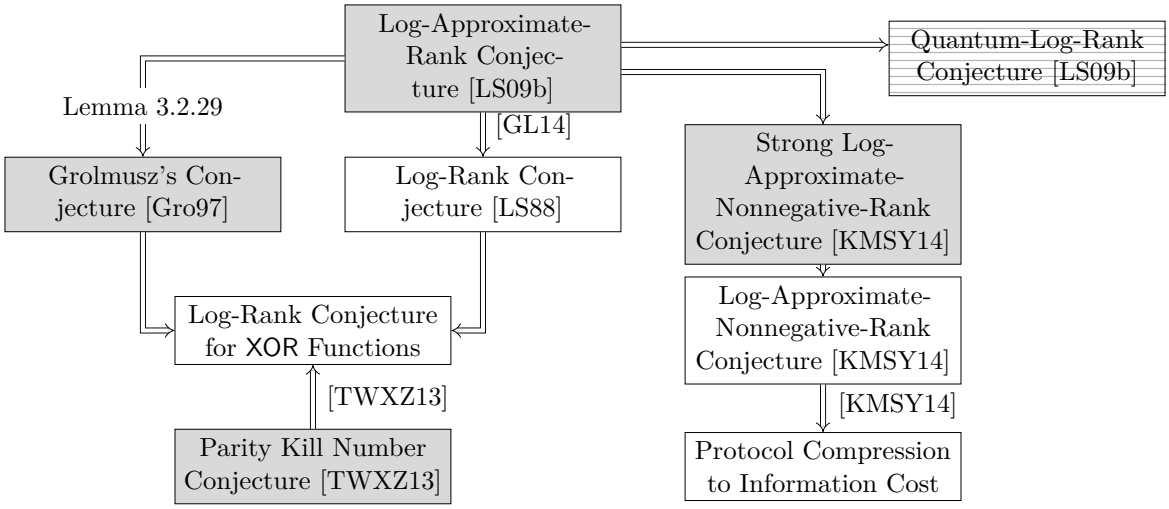


Figure 1.3: Implications between various interesting conjectures. We use the SINK function to disprove the shaded conjectures. The Quantum-Log-Rank Conjecture was subsequently falsified [ABT19, SdW19] using SINK, and the rest remain unresolved.

of disjoint subcubes we ended up with the function SINK. We note that a closely related function has been looked at before in the context of analyzing the query complexity of graph properties [LBvEB74]. The spectral norm of SINK remains small since the union is the same as the sum, and the spectral norm is a norm and hence subadditive. The logarithm of the spectral norm barely increases. It does not seem likely that the randomized communication complexity also barely increases, and that is what we prove.

1.2.3 Towards Stronger Separations: Subspace Designs

Here we detail some questions that arose from our work above, and our progress in solving them. The full details of this work are in Chapter 5.

To start with, we analyze the applicability of our methods above in order to tackle the Log-Rank Conjecture or the Log-Approximate-Nonnegative-Rank Conjecture. The function SINK was a sum of a few subcubes. Each subcube composed with XOR is easy to compute yet $\mathsf{SINK} \circ \mathsf{XOR}$ turns out to be hard for randomized communication. The Log-Approximate-Rank Conjecture was not compatible with such a phenomenon. Similarly if we found a function that is hard for deterministic communication while being a sum of functions that are deterministically easy to compute, the Log-Rank Conjecture would also be refuted. However, as we show in Theorem 5.1.1, a result of Yannakakis [Yan91] already implies that such a function is not possible.

Having ruled out such an approach against the Log-Rank Conjecture, we turn to the Log-Approximate-Nonnegative-Rank Conjecture. In this case, what would suffice to disprove the conjecture is a function f such that (1) $f^{-1}(1)$ is a disjoint union of m_1 subcubes, (2) $f^{-1}(0)$ a disjoint union of m_0 subcubes, and (3) $F := f \circ \mathsf{XOR}$ is hard for randomized communication

complexity (larger than $\mathsf{polylog}(m_0, m_1)$). However such an F is also impossible. This uses a result of Ehrenfeucht and Haussler [EH89] and we show this in Lemma 5.1.2.

However, we need not restrict ourselves to subcubes. Affine subspaces (which we refer to simply as subspaces) of $\mathbb{F}_2^n$ are also as capable as subcubes for our purposes. Can we use subspaces instead of subcubes to get better results? For instance, Lemma 5.1.2 is not yet known for subspaces. That is, if f and its complement were both disjoint unions of a small number of subspaces, does $f \circ \mathsf{XOR}$ necessarily have small randomized communication complexity?

The switch to subspaces has more potential than this. For instance, we show that it can lead to stronger counterexamples of the Log-Approximate-Rank Conjecture, i.e. it can further close the gap between approximate rank and randomized communication complexity (the gap is quartic for SINK). The search for new refutations of the Log-Approximate-Rank Conjecture is also important since any total function that shows an exponential separation of between Randomized and Quantum Communication Complexity would have to refute the Log-Approximate-Rank Conjecture.

In particular, the concept of subspace designs, previously used in coding theoretic applications by [GK16], turns out to be useful. It is a very well spread out combinatorial structure, and we use it to come up with a set of small subspaces that forms a "robust hitting set" for all large subspaces. We then show that computing whether a point is in this hitting set is hard for randomized parity decision trees, and that there are many such hitting sets for which this function has small approximate sparsity. This gives us Theorem 5.4.2 which shows that the gap between approximate sparsity and RPDT complexity can be lessened to cubic. ($O(n^3)$ approximate sparsity versus $\Omega(n)$ RPDT complexity.)

For such functions f, the function $f \circ \mathsf{XOR}$ would have $O(n^3)$ approximate rank. If a lifting theorem were to hold for the XOR gadget, this would translate to $\Omega(n)$ randomized communication complexity. Conversely, if these functions do not witness a merely cubic gap between approximate rank and randomized communication complexity (SINK witnessed a quartic gap), then there is no lifting theorem for the XOR gadget.

Furthermore, the separation of $O(\log n)$ log-approximate-sparsity versus $\Omega(n)$-randomized parity decision tree complexity is the largest possible. Any function with $o(\log n)$ log-approximate-sparsity has subpolynomial *deterministic* parity decision tree complexity.

We also provide a conjecture (Conjecture 5.5.1) that would suffice to prove the randomized communication lower bound for the class of functions we are interested in. It is an intriguing fundamental question about projections of distributions, and might be of independent interest. Go to Chapter 5 for all the details!

This wraps up our results in communication complexity. We now take a look at what the second part of this thesis deals with.

1.3 First-order Non-smooth Convex Optimization

It is not rare in the world of computation to find oneself with access to a function that one can easily compute at a point but that is too unwieldy to globally analyze. For some purposes this might not be that big a limitation. We focus on the task of finding the minimum of a convex function. To solve this one can start at any point, approximate the gradient, and step in the opposite direction in order to get closer to the minimum of the function. This is a classic algorithm known as gradient descent. It is quite natural to ask what the *best* method to get close to the minimum is (see for instance [NY83, Nes04, Bub15]). Note that if we limit ourselves to specifying points and querying the function values and gradients at the points, any algorithm is essentially a query algorithm. The 'nicer' the function is promised to be, the more efficient an algorithm one can find to accomplish the task. We will be dealing with functions that are convex but are not necessarily smooth. There may be points at which the function does not have a gradient. However since it is convex all points must have a subgradient and we can query those. Since these are first-order derivatives that we are querying this task is referred to as *first-order non-smooth convex optimization.*

To put it more formally, we have an unknown function $f : \mathbb{R}^n \to \mathbb{R}$ that is guaranteed to be convex. Let x^* be a point minimizing the value of f. The goal is to output a point x that is ϵ-optimal, that is a point such that $f(x) \leq f(x^*) + \epsilon$. The only way we can gain information about the function is through an oracle O that takes as input $x \in \mathbb{R}^n$ and outputs the value $f(x)$ and a subgradient of f at x. A subgradient is a vector $m \in \mathbb{R}^n$ such that for all $y \in \mathbb{R}^n, f(y) \geq f(x) + \langle m, y - x\rangle$. We say that such a vector m is a subgradient of f at x, or $m \in \nabla f(x)$. It is easy to see that the vector m must be equal to the gradient of f at x if the gradient exists. We wish to make as few calls to the oracle as possible.

The complexity of this task may depend on how steep this function is and on how large the region in which we want to optimize is. However, we may assume without loss of generality that the function is 1-Lipschitz (this is a bound on the steepness of the function defined as $f(x) - f(y) \leq 1 \cdot \|x - y\|$ for all x, y) and that we are optimizing within the unit ball. Why this assumption is without loss of generality is covered in more detail in Chapter 6.

Let $det, rand$ and $quantum$ refer to the deterministic, randomized and quantum query models. We define, for $\mathcal{M} \in \{det, rand, quantum\}$, the set

$$\begin{aligned} Alg_{\mathcal{M},n,\epsilon} = \{&\mathcal{A} \text{ is an } \mathcal{M} \text{ query algorithm making function value and subgradient queries } | \\ &\forall \text{ convex, 1-Lipschitz } f : \mathbb{R}^n \to \mathbb{R}, \text{ any subgradient oracle for } f, \\ &\mathcal{A} \text{ outputs an } \epsilon\text{-optimal point of } f \text{ within the unit ball}\}. \end{aligned}$$

We then define the complexity measure $C_{\mathcal{M}}(n, \epsilon) = \min_{\mathcal{A} \in Alg_{\mathcal{M},n,\epsilon}} cost(\mathcal{A})$.

There has been a lot of prior work dealing with this task [NY83, Bub15, Nes18]. Of note are two kinds of algorithms. There are dimension-dependent algorithms such as the center of

gravity method [Bub15] that shows that $C_{det}(n, \epsilon) \leq O(n \log(1/\epsilon))$, and there are dimension-independent algorithms such as projected gradient descent, proposed by Cauchy in 1847, which shows that $C_{det}(n, \epsilon) \leq O(1/\epsilon^2)$. This implies that when $n \geq \Omega(\epsilon^{-2}/\log(1/\epsilon))$, gradient descent has the better guarantee of the two algorithms. In fact it has been noted [NY83] that for $n \geq \Omega(1/\epsilon^2)$ one can show that there is *no* deterministic algorithm that can solve this task better than gradient descent (i.e. $C_{det}(n, \epsilon) \geq \Omega(1/\epsilon^2)$ when $n \geq 1/\epsilon^2$). A similar statement is also known for randomized algorithms [NY83], that there is no randomized algorithm that can perform better than gradient descent. However the question of whether there is an algorithm that outperforms gradient descent when $n \approx 1/\epsilon^2$, say, was not known (although it was known that gradient descent was still optimal upto log factors). The dimension at which gradient descent is proved to be optimal upto constants was initially a large polynomial in $1/\epsilon$ [NY83], but has been pushed down to $1/\epsilon^4$ in more recent works [WS17, BJL$^+$19].

The quantum complexity of performing machine learning tasks similar to the one above has been a hot topic in recent times [RSW$^+$19, KP20]. The power that quantum computing brings is quite apparent in applications such as computing gradients. By querying f at a superposition of points near x and then applying the Quantum Fourier Transform, a quantum algorithm is able to get an arbitrarily good approximation of the gradient of f at x with a single query. However in our case the gradient and subgradients are provided by the oracle, cancelling the advantage in that respect. The task of figuring out $C_{quantum}(n, \epsilon)$ is explicitly asked as an open problem by Chakrabarti, Childs, Wi and Lu [CCLW20]. A lower bound of $\widetilde{\Omega}(\min\{1/\epsilon, \sqrt{n}\})$ is implicit from their work.

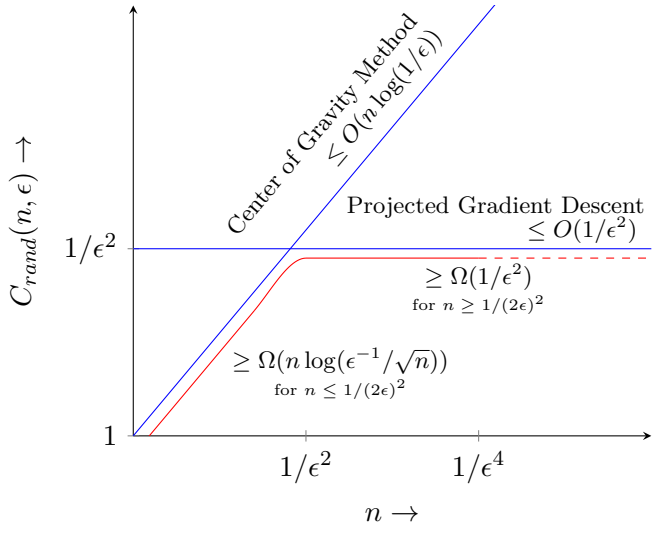
1.3.1 Our Contributions

We prove a number of lower bounds, shown in Figures 1.4a and 1.4b.

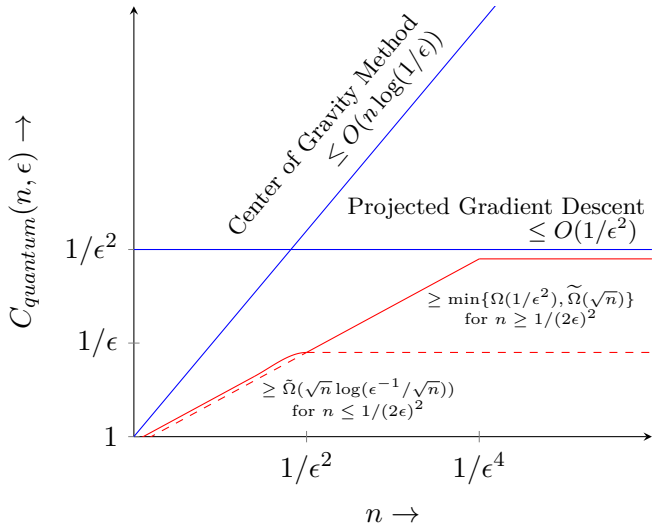
Optimality of Gradient Descent

For randomized algorithms: We show that the optimality of gradient descent with respect to randomized algorithms is true even when the dimension is as small as $O(1/\epsilon^2)$. (Theorem 6.1.3) The functions we use to show this have been used before [Nes18] to show an $\Omega(\epsilon^{-2}/\log(1/\epsilon))$ lower bound against randomized algorithms (i.e. almost as optimal as gradient descent). We do a more thorough analysis of this function and show that there is in fact a lower bound of $\Omega(1/\epsilon^2)$. To the best of our knowledge this has not been observed before in the literature. We believe our lower bound is also simpler than any previous lower bounds that proved that $\Omega(1/\epsilon^2)$ queries are necessary in large dimensions.

For quantum algorithms: We show that quantum algorithms do not exhibit a speedup. The best dimension-independent quantum algorithm we can hope for will still require $\Omega(1/\epsilon^2)$ queries. (Theorem 6.1.5) This lower bound does not use the same functions as we use in the randomized case. In fact, the quantum algorithm does get a quadratic speedup for those



(a)



(b)

Figure 1.4: The plots shown above are with an arbitrary fixed value of ϵ. The axes are plotted in log-scale and the graphs are for representational purposes. Note that $\log(\epsilon^{-1}/\sqrt{n}) = \Omega(1)$ for $n = 1/2\epsilon^2$ and $\Omega(\log(1/\epsilon))$ for $n = 1/\epsilon^{2-\Omega(1)}$. (a) In this thesis, we show the above-plotted lower bounds on the complexity of first-order convex optimization for randomized algorithms. The lower bounds shown are for the most part tight up to a constant factor. These lower bounds were known in the literature, but with a log factor loss. The dashed portion shows what lower bounds were previously known that are tight up to a constant factor: [BJL+19] showed a lower bound of $\Omega(1/\epsilon^2)$ when $n \geq \widetilde{O}(1/\epsilon^4)$. (b) In this thesis, we show the above-plotted lower bounds on the complexity of first-order convex optimization for quantum algorithms. In particular, we show that there is no dimension-independent quantum algorithm that can outperform the deterministic algorithm of Projected Gradient Descent. The dashed lines show the lower bounds prior to our work: A lower bound of $\widetilde{\Omega}(\min\{1/\epsilon, \sqrt{n}\})$ is implicit in [CCLW20].

functions. For the quantum query lower bound, we turn to a class of functions introduced by Woodworth and Srebro [WS17] in a lower bound against randomized algorithms. This class of functions was improved upon by Bubeck, Jiang, Lee, Li and Sidford [BJL+19] in a lower bound against the number of rounds needed in a "highly parallel" query algorithm. Such query algorithms allow one to make polynomially many parallel queries in a single round. Using some of their constructions and a 'hybrid argument' we are able to show a quantum lower bound. However this lower bound is on functions which have dimension $\tilde{\Theta}(1/\epsilon^4)$. It is still open whether we can show the same lower bound for functions of dimension $O(1/\epsilon^2)$ or whether quantum algorithms have an advantage in this regime.

Dependence on ϵ when the dimension is small

Convex optimization algorithms such as the center of gravity method exhibit a $\log(1/\epsilon)$ dependence on ϵ when the dimension is small. We show that even quantum algorithms cannot improve on this dependence on ϵ. We give a family of functions with dimension 1 that requires $\Omega(\log(1/\epsilon))$ queries to minimize (Corollary 6.5.7). Regarding the dependence on both n and ϵ, our dimension-independent results show that when $n \leq 1/\epsilon^4$ we have an $\tilde{\Omega}(\sqrt{n})$ lower bound (Theorem 6.5.1). We are able to say a little more, that when $n \leq 1/\epsilon^2$ we have a lower bound of $\widetilde{\Omega}(\sqrt{n}\log(\epsilon^{-1}/\sqrt{n}))$ (Theorem 6.5.15).

1.4 Organization

- Chapter 2: Models of Computation
  Here we introduce the notions of deterministic and randomized communication complexity. We then introduce query complexity with a focus on parity decision trees. Finally we introduce the basics of quantum computation and the quantum query model, and touch upon the quantum communication model.

- Chapter 3: Complexity Measures and Lower Bounds
  This chapter deals with measures that one comes across in the analysis of communication and query complexity. Many of these measures such as the approximate ranks, corruption, approximate sparsity and parity kill number are central to the subsequent chapters. Measures such as the γ_2 norms do not make an appearance in subsequent chapters but they play an important role in relating various measures to each other. *Note that this chapter has very little original content and is not to be thought of as adding to the contributions of this thesis.*

- Chapter 4: Refuting the Log-Approximate-Rank Conjecture
  Here we present the main result of this thesis, that the Log-Approximate-Rank Conjecture and a multitude of similar conjectures are false.

- Chapter 5: Extensions based on SINK: Subspace Designs
  This chapter primarily contains a class of functions that strengthens the separation between log-approximate-sparsity and randomized parity decision tree complexity from the previous chapter. It uses combinatorial objects known as subspace designs and raises interesting questions about them.

- Chapter 6: Quantum First-Order Convex Optimization
  In this chapter we analyze the quantum complexity of the task of first-order convex optimization, with the highlight being that quantum algorithms do not provide any speedup over gradient descent.

- Chapter 7: Some Open Problems
  We end the thesis by collecting a few open questions that arose from our work and a few intriguing questions that we were not able to answer.

Chapter 2

Models of Computation

In this chapter we formally define the models of computation that we will be considering: communication protocols, parity decision trees, and quantum query algorithms.

2.1 Communication Complexity

We start by formalizing a simple notion of communication. A communicating party communicates some string that depends on the data that the party possesses. This data includes the party's input and the communication already received by the party. An algorithm that multiple communicating parties can use towards solving a task is called a communication protocol. Since we only want to analyze the amount of communication that is needed, we assume that all parties have unbounded computational power. Hence the protocol does not concern itself with how a player goes about computing the string that is to be communicated. The player only needs to know what function they must use to compute the string.

In our model of communication the protocol dictates which player will communicate and what function of their data they will communicate at any given moment during the execution of the protocol. The protocol has a natural representation as a tree.

Definition 2.1.1 (Deterministic 2-Party Communication Protocol)**.** *Let Alice and Bob be two parties trying to compute a function $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$. Alice and Bob have inputs from $\mathcal{X}$ and $\mathcal{Y}$ respectively. A communication protocol Π for the two parties is a binary tree with the following properties.*

- *Every internal node v is labeled with either Alice or Bob. Each Alice node v is associated with its Alice function $f_v : \mathcal{X} \to \{0,1\}$ and each Bob node v is associated with its Bob function $f_v : \mathcal{Y} \to \{0,1\}$.*
- *Each leaf ℓ is labeled with an output $o_\ell \in \{0,1\}$.*

When Alice and Bob have inputs x and y respectively, the protocol executes as follows.

- *Starting at the root, the party whose node it is computes the associated function at that node. The output of the function is communicated to the other party. If the output is* 0*, both parties move to the left child of the node. If it is* 1*, both parties move to the right child of the node.*
- *Upon reaching a leaf ℓ, the output of the protocol is o_ℓ.*

The path that the parties take is called the transcript of the protocol and the output of the protocol is denoted $\Pi(x, y)$.

The maximum depth of a leaf (equivalently the length of the longest transcript) is the cost of the protocol, denoted $c(\Pi)$.

We say that a protocol Π computes the function F if $\forall x, y\ \Pi(x, y) = F(x, y)$.

Definition 2.1.2 (Deterministic Communication Complexity)**.** *The deterministic communication complexity of a function F, denoted $\mathsf{D}^{\mathsf{cc}}(F)$, is the minimum value, among all deterministic communication protocols Π computing F, of $c(\Pi)$.*

> Another way to view this model is that it is a decision tree where the allowed queries are arbitrary functions of Alice's input and arbitrary functions of Bob's input.

For a leaf ℓ of the protocol, let $f_1, f_2, \ldots, f_i$ be the functions associated with the Alice nodes on the path to ℓ, and $g_1, g_2, \ldots, g_k$ be the functions associated with the Bob nodes on the path to ℓ. In order to reach ℓ, let $a_1, a_2, \ldots, a_i$ be the outputs required from $f_1, \ldots, f_i$, and let $b_1, b_2, \ldots, b_j$ be the outputs required from $g_1, \ldots, g_j$. It is easy to observe the following.

Observation 2.1.3 (Leaves as Combinatorial Rectangles)**.** *Let ℓ be the leaf described above.*

- *Every input (x, y) on which the protocol reaches the leaf ℓ must satisfy $\bigwedge_{t\in[i]}(f_t(x) = a_t) \wedge \bigwedge_{t\in[j]}(g_t(y) = b_t)$.*
- *On every input (x, y) that satisfies $\bigwedge_{t\in[i]}(f_t(x) = a_t) \wedge \bigwedge_{t\in[j]}(g_t(y) = b_t)$, the protocol must reach the leaf ℓ.*

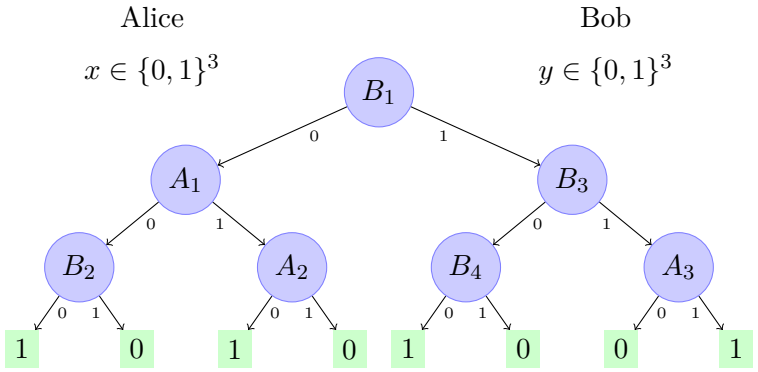
In other words, the set of inputs such that the transcript of Π ends at ℓ is exactly

$$\left\{ x \in \mathcal{X} \middle| \bigwedge_{t\in[i]} (f_t(x) = a_t) \right\} \times \left\{ y \in \mathcal{Y} \middle| \bigwedge_{t\in[j]} (g_t(y) = b_t) \right\}.$$

Sets of the form $A \times B$, where $A \subseteq \mathcal{X}$ and $B \subseteq \mathcal{Y}$, are called combinatorial rectangles. We will refer to them as rectangles.

Since every input must reach a unique leaf, it follows that the set $\mathcal{X} \times \mathcal{Y}$ is in fact partitioned into combinatorial rectangles, with one combinatorial rectangle for each leaf of the protocol.

For a function F we consider its communication matrix, denoted M_F, defined as the matrix whose rows are indexed by $\mathcal{X}$ and columns indexed by $\mathcal{Y}$, and whose (x, y)th entry is $F(x, y)$.



(a)

$y \in \{0,1\}^3$

$x \in \{0,1\}^3$	000	001	010	011	100	101	110	111
000	1	1	0	0	1	1	0	0
001	1	1	0	0	1	1	0	1
010	1	1	0	0	1	1	0	1
011	1	1	1	1	1	1	0	1
100	1	1	1	1	1	1	0	0
101	0	0	0	0	1	1	0	0
110	0	0	0	0	1	1	0	1
111	0	0	0	0	1	1	0	0

(b)

Figure 2.1:
(a) A Communication Protocol of Cost 3. The functions computed at each node are:
B_1: 1 iff $y_1 = 1$.
A_1: 1 iff $x \notin \{000, 001, 010\}$.
B_2: 1 iff $y_2 = 1$.
A_2: 1 iff $x \in \{101, 110, 111\}$.
B_3: 1 iff $y = 111$.
B_4: 1 iff $y = 110$.
A_3: 1 iff $x \in \{001, 010, 011, 110\}$.
(b) The communication matrix of the function computed by the protocol in Figure 2.1a. The inputs reaching the left child of node A_3 are highlighted.

If Π is a cost-c protocol that computes F, then the entries of M_F can be partitioned into at most 2^c *monochromatic* rectangles. See Figure 2.1b for the communication matrix of the function computed by the protocol in Figure 2.1a.

In the case of randomized protocols, there are two natural models to consider. One is that of communication with private randomness and the other with public randomness.

Definition 2.1.4 (2-Party Communication Protocol with Private Randomness)**.** *A private-randomness communication protocol* Π *for two parties Alice and Bob is specified by the following.*

- *Two distributions* $\mathcal{D}_A$ *and* $\mathcal{D}_B$ *with supports* S_A *and* S_B *respectively.*
- *A binary tree with the following properties.*
  - *Every internal node* v *is labeled with either Alice or Bob. Each Alice node* v *is associated with its Alice function* $f_v : S_A \times \mathcal{X} \to \{0,1\}$*. Each Bob node* v *is associated with its Bob function* $f_v : S_B \times \mathcal{Y} \to \{0,1\}$*.*
  - *Each leaf* ℓ *is labeled with an output* $o_\ell \in \{0,1\}$*.*

At the start of the execution of the protocol, Alice is given a sample r_A *distributed according to* $\mathcal{D}_A$ *and Bob is given an independent sample* r_B *distributed according to* $\mathcal{D}_B$*. When Alice and Bob have inputs* x *and* y *respectively, the protocol executes as follows.*

- *Starting at the root, the party whose node it is computes the associated function at that node. (If the party is Alice, it is computed with* r_A *and* x *as inputs. If Bob,* r_B *and* y*.) The output of the function is communicated to the other party. If the output is* 0*, both parties move to the left child of the node. If it is* 1*, both parties move to the right child of the node.*
- *Upon reaching a leaf* ℓ*, the output of the protocol is* o_ℓ*.*

The path that the parties take is called the transcript of the protocol and the output of the protocol is denoted $\Pi(x, y)$*. Note that both the path and the output are random variables.*

The maximum depth of a leaf (equivalently the length of the longest transcript) is the cost of the protocol, denoted $c(\Pi)$*.*

We say that a protocol Π computes the function F to within error ϵ if $\forall x, y \ \Pr[\Pi(x, y) = F(x, y)] \geq 1 - \epsilon$.

Definition 2.1.5 (Private-Randomness ϵ-error Randomized Communication Complexity)**.** *The private-randomness* ϵ*-error randomized communication complexity of a function* F*, denoted* $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},\epsilon}(F)$*, is the minimum value, among all private-randomness communication protocols* Π *computing* F *to within error* ϵ*, of* $c(\Pi)$*.*

We may sometimes omit ϵ or just say bounded-error instead of ϵ-error. In both these cases, we assume ϵ is $1/3$.

For a leaf ℓ of the protocol, let $f_1, f_2, \ldots, f_i$ be the functions associated with the Alice nodes on the path to ℓ, and $g_1, g_2, \ldots, g_k$ be the functions associated with the Bob nodes on the path to ℓ. In order to reach ℓ, let $a_1, a_2, \ldots, a_i$ be the outputs required from $f_1, \ldots, f_i$, and let $b_1, b_2, \ldots, b_j$ be the outputs required from $g_1, \ldots, g_j$. It is easy to observe the following.

Observation 2.1.6 (Leaves as Rank-1 Matrices)**.** *Let ℓ be the leaf described above. We can then characterize the probability that an input (x, y) reaches the leaf ℓ.*

$$\Pr[(x,y) \text{ reaches } \ell] = \Pr_{r_A \sim \mathcal{D}_A, r_B \sim \mathcal{D}_B} \left[\bigwedge_{t \in [i]} (f_t(r_A, x) = a_t) \wedge \bigwedge_{t \in [j]} (g_t(r_B, y) = b_t) \right].$$

Since r_A and r_B are sampled independently, this simplifies to

$$\Pr_{r_B \sim \mathcal{D}_B} \left[\bigwedge_{t \in [i]} (f_t(r_A, x) = a_t) \right] \cdot \Pr_{r_B \sim \mathcal{D}_B} \left[\bigwedge_{t \in [j]} (g_t(r_B, y) = b_t) \right].$$

In other words, let M_ℓ be the matrix that specifies, for a row indexed by x and column indexed by y, the probability that (x, y) reaches ℓ. Then M_ℓ is a rank-1 matrix since the (x, y)th entry is of the form $p_x \cdot q_y$.

We now turn to the model of public-randomness communication protocols.

Definition 2.1.7 (2-Party Communication Protocol with Public Randomness)**.** *A public-randomness communication protocol Π for two parties Alice and Bob is a distribution over deterministic communication protocols.*

The protocol executes as follows when Alice and Bob are given inputs x and y. A deterministic communication protocol Π' is sampled from Π. The protocol Π' is then executed on input (x, y). The output of Π is the output of Π'. The output of Π is a random variable.

The cost of the protocol, denoted $c(\Pi)$, is the maximum of the costs of the deterministic protocols in the support of Π.

Definition 2.1.8 (Public-Randomness ϵ-error Randomized Communication Complexity)**.** *The public-randomness ϵ-error randomized communication complexity of a function F, denoted $\mathsf{R}^{\mathsf{cc}}_\epsilon(F)$, is the minimum value, among all public-randomness communication protocols Π computing F to within error ϵ, of $c(\Pi)$.*

2.2 Parity Decision Tree Complexity

The parity decision tree complexity of a function f measures the number of parities of the input one needs to query in order to compute f on the input. As usual, the query algorithm is best represented as a tree.

Definition 2.2.1 (Parity Decision Trees)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a function we want to compute. A parity decision tree T is a binary tree in which every internal node is labeled with a set $S \subseteq [n]$. The leaves are labeled with outputs from $\{0,1\}$.*

On an input x, the execution of the query algorithm represented by T proceeds as follows. We start with the current node being the root.

- *Let S be the set associated with the current node. Compute $\oplus_{i \in S} x_i$. If it is 0, go to the left child. If it is 1, go to the right child.*
- *When you reach a leaf, output the label of the leaf.*

The output of the parity decision tree is denoted $T(x)$. The cost of the tree, $c(T)$, is the maximum depth of a leaf in the tree.

We say that T computes a function f if for all $x \in \{0,1\}^n, T(x) = f(x)$.

To analyze the mathematical structures that arise from parity decision trees, let us view the input as coming from the vector space $\mathbb{F}_2^n$ instead of thinking of it as $\{0,1\}^n$. We will refer to the $\mathbb{F}_2$-sum as $\oplus$.

For a leaf ℓ of a parity decision tree, let $S_1, S_2, \ldots, S_i$ be the sets associated with the nodes on the path to ℓ, and let $a_1, a_2, \ldots, a_i$ be the outputs required from $\oplus_{S_1}, \ldots, \oplus_{S_i}$ to reach ℓ. It is easy to make the following observation.

Observation 2.2.2 (Leaves as Affine Subspaces)**.** *Let ℓ be the leaf described above.*

- *Every input x on which the tree reaches the leaf ℓ must satisfy $\bigwedge_{t \in [i]} (\oplus_{j \in S_t} x_j = a_t)$.*
- *On every input x that satisfies $\bigwedge_{t \in [i]} (\oplus_{j \in S_t} x_j = a_t)$, the tree must reach the leaf ℓ.*

In other words, the set of inputs that reach ℓ is exactly the $\mathbb{F}_2$ affine subspace specified by the linear equations

$$\forall t \in [i], \quad \oplus_{j \in S_t} x_j = a_t.$$

> To get more comfortable with this model, let us see what happens when we make a query. Suppose the query at the root was $x_1 \oplus x_2$ and the query returned 1. The subfunction that we are now left to compute is no longer a total function with domain $\{0,1\}^n$. However, it is still a total function with its domain being a subspace of $\mathbb{F}_2^n$ of dimension $n-1$, and is equivalent to a total function on $n-1$ bits.

Definition 2.2.3 (Randomized Parity Decision Tree)**.** *A randomized parity decision tree T is a distribution over deterministic parity decision trees.*

The tree executes as follows when given input x. A deterministic tree T' is sampled from T. The protocol T' is then executed on input x. The output of T is the output of T'. The output of T is a random variable.

The cost of the protocol, denoted $c(T)$, is the maximum of the costs of the deterministic trees in the support of T.

We say that a randomized parity decision tree T computes the function f to within error probability ϵ if $\forall x \ \Pr[T(x) = f(x)] \geq 1 - \epsilon$.

Definition 2.2.4 (ϵ-error Randomized Parity Decision Tree Complexity)**.** *The ϵ-error randomized communication complexity of a function f, denoted $\mathsf{R}^{\oplus}_{\epsilon}(f)$, is the minimum value, among all randomized parity decision trees T computing f to within error ϵ, of $c(T)$.*

2.3 Quantum Query and Communication Models

We now introduce the main concepts behind quantum computation. A more thorough introduction can be found in [NC16, Wol19]. Quantum data are stored in registers. A register is associated with a number of dimensions, which is also the number of possible outputs we can get from the register. A qubit is a register of dimension 2. The datum in a register is called the quantum state of the register. The state of the register is either pure, or it is entangled with the states of other registers. In what follows $[d]$ represents the set $\{0, 1, \ldots, d-1\}$.

A pure state: Description and notation

A pure state in a d-dimensional register is represented by a unit vector in $\mathbb{C}^d$. If the register is called R, and the vector v, this state is written as $|v\rangle_R$. There are d canonical mutually orthogonal unit vectors, or "axis" vectors, that form a basis for the space $\mathbb{C}^d$. These are usually referred to as $e_1, e_2, \cdots, e_d$. We use $|0\rangle_R, |1\rangle_R, \ldots, |d-1\rangle_R$ to refer to these states. The vector $(1/\sqrt{2}, 1/\sqrt{2}) \in \mathbb{C}^2$ is represented as $\frac{|0\rangle_R + |1\rangle_R}{\sqrt{2}}$, or $\left(\frac{|0\rangle+|1\rangle}{\sqrt{2}}\right)_R$. It is a superposition of the states $|0\rangle_R$ and $|1\rangle_R$. This is more often written as 'R is in the state $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$', since the register is implicit.

Looking at your quantum data: Measurement

So you have been given a d-dimensional register R which is in some state, say $|v\rangle_R$. What do you see when you peek into the register? Let $|v\rangle_R = \sum_{i\in[d]} \alpha_i |i\rangle_R$. With probability $|\alpha_i|^2$, the register will change its state to $|i\rangle_R$, and i is the output that we will get from the measurement. (We know that since v is a unit vector, $\sum_{i\in[d]} |\alpha_i|^2 = 1$.)

Adding more data: Multiple registers

Consider the scenario where you have two registers: a d_1-dimensional register R_1 and a d_2-dimensional register R_2. We can think of the state of these two registers combined, yielding a register R of dimension $d = d_1 d_2$. The canonical basis states that R can be in are, for each $i_1 \in [d_1], i_2 \in [d_2]$, the state where R_1 is in state $|i_1\rangle_{R_1}$ and R_2 is in state $|i_2\rangle_{R_2}$. In general, if R_1 is in a pure state $|v_1\rangle_{R_1}$ and R_2 is in a pure state $|v_2\rangle_{R_2}$, then the state of R is $|v_1 \otimes v_2\rangle_R$. This is also written as $|v_1\rangle_{R_1}|v_2\rangle_{R_2}$.

Hence we can form a set of basis states of R as $\{|i\rangle_{R_1}|j\rangle_{R_2}\}_{i\in[d_1], j\in[d_2]}$. It follows that every pure state of R is of the form $\sum \alpha_{i,j}|i\rangle_{R_1}|j\rangle_{R_2}$. It is also common to write this as $\sum \alpha_{i,j}|i\rangle|j\rangle$ or even $\sum \alpha_{ij}|ij\rangle$, where it is implicit that the registers are R_1 and R_2, with the R_1 component being written before the R_2 component. States of the form $|v_1\rangle_{R_1}|v_2\rangle_{R_2}$ are

called separable, since they can be separated into two registers holding pure states. However, the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is not a separable state. In this state, the registers R_1 and R_2 do not hold pure states and are instead *entangled* with each other which, as we will see shortly, can lead to weird phenomena when they are measured. This weirdness was famously elaborated on by Einstein, Podolsky and Rosen [EPR35] and the above state is referred to as an EPR pair.

Manipulating your quantum data

The way you change the state of your register is really simple: rotations. Rotations in a d-dimensional space are easy to characterise. Choose d mutually orthogonal unit vectors $b_0, b_1, \ldots, b_{d-1}$. Each choice corresponds to a different rotation. The way your state changes is as follows.

- For each $i \in [d]$, $|i\rangle$ changes to $|b_i\rangle$.
- For a vector $v = \sum_{i\in[d]} \alpha_i |i\rangle$, $|v\rangle$ changes to $\sum_{i\in[d]} \alpha_i |b_i\rangle$.

Mathematically all this is simply characterized by unitary transformations. Let U be a $d \times d$ unitary matrix. U maps the state v to the state Uv.

Consider the register R considered previously (made up of registers R_1 and R_2). We can choose our unitary so that we can transform $|00\rangle$ to any state $|v\rangle_R$ of our choosing, including the EPR pair.

Measuring and manipulating an entangled register

Suppose we had access only to register R_1 from the above example. Even though it is entangled with R_2 which we do not have access to, we can still measure and apply unitaries to R_1. Since register R contains a pure state, i.e. a unit vector in $\mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2}$, it can be written as $v = \sum_{i\in[d_1], j\in[d_2]} \alpha_{i,j}|ij\rangle = \sum_{i\in[d_1]} |i\rangle|v_i\rangle$ where each v_i is a vector (not necessarily a unit vector) in $\mathbb{C}^{d_2}$. Upon measuring R_1, the state of R collapses to $|i\rangle|v_i'\rangle$, where v_i' is v_i normalized, with probability $||v_i||^2$.

Applying a d_1-dimensional unitary U to R_1 no longer makes sense as previously described because the state of R_1 is no longer a d_1-dimensional vector. Such an operation can nevertheless be performed, and the way to interpret this operation is as applying the unitary $U \otimes I$ to the state of R, where I is the d_2-dimensional identity matrix. The result of doing this operation to the state $\sum_{i\in[d_1]} |i\rangle|v_i\rangle$ is the state $\sum_{i\in[d_1]} (U|i\rangle)|v_i\rangle$.

Let us now use the model above to work out one of the strangenesses that Einstein, Podolsky and Rosen had noted. Let R_1 and R_2 be qubits that are in the entangled state $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$. If we were to measure the qubit R_1, we would get $|0\rangle$ with probability $\frac{1}{2}$ and $|1\rangle$ with probability $\frac{1}{2}$. In the former case R_2 would collapse to $|0\rangle$ and in the latter case R_2 would collapse to $|1\rangle$. Now instead, let us apply the Hadamard unitary to R_1. The Hadamard unitary is defined as

$$|0\rangle \mapsto \frac{|0\rangle + |1\rangle}{\sqrt{2}}, |1\rangle \mapsto \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

This transforms our state to

$$\frac{(|0\rangle + |1\rangle)|0\rangle + (|0\rangle - |1\rangle)|1\rangle}{2} = \frac{|0\rangle(|0\rangle + |1\rangle) + |1\rangle(|0\rangle - |1\rangle)}{2}.$$

Now upon measuring R_1, we would still get $|0\rangle$ with probability $\frac{1}{2}$ and $|1\rangle$ with probability $\frac{1}{2}$. However in the former case R_2 would collapse to $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$, and in the latter case R_2 would collapse to $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$. Recall that if we had not applied the Hadamard unitary, R_2 would have collapsed to either $|0\rangle$ or $|1\rangle$. That we could influence the state of R_2 in such a manner by performing operations on R_1 even if R_1 and R_2 were astronomically far apart was what unsettled Einstein, Podolsky and Rosen.

What does quantum computation look like?

A typical scenario for a quantum computational task would be something like the following. You want to compute a function $f : [2^n] \to \{0, 1\}$. You are given a qubit *ans*. You are given a 2^n-dimensional register *input*. You are also allowed to use an auxiliary register *aux* of a dimension of your choosing, say k. The initial state is $|0\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ where $i \in [2^n]$. You want to come up with a unitary U of dimension $2 \cdot 2^n \cdot k$ such that for all $i \in [2^n]$, $U|0\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ is a state which, on measuring the *ans* register, returns $f(i)$ with high probability.

Note that it is easy to come up with such a unitary. One could choose any unitary that maps $|0\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ to $|0\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ for all $i \in f^{-1}(0)$ and $|0\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ to $|1\rangle_{ans}|i\rangle_{input}|0\rangle_{aux}$ for all other is. However, being able to use arbitrary unitaries is akin to allowing a circuit model where each gate computes an arbitrary function. Circuits built with just two bit NAND gates are more representative of computation.

Similarly here, we would want to only use unitaries and registers that are easy to handle. For this purpose, we restrict ourselves to using qubits (2-dimensional registers) and to using unitaries that act on at most 2 qubits at a time. For a register made up of 3 qubits, we concisely denote $|0\rangle|1\rangle|0\rangle$ as $|010\rangle$. Similarly n-bit strings index the basis vectors of a register made up of n qubits. The operations allowed are

- For any qubit, one can apply any unitary transformation.
- For any two qubits, one can apply the CNOT gate, defined as

$$|00\rangle \mapsto |00\rangle, |01\rangle \mapsto |01\rangle, |10\rangle \mapsto |11\rangle, |11\rangle \mapsto |10\rangle.$$

The above transformations are complete, in the sense that any unitary can be obtained by composing the above unitaries. Quantum circuit complexity asks for the minimum k such that by composing k of the above unitaries, one gets a unitary U that solves f in the sense mentioned a few paragraphs above. Note that the gate set above is uncountably large and hence impractical. We instead satisfy ourselves with being able to obtain arbitrarily

good approximations to unitaries, and it is possible to do so with just the following two operations [Kit97]!

- For any qubit, one can apply the Hadamard gate.
- For any two qubits, one can apply the C-P(i) gate, defined as

$$|00\rangle \mapsto |00\rangle, |01\rangle \mapsto |01\rangle, |10\rangle \mapsto |10\rangle, |11\rangle \mapsto i|11\rangle.$$

However we shall not concern ourselves with these details as we will be dealing with query algorithms and communication protocols, both of which allow unbounded computation. Even with these details put aside, the model of quantum computation does seem far removed from other models. It may even seem a bit finicky at times, caring about *how* you did a certain computation. Was the answer register entangled with the ancillary registers? Perhaps your algorithm does place, for each input, the correct answers in the answer register, but are the coefficients of each of the answer states the same? These questions do come up and are important in order to make sure that different algorithms play together well and can be used as subroutines.

But luckily these issues are not brought up in our use cases. If we only consider quantum computations that are computing classical functions, we do not need to care about *how* the answer is being computed. This statement is encapsulated by the statement $\mathsf{BQP}^{\mathsf{BQP}} = \mathsf{BQP}$ and is referred to as the BQP Subroutine Theorem [BBBV97]. They show that one can run a subroutine that computes a *function*, perform error reduction, copy the output of the subroutine (which is almost a basis state thanks to the error reduction), and then 'reverse the computation'. If we had not copied the output, we would end up with the initial state since we reversed the computation. If the output that was copied was a basis state, the output register would not be entangled with the other qubits and we would end up with the output in the output register and the rest of the qubits would be as they were before the computation started. Note that this is the ideal situation since it is exactly how an oracle would work. In our case where the output is nearly a basis state, we end up with a quantum state that is nearly the ideal state, which is good enough. Here too error reduction involves running the algorithm 'many times' and taking the majority ('many times' being logarithmic in the error that we want to amplify it to).

2.3.1 Quantum Query Algorithms

This probably goes without saying, but quantum query algorithms are to quantum algorithms as classical query algorithms are to classical algorithms. In a classical query algorithm, arbitrary operations are allowed between queries. Let us assume that to make a query the algorithm writes down the index to be queried at a specific location, and a query oracle returns the result

of the query at a specific location. All the oracle does is it houses a string $x \in \Sigma^n$ and when invoked it returns the value x_i.

In the case of a quantum query algorithm, we would assume we have an n-dimensional register *in* and a $|\Sigma|$-dimensional register *out*. But for simplicity, and so that it has a better interplay with the rest of the registers, we take *in* to be a register made up of $\lceil \log n \rceil$ qubits, and *out* to be a register made up of $\lceil \log |\Sigma| \rceil$ qubits. We require that the oracle also be a unitary so that it makes quantum mechanical sense. The behaviour of the oracle unitary O_x can be specified as follows on the registers *in* and *out*.

$$\forall i \in \{0,1\}^{\lceil \log n \rceil}, a \in \{0,1\}^{\lceil \log |\Sigma| \rceil} \qquad O_x|i\rangle_{in}|a\rangle_{out} = |i\rangle_{in}|x_i \oplus a\rangle_{out},$$

where $\oplus$ is the bitwise XOR and x_i is the $\lceil \log |\Sigma| \rceil$-bit string representing the actual value in Σ for $i \in [n]$ and is the all-0 string otherwise. Since each basis state is mapped to a unique basis state, this map is unitary.

Between the queries, the quantum query algorithm can do any operation, and this can be encapsulated by a single unitary. Hence every q-query quantum algorithm can be represented as a sequence of unitaries acting on registers *in*, *out* and *anc* (for ancillary). The algorithm chooses how many qubits *anc* is made up of beforehand. It starts in the state $|s_0\rangle = |\overline{0}\rangle_{in}|\overline{0}\rangle_{out}|\overline{0}\rangle_{anc}$. It then applies the unitaries U_0, O_x, U_1, O_x, $\cdots$, U_{q-1}, O_x, U_q. The state after k queries is $|s_k\rangle = O_x U_{k-1}|s_{k-1}\rangle$. The final state is $|s_{final}\rangle = U_q|s_q\rangle$. Certain qubits (specified by the quantum query algorithm) of the state $|s_{final}\rangle$ are then measured to get the output of the quantum query algorithm.

Let us now play with this to make some simple statements about learning bitstrings with quantum bit queries.

Lemma 2.3.1. *There is no* 1*-query quantum algorithm that makes queries to the bits of the string* x_1x_2 *and learns* x_1x_2 *perfectly.*

Proof. Any 1-query quantum algorithm must be of the form $U_1O_xU_0|0\rangle$. Let the state $U_0|0\rangle = \alpha_1|1\rangle_{in}|\phi_1\rangle + \alpha_2|2\rangle_{in}|\phi_2\rangle$ where ϕ_1, ϕ_2 are unit vectors (hence $|\alpha_1|^2 + |\alpha_2|^2 = 1$). Let us denote by $|\phi_{00}\rangle, |\phi_{01}\rangle, |\phi_{10}\rangle$ and $|\phi_{11}\rangle$ the states after the oracle call (i.e., $O_xU_0|0\rangle$) depending on the value of x_1x_2. In order to perfectly output x_1x_2, $U_1|\phi_{x_1x_2}\rangle$ must have $|x_1x_2\rangle$ in the output register, implying that each of the four states must be mutually orthogonal. Hence $|\phi_{x_1x_2}\rangle$ themselves must be mutually orthogonal.

Note that $O_x|1\rangle_{in}|\psi\rangle = O_y|1\rangle_{in}|\psi\rangle$ if $x_1 = y_1$ regardless of ψ. Let c_1 be the value of $\langle O_x|1\rangle_{in}|\phi_1\rangle, O_y|1\rangle_{in}|\phi_1\rangle\rangle$ when $x_1 = 0$ and $y_1 = 1$. Similarly define c_2 to be the value of $\langle O_x|2\rangle_{in}|\phi_2\rangle, O_y|2\rangle_{in}|\phi_2\rangle\rangle$ when $x_2 = 0$ and $y_2 = 1$. Note that c_1 and c_2 have magnitude at most 1.

$\langle\phi_{00}|\phi_{01}\rangle = |\alpha_1|^2 + |\alpha_2|^2 c_2$, which has norm at least $|\alpha_1|^2 - |\alpha_2|^2$. Similarly $\langle\phi_{00}|\phi_{10}\rangle$ has norm at least $-|\alpha_1|^2 + |\alpha_2|^2 + |\alpha_3|^2$. For these to both be 0, $|\alpha_1|^2 = |\alpha_2|^2$ and $c_1 = c_2 = -1$.

Finally, $\langle\phi_{10}|\phi_{01}\rangle = |\alpha_1|^2 c_1 + |\alpha_2|^2 c_2$, which will be a real number smaller than 0. Hence they cannot all be mutually orthogonal.

> The lesson here is not that quantum queries are useless. Continuing this analysis would have shown us that there is a 1-query algorithm that computes $x_1 \oplus x_2$.
> The above argument is also robust. We can show that the states can't even be mutually nearly orthogonal, hence giving us a 2-query lower bound even on learning $x_1 x_2$ with worst-case error probability 0.01, say.

□

Quantum query complexity has been characterized with various measures, and these characterizations have led to strong results, such as this strong direct product theorem by Lee and Roland.

Theorem 2.3.2 (Theorems 3.3 and 4.2, [LR13])**.** *Let $\epsilon > 0$ be a constant. For any function f, any error parameter $2/3 \leq \delta \leq 1$, and any integer $k > 0$, we have*

$$Q_{1-\delta^{k/2}}(f^{(k)}) \geq \Omega(k \ln(3\delta/2) Q_\epsilon(f))$$

where $f^{(k)}$ is k independent copies of the function f and $Q_\epsilon(f)$ is the minimum number of queries required for a quantum query algorithm computing f with worst-case error at most ϵ.

We can now use this theorem in order to prove a statement about learning bitstrings with even a small probability of success. This will be useful for us in Section 6.5.

Lemma 2.3.3. *Let $\mathcal{A}$ be an m-query quantum query algorithm making bit queries to a bitstring $z \in \{0,1\}^n$ satisfying the following condition.*

$$\mathop{\mathbb{E}}_{z\in\{0,1\}^n}[\mathcal{A} \text{ outputs } z] \geq 0.9^{-n/4}.$$

Then $m \geq \Omega(n)$.

Proof. We start by using the strong direct product theorem. Learning an n-bit string is $n/2$ copies of learning a 2-bit string. Setting $\delta = 0.9$, the strong direct product theorem tells us that if there is an m-query algorithm $\mathcal{A}$ such that for all $z \in \{0,1\}^n$, $\Pr[\mathcal{A} \text{ outputs } z] \geq 0.9^{-n/4}$, then $m \geq \Omega(n)$.

We now use a simple argument to show that if there is an m-query algorithm $\mathcal{A}$ satisfying $\mathbb{E}_{z\in\{0,1\}^n}[\mathcal{A} \text{ outputs } z] \geq 0.9^{-n/4}$, then there is an m-query algorithm $\mathcal{A}'$ such that for all $z \in \{0,1\}^n$, $\Pr[\mathcal{A} \text{ outputs } z] \geq 0.9^{-n/4}$. The algorithm $\mathcal{A}'$ proceeds as follows. Sample uniformly at random an n-bit string r. Insert the unitary $|i\rangle_{in}|b\rangle_{out} \mapsto |i\rangle_{in}|b \oplus r_i\rangle_{out}$ after each oracle call of $\mathcal{A}$. Now running $\mathcal{A}'$ on input z will exactly mimic running $\mathcal{A}$ on input $z \oplus r$. $\mathcal{A}'$ then takes the output of $\mathcal{A}$ and outputs its bitwise xor with r. By assumption, $\mathcal{A}$ would have output $z \oplus r$ with probability at least $0.9^{-n/4}$ and hence $\mathcal{A}'$ would output z with probability at least $0.9^{-n/4}$. □

2.3.2 Quantum Communication Complexity

> This subsection can be skipped by most readers, it is mostly technical details that will only be relevant to Section 3.6, which is not really central to this thesis.

A quantum communication protocol looks like the following. Alice and Bob are given x and y respectively and are trying to compute $f(x, y)$ where $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Alice has a register A of a dimension of her choosing, Bob has a register B of a dimension of his choosing, and Alice has a qubit *comm*.

Looking at her input, Alice chooses unitaries $U_0, U_2, \cdots, U_{2k}$ that act on the register made of A and *comm*. Looking at his input, Bob chooses unitaries $U_1, U_3, \cdots, U_{2k+1}$ that act on the register made of B and *comm*. Alice proceeds to apply U_0, send *comm* to Bob, who applies U_1 and sends *comm* back to Alice and so on.

After applying the unitaries, they get

$$(I_A \otimes U_{2k+1})(U_{2k} \otimes I_B) \dots (I_A \otimes U_1)(U_0 \otimes I_B)|\overline{0}\rangle_A|0\rangle_{comm}|\overline{0}\rangle_B.$$

They measure *comm* and get the output of the protocol. The cost of the protocol is $2k + 1$.

Note that Alice's unitaries could depend on the input x, and similarly Bob's unitaries could depend on the input y. However this is not necessary in the definition of a protocol. Alice could have been given x in an n-qubit register A_{in}, and Bob could have been given y in an n-qubit register B_{in}. Then Alice could have chosen a universal unitary acting on A_{in}, A and *comm* that looks at the value in A_{in} and based on that performs a unitary on A and *comm*.

Simulating a classical protocol

The protocol described above could be called a deterministic protocol, since given inputs x and y its final state is determined. However, the measurement at the end gives the output as a random variable. This is akin to how in a randomized protocol, given x and y a distribution on transcripts is determined. We can then sample from the distribution to get the output as a random variable. Indeed we describe a quantum protocol that simulates a cost-c private coin randomized protocol (assuming a finite sample space for the randomized protocol) in which the players alternate in sending messages of length 1 bit and the last bit of the transcript is the output. We also assume all leaves are at depth c, we insert dummy communication nodes to make sure of this.

> We can always assume that the sample space is finite. It is not hard to see that any private coin randomized protocol can be simulated by a protocol with the following condition: A player can toss at most one coin every time they need to send a bit. The bias of this coin may be hard to compute, but the players are computationally unbounded.

Let Alice's private randomness be a sample from a distribution $\mathcal{D}_A$ and Bob's be from a distribution $\mathcal{D}_B$. Alice has a register A_{rand} initialized to $\sum_{r \in supp(\mathcal{D}_A)} \sqrt{\mathcal{D}_A(r)}|r\rangle$, and similarly Bob has his register.

Alice has a c-qubit register $A_{transcript}$ initialized to $|\overline{0}\rangle$ and Bob has a similar register.

The classical protocol is simulated as follows. Let the transcript of the protocol, conditioned on $r \sim \mathcal{D}_A, r' \sim \mathcal{D}_B$ be $t(x, y, r, r')$. The state that the quantum protocol will have after simulating i bits of classical communication will be

$$|x\rangle_{A_{in}}|y\rangle_{B_{in}} \sum_{r \in supp(\mathcal{D}_A)} \sum_{r' \in supp(\mathcal{D}_B)} \Big(\sqrt{\mathcal{D}_A(r)}|r\rangle_{A_{rand}} \sqrt{\mathcal{D}_B(r')}|r'\rangle_{B_{rand}}$$
$$|t(x,y,r,r')_{\leq i} 0^{c-i}\rangle_{A_{transcipt}} |t(x,y,r,r')_{\leq i} 0^{c-i}\rangle_{B_{transcipt}}\Big)|0\rangle_{comm}.$$

We now see how to simulate the next bit of the protocol. Assume it is Alice's turn to send a bit. Alice's unitary will operate on the first i qubits of $A_{transcript}$, on the registers A_{in}, A_{rand} and on the register *comm* in order to set the state of *comm* to be the bit that Alice would have sent. This state of *comm* is also copied, via a C-NOT gate, to the $i+1$th qubit of $A_{transcript}$. Alice then sends *comm* to Bob, who swaps *comm* and the $i+1$th qubit of $B_{transcript}$. At this point the state of the quantum protocol is as it should be after simulating $i+1$ bits of communication.

For the last bit of the simulation, *comm* is not swapped with the transcript register. *comm* contains the kth qubit of $A_{transcript}$. At this point the state is $|0\rangle_{comm}|v_0\rangle_{the\ rest} + |1\rangle_{comm}|v_1\rangle_{the\ rest}$, where the square of the length of v_0 is

$$\sum_{r \in supp(\mathcal{D}_A), r' \in supp(\mathcal{D}_B) \text{ such that } t(x,y,r,r')_c = 0} \mathcal{D}_A(r)\mathcal{D}_B(r').$$

Thus when measuring *comm*, $|0\rangle$ is measured with the probability that the classical protocol outputs 0.

Allowing pre-entangled qubits

Public coin protocols cannot be simulated this way. However, if Alice and Bob share a $k+k$ qubit register set to $\frac{1}{2^{k/2}} \sum_{r \in \{0,1\}^k} |r\rangle|r\rangle$ with the first k qubits with Alice and the latter k with Bob, then they can measure these registers and get k bits uniformly at random, with the guarantee that Alice's k bits are the same as Bob's k bits. In this model, public coin protocols can also be easily simulated. However, the power of entanglement may be more than just sharing public coin tosses. It is still open whether one can achieve a massive gain in efficiency by having pre-shared entangled registers.

Chapter 3

Complexity Measures and Lower Bounds

In the previous chapter we looked at some models of computation. A function that is easy to compute in any of those models would have a concise representation, and hence would take small values when analyzed under a variety of mathematical measures.

In this chapter we look at some such measures and what we know about them. These measures roughly fall into two categories, which we'll informally think of as counting measures and weighing measures. An example of a counting measure is the Fourier sparsity of a function which is a count of how many non-zero Fourier coefficients it has. An example of a weighing measure is the Fourier ℓ_1 norm of a function, which is the sum of the absolute values of its Fourier coefficients.

★ We now give a list of the topics covered in this chapter, highlighting with a ★ some observations/expositions that we do not believe are found readily in the literature. When you reach such an observation/exposition in the chapter the same symbol appears in the margin, as it does next to this paragraph.

- Measures pertaining to Deterministic Communication
  - Rectangle Partition Number
  - Rank
  - Nonnegative Rank
  - γ_2 Norm
- Measures pertaining to Randomized Communication
  - Approximate Rank
  - Approximate Nonnegative Rank
  - Sign Rank

  - Partition Bound
  - Smooth Rectangle Bound
  - Corruption Bound
  - Approximate γ_2: γ_2^α
  - γ_2^∞
    - ★ Equivalence between γ_2^∞ and weakly unbounded error communication complexity
  - Grolmusz's Conjecture

- Measures pertaining to Parity Decision Trees
  - Large Affine Space Partition Number
  - Sparsity and Spectral Norm
  - Parity Leaf Complexity
  - Parity Kill Number

- Measures pertaining to Randomized Parity Decision Trees
  - Approximate Sparsity and Approximate Spectral Norm
  - Polynomial Margin

★ A Bridge Between Counting and Weighing
  - Grolmusz's Theorem: Approximate Sparsity vs. Approximate Spectral Norm
  - Newman's Theorem
  - Approximate Rank vs. Approximate γ_2
  - Approximate Nonnegative Rank vs. Smooth Rectangle Bound

- Lifting from PDT Measures to Communication Measures
  - Sparsity to Rank, Spectral Norm to γ_2
  - ★ Margin to γ_2^∞
  - ★ Approximate Sparsity to Approximate Rank

★ Quantum Communication is Upper Bounded by γ_2^∞

- Figure incorporating many of the relations seen in this chapter

We now give a brief description of the starred items.

- The equivalence between γ_2^∞ and weakly unbounded error communication complexity has long been known [LS09b]. However, this equivalence goes via linear programming duality through another quantity called discrepancy. Here we give a much simpler proof of the equivalence which does not involve any duality.
- The results given in the Bridge Between Counting and Weighing section are also known. We observe that all the proofs in the literature follow from similar ideas that can be unified under a single framework. We attempt to present them from the viewpoint of this framework.
- Lifting approximate sparsity to approximate rank is a simple consequence of previous observations. It is nevertheless an interesting statement and we have not seen it stated before. It is still open whether it can be made tighter.
- That margin lifts to γ_2^∞ up to a multiplicative constant is implicit in [CM17]. However this goes through LP duality and discrepancy. Here we note that as a simple corollary of known theorems one can see that margin lifts *exactly* to γ_2^∞.
- Quantum communication is known to be at most the square root of the approximate rank [GS19]. This proof goes via approximate γ_2. We note that essentially the same proof can be extended further to show that quantum communication is upper bounded by γ_2^∞.

> This remark is an aside, used to give helpful or interesting facts that are relevant but perhaps not necessary. The proofs in this chapter are enclosed in boxes to allow the reader to scan the chapter with more ease.

3.1 Preliminaries

Before we get started with introducing measures and analyzing them, let us first build a useful set of tools that we will use in our analysis.

Hoeffding's lemma is a concentration bound that is useful when we want to reason about efficient approximations.

Lemma 3.1.1 (Hoeffding's Inequality [Hoe63])**.** *Let $X_i \in [a_i, b_i]$ for $i = 1, 2, \ldots, n$ be independent random variables and $X = \sum_i X_i$. Then*

$$\Pr[|X - \mathbb{E}[X]| \geq t] < 2 \exp\left(-\frac{2t^2}{\sum_i (b_i - a_i)^2}\right).$$

In the section on communication functions, we will be using matrices everywhere. For a matrix M whose rows are indexed by elements of $\mathcal{X}$ and whose columns are indexed by elements of $\mathcal{Y}$, we will use $M[x, y]$ to denote the (x, y)th entry, $M[x, *]$ to denote the row indexed by x and $M[*, y]$ to denote the column indexed by y.

The following observations will be useful when analyzing decompositions of matrices.

- For vectors $u, v \in \mathbb{R}^k$ and a real number p, $\langle \sqrt{p}u, \sqrt{p}v \rangle = p\langle u, v \rangle$.
- For vectors $u_1, v_1 \in \mathbb{R}^{k_1}$ and $u_2, v_2 \in \mathbb{R}^{k_2}$, $\langle u_1 \circ u_2, v_1 \circ v_2 \rangle = \langle u_1, v_1 \rangle + \langle u_2, v_2 \rangle$ where $u_1 \circ u_2$ is the vector in $\mathbb{R}^{k_1+k_2}$ obtained by concatenating u_1 and u_2.
- For vectors $u_1, v_1 \in \mathbb{R}^{k_1}$ and $u_2, v_2 \in \mathbb{R}^{k_2}$, $\langle u_1 \otimes u_2, v_1 \otimes v_2 \rangle = \langle u_1, v_1 \rangle \cdot \langle u_2, v_2 \rangle$ where $u_1 \otimes u_2$ is the vector in $\mathbb{R}^{k_1 k_2}$ obtained by tensoring u_1 and u_2.

Since the matrix product XY is created by taking inner products of rows and columns of X and Y, the above observations allow us to create decompositions of new matrices.

Observation 3.1.2. *If $M_1 = X_1Y_1$ and $M_2 = X_2Y_2$ are in $\mathbb{R}^{m \times n}$, then the following hold.*

- *$M' = M_1 + M_2$ has the decomposition $X'Y'$ where $X'[i,*] = X_1[i,*] \circ X_2[i,*]$ and $Y'[*,j] = Y_1[*,j] \circ Y_2[*,j]$.*
- *The matrix created by taking the entrywise products of M_1 and M_2, has the decomposition $X'Y'$ where $X'[i,*] = X_1[i,*] \otimes X_2[i,*]$ and $Y'[*,j] = Y_1[*,j] \otimes Y_2[*,j]$.*

We will also be dealing with the Fourier spectrum of functions (see [O'D14] for a thorough introduction to Fourier analysis of Boolean functions).

Definition 3.1.3 (Fourier coefficients)**.** *Consider the vector space of functions $V = \{f : \{0,1\}^n \to \mathbb{R}\}$ equipped with the inner product defined by*

$$\langle f, g \rangle := \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)g(x).$$

The set of parity *functions $\{\chi_S : \{0,1\}^n \to \{-1,1\}\}_{S \subseteq [n]}$, where $\chi_S(x) = (-1)^{\sum_{i \in S} x_i}$, forms an orthonormal basis for this vector space under the inner product defined above. Thus, every function $f : \{0,1\}^n \to \mathbb{R}$ has a unique representation $f = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S$. The coefficients $\{\hat{f}(S)\}_{S \subseteq [n]}$ are called the* Fourier coefficients *of f.*

Theorem 3.1.4 (Parseval's Theorem)**.** *For a function $f : \{0,1\}^n \to \mathbb{R}$, we have*

$$\sum_{S \subseteq [n]} \hat{f}(S)^2 = \hat{f}(\emptyset)^2.$$

3.2 Measures for Communication Functions

3.2.1 Deterministic

Recall from Observation 2.1.3 that a cost-c deterministic communication protocol is a depth c binary tree which has, among others, the following properties.

- Each leaf corresponds to a combinatorial rectangle.
- The rectangles at the leaves partition the whole input space.
- Given an input, the output of the protocol is the label of the leaf that the input lands in.

Rectangle Partition Number

It follows that if F is a function computed by a cost-c deterministic communication protocol, then the at most 2^c leaves of the protocol partition the set of inputs into at most 2^c rectangles, each rectangle being monochromatic (each rectangle consists solely of 1-inputs or solely of 0-inputs). This yields the following lower bound.

Definition 3.2.1 (Rectangle Partition Number)**.** *Let* $\mathsf{Rect}_1(F)$ *be defined as the minimum number* k *for which there exist* k *disjoint rectangles* $\{A_i \times B_i\}_{i\in[k]}$ *such that* $\bigcup_{i\in[k]} A_i \times B_i = F^{-1}(1)$*. We similarly define* $\mathsf{Rect}_0(F)$ *with* $F^{-1}(0)$ *instead.*

Lower Bound 1: Rectangle Partition Number [Yao79]

$\mathsf{D}^{\mathsf{cc}}(F) \geq \log(\mathsf{Rect}_1(F))$.

Note that the above bound is extremely one-sided in that it cares only about partitioning the 1-inputs. The lower bound could have been derived as $\mathsf{D}^{\mathsf{cc}}(F) \geq \log(\mathsf{Rect}_0(F) + \mathsf{Rect}_1(F))$. However, the following theorem shows that the weak bound itself is quadratically tight.[1]

Theorem 3.2.2 ([AUY83, Yan91])**.** $\mathsf{D}^{\mathsf{cc}}(F) \leq O(\log^2(\mathsf{Rect}_1(F)))$.

Proof. Let $\mathcal{R}$ be a set of monochromatic 1-rectangles of size $\mathsf{Rect}_1(F)$ partitioning $F^{-1}(1)$. Given an input (x, y), Alice and Bob compute $\mathcal{R}_x = \{R = A \times B \in \mathcal{R} | x \in A\}$ and $\mathcal{R}_y = \{R = A \times B \in \mathcal{R} | y \in B\}$ respectively. Note that $\mathcal{R}_x \cap \mathcal{R}_y$ is the set of 1-rectangles of $\mathcal{R}$ that (x, y) lies in, and hence has size either 0 or 1. Given rectangles $R_1 = A_1 \times B_1$ and $R_2 = A_2 \times B_2$, we say that R_1 and R_2 row-intersect if $A_1 \cap A_2 \neq \emptyset$, and column-intersect if $B_1 \cap B_2 \neq \emptyset$.

Given x, it would be helpful if Alice could convey a lot of information about $\mathcal{R}_x$ to Bob. In particular, if there was a rectangle $R = A \times B \in \mathcal{R}_x$ that row-intersects with at most $\mathsf{Rect}_1(F)/2$ other 1-rectangles, then Alice could mention that rectangle to Bob and Bob could cut down Alice's possible rectangles by half. In other words they recurse on the function F', which is F restricted to the inputs $A \times \mathcal{Y}$. By definition, $\mathsf{Rect}_1(F') \leq \mathsf{Rect}_1(F)/2 + 1$.

Similarly if Bob could find a rectangle $R \in \mathcal{R}_y$ that doesn't column-intersect with many rectangles, he could convey this to Alice and they can again recurse.

[1] A consequence of this theorem is the neat combinatorial fact that $\log(\mathsf{Rect}_1(F)) \leq O(\log^2(\mathsf{Rect}_0(F)))$.

If $\mathsf{Rect}_1(F) = O(1)$, it is easy to see that there is a constant cost protocol for it.
If neither of them can find such a rectangle, it turns out that the input is not in a 1-rectangle and they can output 0. We can reason as follows. Let $(x, y) \in R \in \mathcal{R}_x \cap \mathcal{R}_y$. We know R row-intersects with at least $\mathsf{Rect}_1(F)/2 + 1$ other 1-rectangles and column-intersects with at least $\mathsf{Rect}_1(F)/2 + 1$ other 1-rectangles. Hence it intersects with another 1-rectangle, which is not possible since $\mathcal{R}$ is a partition.
The number of times the recursion can happen is at most $O(\log(\mathsf{Rect}_1(F)))$. The number of bits required to be communicated in each recursion is at most $\lceil \log(\mathsf{Rect}_1(F)) \rceil + O(1)$.
The total cost of the protocol is at most $O(\log^2(\mathsf{Rect}_1(F)))$. □

There are functions known that do exhibit the quadratic gap [GPW15].

Rank and Nonnegative Rank

For a rectangle R, let $R(x, y)$ be the function that is 1 if $(x, y) \in R$ and 0 otherwise. A consequence of having a partition of F into monochromatic rectangles is that F can be written as $F(x, y) = \sum_{i \in [\mathsf{Rect}_1(F)]} R_i(x, y)$, where the rectangles R_i come from the definition of $\mathsf{Rect}_1(F)$ (Definition 3.2.1). Note that the communication matrix for the function $R(x, y)$ is a rank-1 matrix. By the subadditivity of rank we get that $\mathsf{rank}(F) \leq \mathsf{Rect}_1(F)$, and consequently we have the following lower bound.

Lower Bound 2: Rank [MS82]

$\mathsf{D}^{\mathsf{cc}}(F) \geq \log(\mathsf{rank}(F))$.

Unlike the case with the rectangle partition number, we do not know whether this lower bound is tight. Given that rank is such a fundamental measure, this open question is regarded as the biggest open problem in communication complexity.

Conjecture 3.2.3: The Log-Rank Conjecture (LRC) [LS88]

There exists a universal constant α such that for any total communication function F, we have that $\mathsf{D}^{\mathsf{cc}}(F) \leq O(\log^{\alpha}(\mathsf{rank}(F)))$.

The best known upper bound in terms of rank is $\mathsf{D}^{\mathsf{cc}}(F) \leq \sqrt{\mathsf{rank}(F)} \log(\mathsf{rank}(F))$ [Lov16]. There is a much easier upper bound of $\mathsf{rank}(F)$ itself, but we will see a better (and surprising) upper bound using approximate rank later on in this chapter (Theorem 3.2.15).

Towards falsifying the conjecture, the quadratic separation between deterministic communication and the logarithm of the rectangle partition number gives us a quadratic separation between deterministic communication and the logarithm of the rank.

One reason why rank might be much smaller than the rectangle partition number is that rank-1 matrices are not constrained to only have entries in $\{0, 1\}$, like rectangles are constrained

to do. Relaxing this constraint partway gives us an intermediate measure, that of nonnegative rank.

Definition 3.2.4 (Nonnegative Rank)**.** *For a function F with communication matrix M the nonnegative rank of F, $\mathsf{rank}^+(F)$, is defined as the minimum k for which there exist nonnegative rank-1 matrices $\{M_i\}_{i\in[k]}$ such that $M = \sum_{i\in[k]} M_i$.*

> Nonnegative rank is actually defined for any nonnegative matrix and is an interesting measure on matrices even when the matrices are not Boolean [Yan91]. Interestingly, the logarithm of the nonnegative rank of *any* nonnegative matrix is exactly characterized by the *randomized* communication complexity of an associated task [FFGT15]: Accept input (x, y) with probability proportional to $M[x, y]$.

Lower Bound 3: Nonnegative Rank [Yan91]

$\mathsf{D}^{\mathsf{cc}}(F) \geq \log(\mathsf{rank}^+(F))$.

Note that this lower bound is one-sided, similar to our rectangle partition number lower bound (Lower bound 1). This could actually be strengthened to

$$\mathsf{D}^{\mathsf{cc}}(F) \geq \max\left\{\log(\mathsf{rank}^+(F)), \log(\mathsf{rank}^+(\overline{F}))\right\},$$

since D^{cc} is closed under complementation. However, this does not make much of a difference thanks to the following theorem that says that the one-sided lower bound itself is quadratically tight.[2]

Theorem 3.2.5 ([Lov90])**.** $\mathsf{D}^{\mathsf{cc}}(F) \leq O(\log^2(\mathsf{rank}^+(F)))$.

Proof. The communication matrix of F decomposes into $r = \mathsf{rank}^+(F)$ nonnegative rank-1 matrices $M_1, M_2, \ldots, M_r$. Since they are nonnegative and rank-1, the non-zero elements of each M_i must actually be rectangles in $F^{-1}(1)$. Let these rectangles be $R_1, R_2, \ldots, R_r$. Note that they need not be disjoint.

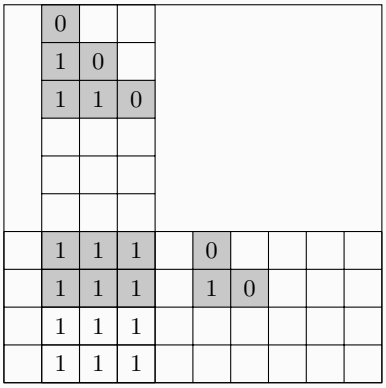
The measure that is used to mark the progress of the protocol is a bit weird. It is the maximum k for which there is a $k \times k$ submatrix of F with diagonal entries 0 and all entries below it 1. Let us refer to this as $k(F)$. Note that such a submatrix already gives a lower bound of $k(F) - 1$ on $\mathsf{rank}(F)$.

For each of the r rectangles $R_i = A_i \times B_i$, define $S_i = A_i \times \mathcal{Y}$ and $T_i = \mathcal{X} \times B_i$. Note that $k(F) \geq k(S_i) + k(T_i)$ since the submatrices in S_i and T_i can be combined using the 1s in R_i.

[2] A consequence of this theorem is the interesting algebraic fact that $\log(\mathsf{rank}^+(F)) \leq O(\log^2(\mathsf{rank}^+(\overline{F})))$.

The figure to the right illustrates how for any 1-rectangle $R = A \times B$,

$$k(F) \geq k(A \times \mathcal{Y}) + k(\mathcal{X} \times B).$$



If $k(S_i) \leq k(F)/2$, we call R_i an Alice-favoured rectangle. We similarly define Bob-favoured rectangles. Each of the rectangles is either Alice-favoured or Bob-favoured. If there is an Alice-favoured rectangle R_i such that $x \in A_i$, then Alice sends Bob i and they recurse on the function F restricted to S_i. Similarly, if there is a Bob-favoured rectangle R_i with $y \in B_i$, they recurse on F restricted to T_i. If none of these conditions hold, this means that (x, y) is in none of the 1-rectangles and $F(x, y) = 0$.
When $k(F) \leq 1$, it is easy to see that the function has constant communication cost. The total number of recurrences is at most $O(\log(k(F)))$, and each recurrence uses communication at most $\log(r) + O(1)$. The total communication in the protocol is hence at most $O(\log(r) \log(k(F))) \leq O(\log(\mathsf{rank}^+(F)) \log(\mathsf{rank}(F)))$. □

This also gives an equivalent and interesting reformulation of the Log-Rank Conjecture: Is there a Boolean matrix such that its rank is much smaller than its nonnegative rank?

We also note that the theorem is slightly stronger than stated. The proof shows that $\mathsf{D}^{\mathsf{cc}}(F) \leq \log \mathsf{rank}(F)\mathsf{NP}^{\mathsf{cc}}(F)$, since the NP^{cc} complexity of a function is the logarithm of the minimum number of 1-monochromatic rectangles needed to cover the 1s of the function.

γ_2 Norm

A final lower bound that we shall look at for deterministic communication complexity is a weighing measure that has been widely studied by Banach space theorists and introduced to communication complexity relatively recently [LMSS07].

Definition 3.2.6 (γ_2)**.** *Let F be a function with communication matrix M. $\gamma_2(M)$ is defined as $\min_{M=XY} \mathsf{row}(X)\mathsf{col}(Y)$, where $\mathsf{row}(X)$ is the maximum of the ℓ_2 norms of the rows of X and $\mathsf{col}(Y)$ is the maximum of the ℓ_2 norms of the columns of Y. By $\gamma_2(F)$, we refer to $\gamma_2(M)$.*

γ_2 is in fact a norm. $\gamma_2(M)$ can only be zero when one of the matrices in the decomposition is an all-zero matrix, and hence M itself is the all-zero matrix. It is easy to see that $\gamma_2(\alpha M) = \alpha\gamma_2(M)$. We can see the subadditivity of γ_2 as follows.

If $\gamma_2(M_1) = a_1$ and $\gamma_2(M_2) = a_2$, then there exist X_1, Y_1, X_2, Y_2 such that $M_1 = X_1Y_1$, $M_2 = X_2Y_2$, $\mathsf{row}(X_1) = \mathsf{col}(Y_1) = \sqrt{a_1}$ and $\mathsf{row}(X_2) = \mathsf{col}(Y_2) = \sqrt{a_2}$.[3] We now create matrices X' and Y' as per Observation 3.1.2 so that $M_1 + M_2 = X'Y'$. Now $\mathsf{row}(X') \leq \sqrt{\mathsf{row}(X_1)^2 + \mathsf{row}(X_2)^2} = \sqrt{a_1 + a_2}$. Similarly $\mathsf{col}(Y') \leq \sqrt{a_1 + a_2}$. Hence $\gamma_2(M_1 + M_2) \leq \gamma_2(M_1) + \gamma_2(M_2)$.

If $R = A \times B$ is a rectangle, it can be written as the outer product of the characteristic vectors of A and B. Hence $\gamma_2(R) = 1$, and $\gamma_2(F) \leq \mathsf{Rect}_1(F)$. This gives us a weaker lower bound.

Lower Bound 4: γ_2 [LS09d]

$\mathsf{D}^{\mathsf{cc}}(F) \geq \log(\gamma_2(F))$.

It is known that this bound is not tight. For instance, take the equality function where Alice gets n bits and Bob gets n bits and they want to accept iff the two bitstrings are equal. Its communication matrix is just the identity matrix, which can be decomposed as identity times identity. All rows and columns of the identity matrix have norm 1. Hence the equality function has γ_2 norm 1, but its communication complexity is $n + 1$.[4]

Seemingly coincidentally, equality has public coin randomized communication complexity $O(1)$. In fact, it was observed that no function is known to have small randomized communication complexity and large γ_2 norm [LS09d]. Perhaps, they conjectured, γ_2 lower bounds randomized communication complexity. This would be surprising since, as we will give some evidence for later on, none of the randomized lower bounds we know of involve a quantity that is an *exact* complexity measure. Yet the conjecture is open.

Conjecture 3.2.7: γ_2 as a Randomized Lower Bound [LS09d]

$\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \geq \log \gamma_2(F)$.

However, it might not be so surprising that there is no known counterexample to the above conjecture. After all, nearly every function known to have small randomized communication complexity has small complexity in a certain deterministic model that γ_2 continues to serve as a lower bound for. Consider the model of communication complexity wherein Alice and Bob are also given access to an equality oracle. This corresponds to a deterministic protocol tree with oracle nodes in addition to Alice and Bob nodes. For every oracle node v, there is an associated Alice function $f_{v,A} : \mathcal{X} \to \{0,1\}^*$ and a Bob function $f_{v,B} : \mathcal{Y} \to \{0,1\}^*$. On input (x, y) at such an oracle node v, the oracle communicates 1 if $f_{v,A}(x) = f_{v,B}(y)$, and 0 otherwise. Alice and Bob traverse the tree accordingly and output the value at the leaf that they reach. We call such trees P^{EQ} trees. A P^{EQ} tree computes a function F iff for every input

[3] This could involve suitably scaling matrices X_1 and Y_1 if $\mathsf{row}(\mathsf{X}_1)$ was not equal to $\mathsf{col}(\mathsf{Y}_1)$.

[4] The lower bound of $n + 1$ follows from the observation that the equality function has $\mathsf{Rect}_1 = 2^n$, and $\mathsf{Rect}_1 + \mathsf{Rect}_0 > 2^n$

(x, y), the leaf that (x, y) reaches is labeled $F(x, y)$.

Definition 3.2.8 ($\mathsf{P}^{\mathsf{EQcc}}$ complexity)**.** *The* $\mathsf{P}^{\mathsf{EQcc}}$ *complexity of* F *is defined as*

$$\min_{\mathsf{P}^{\mathsf{EQ}} \text{ trees } T \text{ computing } F} \mathsf{depth}(T).$$

Theorem 3.2.9. $\mathsf{P}^{\mathsf{EQcc}}(F) \geq \frac{\log(\gamma_2(F))}{2}$.

Proof. Let ℓ be a leaf of a P^{EQ} tree, with the path from the root being $r = v_1 \xrightarrow{b_{v_1}} v_2 \xrightarrow{b_{v_2}} v_3 \cdots v_k \xrightarrow{b_{v_k}} \ell$. Note that an input (x, y) reaches ℓ iff the following conditions hold.

- For every Alice node v along the path, $f_v(x) = b_v$.
- For every Bob node v along the path, $f_v(y) = b_v$.
- For every oracle node v along the path, $f_{v,A}(x) = f_{v,B}(y)$ iff $b_v = 1$.

We now see which inputs reach ℓ. Corresponding to every node v along the path, define the matrix M_{v,b_v} such that $M_{v,b_v}[x, y] = 1$ if (x, y) satisfies the above condition for node v and 0 otherwise. Define $M_\ell = \bigwedge_{i\in[k]} M_{v_i,b_{v_i}}$, where $M_1 \wedge M_2$ is obtained by taking the entrywise products of M_1 and M_2. Since the entries are Boolean, this is the same as taking their logical AND. M_ℓ is hence the characteristic matrix of the inputs that reach the leaf ℓ.
If v is an Alice node or a Bob node, then M_{v,b_v} is a rectangle and hence $\gamma_2(M_{v,b_v}) = 1$.
If v is an oracle node, then we claim that $M_{v,1} = 1$ and $M_{v,0} \leq 2$. To see this, let $S = \mathsf{range}(f_{v,A}) \cup \mathsf{range}(f_{v,B})$. Then $M_{v,1} = XY$, where the row of X corresponding to input x is the vector in $\{0, 1\}^S$ with a 1 in the coordinate corresponding to $f_{v,A}(x)$ (i.e. the unit vector $e_{f_{v,A}(x)}$) and the column of Y corresponding to input y is the vector in $\{0, 1\}^S$ with a 1 in the coordinate corresponding to $f_{v,B}(y)$. Hence $\gamma_2(M_{v,1}) = 1$. $M_{v,0}$ is the all-ones matrix minus $M_{v,1}$. Since γ_2 is a norm, it follows that $\gamma_2(M_{v,0}) \leq 2$.
One last observation we will need is that $\gamma_2(M_1 \wedge M_2) \leq \gamma_2(M_1)\gamma_2(M_2)$. Let $M = M_1 \wedge M_2$, $M_1 = X_1Y_1$ and $M_2 = X_2Y_2$. We now construct X' and Y' as per Observation 3.1.2 so that $M_1 \wedge M_2 = X'Y'$. Now $\mathsf{row}(X') \leq \mathsf{row}(X_1)\mathsf{row}(X_2)$ and similarly for the columns of Y'. This establishes that $\gamma_2(M_1 \wedge M_2) \leq \gamma_2(M_1)\gamma_2(M_2)$, and that $\gamma_2(M_\ell) \leq 2^k$.
Let T be a depth-c P^{EQ} tree computing F. Consider $\mathcal{L}$, the set of leaves labeled 1. They partition $F^{-1}(1)$ into at most 2^c sets. We can write $F(x, y) = \sum_{\ell\in\mathcal{L}} M_\ell(x, y)$. Since γ_2 is a norm, $\gamma_2(F) \leq \sum_{\ell\in\mathcal{L}} \gamma_2(M_\ell) \leq 2^c \cdot 2^c = 2^{2c}$. □

Until very recently, every total function that was known to be easy for randomized communication protocols was also known to be easy for P^{EQ} protocols. Hence γ_2 was small

for all known functions with low randomized communication cost. Things changed when Chattopadhyay, Lovett and Vinyals [CLV19] published a remarkable paper with a function that was easy for randomized communication but hard for P^{EQ} protocols. This gives us the first candidate function which could show that γ_2 does not lower bound randomized communication complexity. It is as yet unknown whether this function has large γ_2 norm.

More will be said about γ_2 and related measures when we analyze lower bounds on randomized communication complexity. To that end, it is useful to note the following link between γ_2 and rank, well known to matrix theorists.

Theorem 3.2.10. *For any matrix M, $\gamma_2(M) \leq \sqrt{\mathsf{rank}(M)} \max_{i,j} |M[i,j]|$.*

One of the proofs of this theorem uses John's theorem, which we state here without proof.

Theorem 3.2.11 (John's Theorem [Joh14])**.** *For any symmetric convex set $K \subset \mathbb{R}^d$, there is an ellipsoid E such that $E \subseteq K \subseteq \sqrt{d}E$.*

Here an ellipsoid refers to any shape obtained by taking the unit ball in $\mathbb{R}^d$ and transforming it by a linear transformation taking e_i to v_i for some set of orthogonal vectors $v_1, \ldots, v_d$. We now prove Theorem 3.2.10.

Proof. Let $\mathsf{rank}(M) = r$, with a rank decomposition $M = AB$. Let K be the symmetric convex set in $\mathbb{R}^r$ obtained by taking the convex hull of the columns of B and $-B$. Let E be the ellipsoid guaranteed by John's Theorem. The convex hull contains E and is contained in $\sqrt{r}E$. Let T be a linear transformation taking the unit ball to E.
We now consider the decomposition $M = ATT^{-1}B$. Let $A' = AT$ and $B' = T^{-1}B$. The convex hull of the columns of B' and $-B'$ will now contain the unit ball and be contained in the ball of radius $\sqrt{r}$. A consequence of this is that for any vector $v \in \mathbb{R}^r$, $\frac{v}{\|v\|}$ is in this convex hull. Hence there is a column w of B' or $-B'$ such that $\langle v, w \rangle \geq \langle v, \frac{v}{\|v\|} \rangle = \|v\|$.
In particular for any row of A', say $A'[i,*]$, $\|A'[i,*]\| \leq \max_j |\langle A'[i,*], B'[*,j] \rangle| = \max_j |M[i,j]|$. Since additionally every column of B' has length at most $\sqrt{r}$, $\gamma_2(M) \leq \sqrt{r} \max_{i,j} |M[i,j]|$. □

The proof above also gives us the following observation.

Observation 3.2.12. *Every matrix M of rank r has a rank decomposition $M = AB$ which also witnesses $\gamma_2(M) \leq \sqrt{r} \max_{i,j} |M[i,j]|$.*

3.2.2 Randomized

We now move on to randomized communication protocols. For a randomized protocol computing F to output the correct answer on input (x, y), the protocol must output 1 with probability close to $F(x, y)$. In our attempts to lower bound $\mathsf{R}^{\mathsf{cc}}_{\epsilon}(F)$, we will try to show that no matrix of

probabilities that is close to the communication matrix of F is 'mathematically simple', hence implying that none of them could have come from a small-cost protocol. In case there is such a 'mathematically simple' matrix close to the communication matrix of F but F has no small randomized protocol, then our lower bound will not be able to observe the hardness of F.

Note that we have gone from hoping (in the deterministic case) that 'every Boolean matrix which is simple must come from a small-cost deterministic protocol' to now essentially hoping that 'every matrix of probabilities which is simple must come from a small-cost randomized protocol'.

Nevertheless one lower bound is better than none. So let us see what lower bounds we get with this hope and how good they are.

Recall from Observation 2.1.6 that a cost-c private coin communication protocol is a depth c binary tree which has, among others, the following properties.

- For any leaf ℓ, the matrix whose entries are the probabilities that the various inputs (x, y) reach ℓ is a rank-1 matrix.
- For any input, the probability that the protocol outputs o is the probability that the input reaches a leaf labeled o.

Approximate Ranks

Let Π be a cost-c private coin communication protocol and let $\mathcal{L}$ be the set of leaves of Π labeled 1. Let M_ℓ be the matrix such that $M_\ell[x, y] = \Pr[(x, y) \text{ reaches } \ell]$. We know that M_ℓ is a rank-1 matrix. Since $\Pr[\Pi(x, y) = 1] = \sum_{\ell \in \mathcal{L}} M_\ell(x, y)$, we have that M_Π, the matrix such that $M_\Pi[x, y] = \Pr[\Pi \text{ accepts } (x, y)]$, has rank at most 2^c. This motivates the following definition.

Definition 3.2.13 (Approximate Rank)**.** *For a function F with communication matrix M the ϵ-approximate rank of F, which we denote as $\mathsf{rank}_\epsilon(F)$, is defined as*

$$\min_{M' : ||M - M'||_\infty \leq \epsilon} \mathsf{rank}(M').$$

The paragraph above showed that for any F with an ϵ-error cost-c private coin communication protocol, $\mathsf{rank}_\epsilon(F) \leq 2^c$.

Lower Bound 5: Approximate Rank [Kra96]

$\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},\epsilon}(F) \geq \log(\mathsf{rank}_\epsilon(F))$.

As was the case with the deterministic rank lower bound, it had long been open[5] whether this lower bound is tight.

[5]The earliest reference we could find to the conjecture was from Wikipedia user ForgeGod on the page for 'Communication Complexity' [For05]. It did have a typographical error, so we will default to crediting Lee and Shraibman [LS09b].

Conjecture 3.2.14: The Log-Approximate-Rank Conjecture (LARC) [LS09b]

There exists a universal constant α such that for any total communication function F, we have that $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\log^{\alpha}(\mathsf{rank}_{1/3}(F)))$.

This conjecture has been proven false in this thesis in Chapter 4.

Interestingly the Log-Approximate-Rank Conjecture is known to imply the Log-Rank Conjecture [GL14].

It is worth noting that one can decrease the error parameter in the approximate rank with a modest increase in the approximate rank [Alo03]. Given a matrix M and a degree-d univariate real polynomial $p(t) = \sum a_i t^i$, one can create a matrix M' such that $M'[x,y] = p(M[x,y])$ with $\mathsf{rank}(M') \leq \sum_{i\in[d]} \mathsf{rank}(M)^i$. This follows directly by using Observation 3.1.2. Given any $\epsilon, \epsilon' < \epsilon$, there is an $O(\frac{\log(1/\epsilon')}{1-2\epsilon})$ degree polynomial that maps the intervals $[0,\epsilon]$ and $[1-\epsilon,1]$ to within the intervals $[0,\epsilon']$ and $[1-\epsilon',1]$ respectively.

> Consider a t-biased coin tossed d times. The probability of seeing more than $d/2$ Heads can be written as a degree-d polynomial in t. Using Hoeffding's lemma (Lemma 3.1.1), one can see that with the mentioned degree, the polynomial satisfies the required conditions.

Putting the above together, the ϵ'-approximate rank of M is at most $\mathsf{rank}_{\epsilon}(M)^{O(\frac{\log(1/\epsilon')}{1-2\epsilon})}$. This blow-up is small when one looks at $\log \mathsf{rank}_{\epsilon}$.

The best known upper bound in terms of approximate rank is $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\mathsf{rank}_{\epsilon}(F))$ for any constant $\epsilon < \frac{1}{2}$ [GS19]. While they give multiple ways to prove this, a particularly striking method shows that the deterministic one-way communication cost[6] of F is $O(\mathsf{rank}_{\epsilon}(F))$. We reproduce one such elegant proof here.

Theorem 3.2.15. *[[GS19]] For any communication function F, the one-way deterministic communication complexity of F is at most $\lceil(\mathsf{rank}_{\epsilon}(F)+1)\log(\frac{2}{1-2\epsilon})\rceil$.*

Proof. The idea behind this proof is essentially that one can not pack a large number of cubes in a larger cube if the dimension is small.
Let $r = \mathsf{rank}_{\epsilon}(F)$. This implies the existence of a matrix M that is entrywise ϵ-close to the communication matrix of F and that has rank r. Consider the matrix $M' = M - \frac{1}{2}J$. It has rank at most $r+1$ and has entries in $[-\frac{1}{2}-\epsilon, -\frac{1}{2}+\epsilon] \cup [\frac{1}{2}-\epsilon, \frac{1}{2}+\epsilon]$. Let $v_1, v_2, \cdots, v_{r+1} \in \mathbb{R}^{2^n}$ be rows of M' that span the rows of M'. Let the subspace S be defined as their span.
Let the number of distinct rows in the communication matrix of F be N_F. Any pair of rows among these has ℓ_∞ distance at least 1 since they must differ in at least 1 entry. The corresponding rows in M' will then have ℓ_∞ distance at least $1-2\epsilon$. Let the rows in

[6] The number of bits Alice needs to send Bob for Bob to output the correct answer.

M' corresponding to the N_F distinct rows of F be $w_1, w_2, \cdots, w_{N_F}$. Hence if we draw open ℓ_∞ balls (essentially hypercubes) of radius $\frac{1}{2} - \epsilon$ around each w_i, these must be disjoint. Let us call these balls $B_1, B_2, \cdots, B_{N_F}$. All these open balls are contained in another open ball: the open ℓ_∞ ball B of radius 1 centered around the origin. We now note that a similar structure can be observed even when restricted to the subspace S. Note that $B_i \cap S = w_i + (\frac{1}{2} - \epsilon)(B \cap S)$, since the right hand side is exactly those points that are in S and are at a distance of at most $\frac{1}{2} - \epsilon$ from w_i. Hence $B \cap S$ contains the N_F disjoint sets $w_i + (\frac{1}{2} - \epsilon)(B \cap S)$. Since the ratio of the volumes (when measured in the subspace S) of $(\frac{1}{2} - \epsilon)(B \cap S)$ and $B \cap S$ is $(\frac{1}{2} - \epsilon)^{\dim(S)}$, we see that $N_F \leq (1/(\frac{1}{2} - \epsilon))^{r+1}$.
For the one-way protocol, Alice merely needs to tell Bob which of the N_F rows her input x corresponds to, which takes $\lceil \log(N_F) \rceil$ bits. □

The same paper shows that in the case of quantum communication complexity, one has a stronger upper bound of $O\left(\frac{1}{(1-2\epsilon)^2}\sqrt{\mathsf{rank}_\epsilon(F)}\log(\mathsf{rank}_\epsilon(F)\right)$.

Towards falsifying the conjecture, the set disjointness function shows a quadratic separation. Its randomized communication complexity is known to be $\Omega(n)$ [KS92, Raz92]. Its approximate rank is at most $2^{O(\sqrt{n})}$ since its quantum communication complexity is $O(\sqrt{n})$ [AA05] and the logarithm of the approximate rank lower bounds quantum communication complexity [BdW01]. A fourth-power separation between randomized communication and the logarithm of the approximate rank was recently discovered [GJPW17].

One reason approximate rank might not be a good representation of randomized communication complexity is that it allows decompositions involving negative numbers, whereas the decomposition coming from the protocol did not use negative numbers. In the case of deterministic communication, disallowing negative decompositions made the lower bound tight. It would be worthwhile to try it out here too.

Definition 3.2.16 (Approximate Nonnegative Rank)**.** *For a function F with communication matrix M the ϵ-approximate nonnegative rank of F, which we denote as $\mathsf{rank}^+_\epsilon(F)$, is defined as $\min_{M':||M-M'||_\infty \leq \epsilon} \mathsf{rank}^+(M')$.*

Lower Bound 6: Approximate Nonnegative Rank [Kra96]

$\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},\epsilon}(F) \geq \max\{\log(\mathsf{rank}^+_\epsilon(F)), \log(\mathsf{rank}^+_\epsilon(\overline{F}))\}$.

Set Disjointness has 1/3-approximate nonnegative rank $2^{\Omega(n)}$ [Raz92, KMSY14], so this method is known to give better lower bounds than approximate rank. However, unlike the deterministic case, it is still open whether this bound is tight. Lee [Lee12] asked how large the separation can be between approximate rank and approximate nonnegative rank. Again,

unlike the deterministic case, it was also not known whether $\log(\mathsf{rank}^+_\epsilon(F))$ and $\log(\mathsf{rank}^+_\epsilon(\overline{F}))$ could be very different. This gives rise to two conjectures.

Conjecture 3.2.17: Log-Approximate-Nonnegative-Rank Conjecture [KMSY14]

There exists a universal constant α such that for any total communication function F, we have that $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\log^\alpha(\max\{\mathsf{rank}^+_\epsilon(F), \mathsf{rank}^+_\epsilon(\overline{F})\})$.

The second conjecture is a stronger version of the above, that looks at only F and not $\overline{F}$. We refer to this as the Strong Log-Approximate-Nonnegative-Rank Conjecture.

Conjecture 3.2.18: Strong Log-Approximate-Nonnegative-Rank Conjecture [KMSY14]

There exists a universal constant α such that for any total communication function F, we have that $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\log^\alpha(\mathsf{rank}^+_\epsilon(F))$.

This conjecture has been proven false in this thesis in Chapter 4.

The last rank measure we shall look at is that of sign rank, for which a very tight log-rank statement holds.

Definition 3.2.19 (Sign Rank [PS86])**.** *Given a function F with communication matrix M, the sign rank of F, denoted $\mathsf{rank}_\pm(F)$, is defined as $\min_{M':||M-M'||_\infty<\frac{1}{2}} \mathsf{rank}(M')$.*

The reason it is called sign rank becomes clear when it is used on functions that output values in $\{-1, 1\}$ instead of $\{0, 1\}$. Then the definition changes to being the minimum rank among all matrices M' that entrywise agree in sign with M. The sign ranks of a function under these two definitions differ by at most 1.

Let us define $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},<\frac{1}{2}}(F)$ to be the minimum cost among private coin protocols that on all inputs (x, y), output $F(x, y)$ with probability greater than half. This measure is called the unbounded error communication cost.

Theorem 3.2.20 ([PS86])**.** $\lceil \log(\mathsf{rank}_\pm(F) + 1) \rceil + 2 \geq \mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},<\frac{1}{2}}(F) \geq \log \mathsf{rank}_\pm(F)$.

Proof. The lower bound follows just like the other rank lower bounds in this section. If a protocol of cost c correctly computes F, the probability of acceptance matrix has rank at most 2^c and this matrix must also approximate the communication matrix to within error $< \frac{1}{2}$. Therefore $\mathsf{rank}_\pm(F) \leq 2^{\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},<\frac{1}{2}}(F)}$.

For the upper bound, let the sign rank of F be r and let M' be a matrix in the definition of sign rank that achieves the rank r. Note that $M^\pm \triangleq 2M' - J$ has rank at most $r + 1$, where J is the all-ones matrix. Now $M^\pm[x, y]$ is positive iff $F[x, y] = 1$. Let $M^\pm = \sum_{i\in[r+1]} M_i$, where each M_i has rank 1. Scale $M^\pm$ till each matrix in the

decomposition has absolute value at most 1 for all its entries (we will continue to refer to the scaled matrices by their old names). Alice and Bob, who have inputs x and y respectively, will now aim to output 1 with probability $\frac{1}{2} + \frac{M^{\pm}[x,y]}{2(r+1)}$, which gives them a positive advantage, completing the theorem.

To do so, Alice samples a uniformly random number i from $[r+1]$ and sends it to Bob. Now M_i can be written as the outer product of two vectors v and w, each of which have the absolute value of any entry at most 1. Alice tosses a Bernoulli$\left(\frac{1+v[x]}{2}\right)$ coin and sends the outcome to Bob. Bob then tosses a Bernoulli$\left(\frac{1+w[y]}{2}\right)$ coin. He sends Alice a 1 if both were Heads or if both were Tails. Alice and Bob output that last bit.

The probability of outputting 1 given that they sampled M_i is

$$\left(\frac{1+v[x]}{2}\right)\left(\frac{1+w[y]}{2}\right) + \left(\frac{1-v[x]}{2}\right)\left(\frac{1-w[y]}{2}\right) = \frac{1+v[x]w[y]}{2} = \frac{1}{2} + \frac{1}{2}M_i[x,y].$$

The probability of outputting 1 in the overall protocol is $\frac{1}{r+1}\sum_{i\in[r+1]}(\frac{1}{2} + \frac{1}{2}M_i[x,y])$ which is $\frac{1}{2} + \frac{M^{\pm}[x,y]}{2(r+1)}$ as required. The total communication cost is $\lceil \log(\mathsf{rank}_{\pm}(F)+1)\rceil + 2$. □

There are many more fascinating lower bound methods known that follow from looking at public coin protocols. Recall that a cost-c public coin protocol is a distribution over cost-c deterministic protocols. Given an input (x, y) and a public coin protocol Π, the probability that Π outputs 1 on (x, y) is the probability that a deterministic protocol sampled from the distribution Π outputs 1 on (x, y).

Let Π be the distribution that samples deterministic protocols $\Pi_1, \ldots, \Pi_k$ with probabilities $p_1, \ldots, p_k$ respectively. Each Π_i is a partition of the input space into 2^c 0/1-labeled rectangles. Hence the protocol Π induces a weighted set of labeled rectangles. If the rectangle R appears with the label b in the partitions corresponding to protocols $\{\Pi_i\}_{i\in I}$, then (R, b) is given weight $w_{R,b} = \sum_{i\in I} p_i$. If Π is a protocol computing F to within error ϵ, this weighted set satisfies the following properties.

- $\forall (x,y) \in F^{-1}(1), \qquad \sum_{R:(x,y)\in R} w_{R,1} \in [1-\epsilon, 1]$.
- $\forall (x,y) \in F^{-1}(0), \qquad \sum_{R:(x,y)\in R} w_{R,0} \in [1-\epsilon, 1]$.
- $\forall (x,y), \qquad \sum_{R:(x,y)\in R} w_{R,0} + w_{R,1} = 1$.
- $\sum_{(R,b)} w_{R,b} \leq 2^c$.

The first two points above follow from the correctness of Π. The third follows from the fact that (x, y) lands in a rectangle with probability 1. The last point follows from the fact that the deterministic protocol Π_i can contribute at most $p_i 2^c$ weight and $\sum p_i = 1$.

The LP Bounds

A lower bound measure follows immediately from the above description. Let $\mathcal{R}$ be the set of all rectangles in a communication matrix.

Definition 3.2.21 ([JK10])**.** *For a function F, let $\mathsf{part}_\epsilon(F)$ be the optimal value of the following LP.*

$$
\begin{array}{lll}
\textit{Variables} & \{w_{R,b} : R \in \mathcal{R}, b \in \{0,1\}\} & \\
\textit{Minimize} & \sum_{R\in\mathcal{R}, b\in\{0,1\}} w_{R,b} & \\
\textit{s.t.} & \forall (x,y) \in F^{-1}(1) & \sum_{R:(x,y)\in R} w_{R,1} \geq 1-\epsilon \\
 & \forall (x,y) \in F^{-1}(0) & \sum_{R:(x,y)\in R} w_{R,0} \geq 1-\epsilon \\
 & \forall (x,y) & \sum_{R:(x,y)\in R} w_{R,0} + w_{R,1} = 1 \\
 & \forall R \in \mathcal{R}, b \in \{0,1\} & w_{R,b} \geq 0
\end{array}
$$

Lower Bound 7: Partition Bound [JK10]

$\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \log(\mathsf{part}_\epsilon(F))$.

However, lower bounds proved on the partition bound typically do not use all the constraints. The lower bounds obtained are mostly on relaxations of this LP. Let us look at a couple of such relaxations.

The smooth rectangle bound is obtained by either looking at the weights of rectangles labeled 1 or those of rectangles labeled 0. Since it is one-sided, we can parametrize the measure by which side we are looking at.

Definition 3.2.22 (Smooth Rectangle [JK10])**.** *For a function F, let $\mathsf{srect}^z_\epsilon(F)$ be the optimal value of the following LP.*

$$
\begin{array}{lll}
\textit{Variables} & \{w_R : R \in \mathcal{R}\} & \\
\textit{Minimize} & \sum_{R\in\mathcal{R}} w_R & \\
\textit{s.t.} & \forall (x,y) \in F^{-1}(z) & \sum_{R:(x,y)\in R} w_R \geq 1-\epsilon \\
 & \forall (x,y) \in F^{-1}(z) & \sum_{R:(x,y)\in R} w_R \leq 1 \\
 & \forall (x,y) \in F^{-1}(\overline{z}) & \sum_{R:(x,y)\in R} w_R \leq \epsilon \\
 & \forall R \in \mathcal{R} & w_R \geq 0
\end{array}
$$

Lower Bound 8: Smooth Rectangle Bound [JK10]

$\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \max\{\log(\mathsf{srect}^1_\epsilon(F)), \log(\mathsf{srect}^0_\epsilon(F))\}$.

This lower bound is particularly relevant as it has a connection to a previously seen lower bound. Kol et al. [KMSY14] show that this measure is almost equivalent to the approximate nonnegative rank. We shall see a proof of this in Section 3.4.

Again, the smooth rectangle bound is a stronger lower bound than a previously known and widely used bound. In this relaxation, we relax the approximation guarantee.

Definition 3.2.23 (Rectangle Bound [Lov90])**.** *For a function F, let $\mathsf{rect}^z_\epsilon(F)$ be the optimal value of the following LP.*

$$\begin{array}{lll} \textit{Variables} & \{w_R : R \in \mathcal{R}\} & \\ \textit{Minimize} & \sum_{R \in \mathcal{R}} w_R & \\ \textit{s.t.} & \forall (x,y) \in F^{-1}(z) & \sum_{R:(x,y)\in R} w_R \geq 1-\epsilon \\ & \forall (x,y) \in F^{-1}(\overline{z}) & \sum_{R:(x,y)\in R} w_R \leq \epsilon \\ & \forall R \in \mathcal{R} & w_R \geq 0 \end{array}$$

Lower Bound 9: Rectangle Bound [Lov90, Yao83, BPSW06]

$\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \max\{\log(\mathsf{rect}^1_\epsilon(F)), \log(\mathsf{rect}^0_\epsilon(F))\}$.

However, it turns out to be equivalent [JK10] to another previously known combinatorial lower bound known as the corruption bound, used implicitly by Yao [Yao83] and refined by Beame, Pitassi, Segerlind and Wigderson [BPSW06]. This bound has an intuitive interpretation, saying that every rectangle that's not minuscule is "corrupt". We provide its definition below and then show how it is equivalent to the smooth rectangle bound.

Definition 3.2.24 (Corruption Bound [Yao83, BPSW06])**.** *Fix an $\epsilon > 0$. Let μ be a distribution on the inputs of F with $\mu(F^{-1}(1)) \geq \frac{1}{2}$. Let $\beta \geq 0$ be such that for all rectangles R with $\mu(R) \geq \beta$ (i.e. for all large rectangles), $\mu(R \cap F^{-1}(0)) \geq \epsilon\mu(R \cap F^{-1}(1))$ (i.e. the rectangle is corrupt and cannot be too 1-biased). Then $\mathsf{corr}^1_\epsilon(F) = \min_{\mu,\beta} 1/\beta$.*

The proof that the corruption bound lower bounds communication is the same as the proof that it lower bounds the rectangle bound, so it is a corollary of the first part of the following theorem.

Theorem 3.2.25 (Equivalence of corruption and rectangle bounds [JK10])**.** *Fix an $\epsilon \geq 0$. Then for any function F,*

$$\mathsf{rect}^1_{\epsilon/2}(F) \geq \frac{1-\epsilon}{4}\mathsf{corr}^1_\epsilon(F) \text{ and } \mathsf{corr}^1_\epsilon(F) \geq \frac{1}{1+\epsilon}\mathsf{rect}^1_{2\epsilon}(F).$$

Proof. For the first part, let μ, β be the minimizers in the definition of $\mathsf{corr}^1_\epsilon(F)$. We will use them to prove a lower bound on $\mathsf{rect}^1_{\epsilon'}(F)$, finally setting $\epsilon' = \epsilon/2$. Let $\{w_R\}$ be a set of rectangle weights minimizing the LP in the definition on $\mathsf{rect}^1_{\epsilon'}(F)$. By the fact that $\mu(F^{-1}(1)) \geq \frac{1}{2}$ and that $\sum_{R \ni (x,y)} w_R$ for each $(x,y) \in F^{-1}(1)$, we derive that

$$\sum_R \mu(R \cap F^{-1}(1)) w_R \geq \frac{1-\epsilon'}{2} \text{ and } \sum_R \mu(R \cap F^{-1}(0)) w_R \leq \frac{\epsilon'}{2}.$$

Let $\mathcal{R}$ be the set of 'large' rectangles, namely those that satisfy $\mu(R) \geq \beta$. These are 'corrupt', so putting a lot of weight on them will introduce too much error. Indeed if $\sum_{R \in \mathcal{R}} \mu(R \cap F^{-1}(1)) w_R > \frac{\epsilon'}{2\epsilon}$, then $\sum_{R \in \mathcal{R}} \mu(R \cap F^{-1}(0)) w_R > \frac{\epsilon'}{2}$ contradicting the constraint from the rectangle bound. Hence

$$\sum_{R \in \mathcal{R}} \mu(R \cap F^{-1}(1)) w_R \leq \frac{\epsilon'}{2\epsilon}.$$

So a large portion of the weight must be covered by small rectangles. This will lead us to the lower bound.

$$\beta \sum_R w_R \geq \beta \sum_{R \notin \mathcal{R}} w_R \geq \sum_{R \notin \mathcal{R}} \mu(R) w_R \geq \sum_{R \notin \mathcal{R}} \mu(R \cap F^{-1}(1)) w_R \geq \frac{1-\epsilon'}{2} - \frac{\epsilon'}{2\epsilon}.$$

Hence $\mathsf{rect}^1_{\epsilon/2}(F) = \sum_R w_R \geq \frac{1-\epsilon}{4\beta} = \frac{1-\epsilon}{4} \mathsf{corr}^1_\epsilon(F)$.

The other direction is proved via linear programming duality. The dual of the given LP for the rectangle bound is as follows.

Variables	$\{\gamma_{x,y} : (x,y) \in \mathcal{X} \times \mathcal{Y}\}$	
Maximize	$(1-\epsilon) \left(\sum_{(x,y) \in F^{-1}(1)} \gamma_{x,y} \right)$	$-\epsilon \left(\sum_{(x,y) \in F^{-1}(0)} \gamma_{x,y} \right)$
s.t.	$\forall R \in \mathcal{R}$	$\sum_{\substack{(x,y) \in R \\ (x,y) \in F^{-1}(1)}} \gamma_{x,y} - \sum_{\substack{(x,y) \in R \\ (x,y) \in F^{-1}(0)}} \gamma_{x,y} \leq 1$
	$\forall (x,y) \in \mathcal{X} \times \mathcal{Y}$	$\gamma_{x,y} \geq 0$

We start with an optimal solution $\{\gamma_{x,y}\}$ for the dual LP with parameter ϵ'. Note that the constraints on the rectangles are a corruption-like constraint, saying that a rectangle cannot be both large and 1-biased. Indeed we start by creating a distribution that witnesses that all 'large' rectangles are corrupt. From $\{\gamma_{x,y}\}$, we create a distribution μ which samples $z \in \{0,1\}$ uniformly at random, and then chooses a z-input according to the weights $\gamma_{x,y}$. To understand μ, we note that there are constants c_0, c_1 such that for 0-inputs (x,y), $\mu(x,y) = c_0 \gamma_{x,y}$ and for 1-inputs (x,y), $\mu(x,y) = c_1 \gamma_{x,y}$. We get some

handle on these constants by looking at the objective function, which implies that

$$\sum_{1-\text{inputs}} \gamma_{x,y} \geq \mathsf{rect}^1_{\epsilon'}(F) \text{ and } \sum_{1-\text{inputs}} \gamma_{x,y} \geq \epsilon' \sum_{0-\text{inputs}} \gamma_{x,y}.$$

As a consequence, $c_1 \leq 1/2\mathsf{rect}^1_{\epsilon'}(F)$ and $c_0 \geq \epsilon' c_1$.
Using these, we can simplify the corruption-like constraint $\sum_{1-\text{inputs}\ \in R} \gamma_{x,y} - \sum_{0-\text{inputs}\ \in R} \gamma_{x,y} \leq 1$ to get the following implication about μ.

$$\begin{aligned}
&\frac{\mu(R \cap F^{-1}(1))}{c_1} - \frac{\mu(R \cap F^{-1}(0))}{c_0} \leq 1 \\
\implies &\mu(R \cap F^{-1}(1)) - \frac{1}{\epsilon'}\mu(R \cap F^{-1}(0)) \leq \frac{1}{2\mathsf{rect}^1_{\epsilon'}(F)} \\
\implies &\mu(R \cap F^{-1}(0)) \geq \epsilon' \left(\mu(R \cap F^{-1}(1)) - \frac{1}{2\mathsf{rect}^1_{\epsilon'}(F)}\right).
\end{aligned}$$

We now show that this distribution gives us a good corruption bound. Let R be a rectangle that is not ϵ-corrupt, i.e. $\mu(R \cap F^{-1}(0)) \leq \epsilon\mu(R \cap F^{-1}(1))$. Combining this with the above, we get that $\epsilon/\epsilon'\mu(R \cap F^{-1}(1)) \geq \mu(R \cap F^{-1}(1)) - \frac{1}{2\mathsf{rect}^1_{\epsilon'}(F)}$, or

$$\mu(R \cap F^{-1}(1)) \leq \frac{1}{2\mathsf{rect}^1_{\epsilon'}(F)(1 - \epsilon/\epsilon')}.$$

That is, all non-corrupt rectangles must have small 1-mass. Since they are not ϵ-corrupt, their 0-masses must also be at most an ϵ-fraction of this. Hence their whole masses can be upper bounded as $\mu(R) \leq \frac{1+\epsilon}{2\mathsf{rect}^1_{\epsilon'}(F)(1-\epsilon/\epsilon')}$. So any rectangle with $\mu(R)$ greater than this must be ϵ-corrupt. Hence $\mathsf{corr}^1_\epsilon(F) \geq \frac{2\mathsf{rect}^1_{\epsilon'}(F)(1-\epsilon/\epsilon')}{1+\epsilon}$. □

Interestingly the smooth rectangle bound is so named because it follows by generalizing the rectangle bound as follows [JK10]: Given a function f, find a function g that is close to f under a suitable probability distribution. The maximum such rectangle bound you can get from a g would be the smooth rectangle bound of f.

Note that the rectangle bound follows by relaxing the approximation guarantee of the smooth rectangle bound and the approximate rank bound follows by relaxing the nonnegativity constraint of the approximate nonnegative rank bound. So although both corruption and approximate rank are no larger than the approximate nonnegative rank (recall that we mentioned that approximate nonnegative rank and smooth rectangle bound are nearly equivalent), it is unclear whether one of them dominates the other. Although we prove in Chapter 4 that even the smaller of the two one-sided corruption bounds can be far larger than approximate rank, it is unknown whether the larger of the two one-sided corruption bounds is *always* at

least as large as the approximate rank.

The γ_2^α Measure

Since γ_2 is a norm, $\gamma_2\left(\sum p_i F_i\right) \leq \sum p_i \gamma_2(F_i)$. For a public coin protocol Π, the probability-of-acceptance matrix M_Π is exactly $\sum_i p_i \Pi_i$ where the p_is form a distribution. Since each Π_i has cost at most 2^c, $\gamma_2(\Pi_i)$ and hence $\gamma_2(M_\Pi)$ is at most 2^c. This gives us the requisite lower bound of $\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \log \gamma_{2,\epsilon}(F)$.

> The above reasoning would also show that $\gamma_{2,\epsilon}(F) \leq \mathsf{srect}^1_\epsilon(F)$, since $\gamma_2(R) = 1$ for any rectangle R.

However, "approximate γ_2" is not defined in this way. One writes approximate norms a bit differently. The reason will become clear when we look at γ_2^∞. For the actual definition, we need to look at functions that output values in $\{-1, 1\}$. (For clarity, let us call their communication matrices sign matrices.)

> The transformation $t \to 2t - 1$ does the required change from $\{0, 1\}$ to $\{-1, 1\}$. Consequences of this transformation are: the rank changes by at most 1, nonnegative ranks don't make sense, approximate rank also changes by at most 1 (with the error suitably changed to 2ϵ), and the γ_2 norm changes to at most twice its previous value, plus 1. Now that that's out of the way, let us look at the definition of γ_2^α.

Definition 3.2.26 (γ_2^α [LS09d])**.** *For a function F with sign matrix M, $\gamma_2^\alpha(F)$ is defined as* $\min_{M' : 1 \leq M[x,y]M'[x,y] \leq \alpha} \gamma_2(M')$.

Consider a cost-c public coin protocol Π computing F to within error $\frac{1}{2}(1 - \frac{1}{\alpha})$. Let M_Π be the probability-of-acceptance matrix as defined a few paragraphs ago. The entries of M_Π are in $[0, \frac{1}{2} - 1/2\alpha] \cup [\frac{1}{2} + 1/2\alpha, 1]$. The entries of $\alpha(2M_\Pi - J)$ are in $[-\alpha, -1] \cup [1, \alpha]$. Since $\gamma_2(M_\Pi) \leq 2^c$, we have $\gamma_2^\alpha(F) \leq \alpha(2 \cdot 2^c + 1)$.

Lower Bound 10: γ_2^α [LS09d]

$\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \log\left(\frac{\gamma_2^\alpha(F)}{\alpha} - 1\right) - 1$ where $\alpha = \frac{1}{1-2\epsilon}$.[a]

[a] This can also be written as $\mathsf{R}^{\mathsf{cc}}_\epsilon(F) \geq \log\left(\frac{\gamma_2^\alpha(F)}{\alpha}\right) - 2$ since $\log(x) - 2 \leq \max\{\log(x-1) - 1, 0\}$.

It was conjectured [LS09d] that $\mathsf{R}^{\mathsf{cc}}_{1/3}(F)$ could be polynomially related to the logarithm of approximate γ_2. This conjecture turns out to be almost equivalent to the Log-Approximate-Rank Conjecture since γ_2^α is closely related to approximate rank. Similar to the definition of γ_2^α, $\mathsf{rank}^\alpha(F)$ can be defined as the minimum rank among matrices M that sign-agree with the sign matrix of F and have values in $[-\alpha, -1] \cup [1, \alpha]$. It is essentially $\mathsf{rank}_\epsilon(F)$, where ϵ is around $\frac{1}{2} - \frac{1}{2\alpha}$. From Observation 3.2.12, we can see that $\gamma_2^\alpha(F) \leq \alpha\sqrt{\mathsf{rank}^\alpha(F)}$. We will look at a lower bound for γ_2^α in terms of $\mathsf{rank}^\alpha(F)$ in Section 3.4.

We now move to γ_2^∞, the most accomplished of the approximate γ_2 norms. It is defined by setting α to ∞ in the definition of γ_2^α.

Note that if we had defined $\gamma_{2,<\frac{1}{2}}$ the way we defined sign rank, then $\gamma_{2,<\frac{1}{2}}$ would tend to $\frac{1}{2}$ for every function F. This is because the matrix $J/2 + F/1000$ does approximate the matrix of F to within half, but has γ_2 norm $\leq \frac{1}{2} + \gamma_2(F)/1000$. This is also to be expected, as every function has a public coin protocol of cost 2 and advantage greater than 0: Alice sees the first n public coin tosses, and sends Bob a 1 if those tosses matched her input. Bob then sends Alice the output of the function. If Alice sent a 0, Bob would send a uniformly random bit to Alice and they would output that. The protocol outputs correctly with probability $\frac{1}{2} + 1/2^{n+1}$ on every input.

Let us define $\mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F)$ to be the minimum, over all public coin protocols that output the correct answer with probability $\geq \frac{1}{2} + \delta$ on every input, of the cost of the protocol plus $\log(1/\delta)$. This measure is called the weakly unbounded error communication cost [BFS86].

★ **Theorem 3.2.27.** $\log(\gamma_2^\infty(F)) + 2 \geq \mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F) \geq \log(\gamma_2^\infty(F)) - 1$.

The above theorem is known via a measure known as discrepancy. Logarithm of the discrepancy of F is known, via distributional complexity, to be an additive constant away from $\mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F)$ [CG85, Kla07]. It is also known that the logarithms of discrepancy and γ_2^∞ are the same up to an additive constant [LS09c], using LP duality and Grothendieck's inequality. The proof presented here is much simpler and does not involve any duality.

Proof. Let's start with the lower bound. Let Π be a cost-c public coin protocol that witnesses $\mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F) = r$. Say Π outputs the correct answer with probability $\geq \frac{1}{2} + \delta$ on all inputs. Our lower bound gives us that the cost of Π is at least $\log\left(\frac{\gamma_2^{1/2\delta}(F)}{1/2\delta}\right) - 2$. Hence

$$\begin{aligned} r &\geq \log\left(\frac{\gamma_2^{1/2\delta}(F)}{1/2\delta}\right) - 2 + \log(1/\delta) \\ &\geq \log(\gamma_2^{1/2\delta}(F)) - \log(1/2\delta) - 2 + \log(1/\delta) \\ &= \log(\gamma_2^{1/2\delta}(F)) - 1 \geq \log(\gamma_2^\infty(F)) - 1. \end{aligned}$$

Let $r = \gamma_2^\infty(F)$. Before we get to the communication protocol, we do some preprocessing. By definition, there is a matrix M with entries in $(-\infty, -1] \cup [1, \infty)$ that agrees in sign with the sign matrix of F and has a decomposition $M = XY$, with each row of X having ℓ_2 norm at most $\sqrt{r}$ and each column of Y having ℓ_2 norm at most $\sqrt{r}$. We add two extra coordinates to the rows of X and to the columns of Y so that each row of X has ℓ_2 norm exactly $\sqrt{r}$ and each column of Y has ℓ_2 norm exactly $\sqrt{r}$ without changing the inner product of any row and column. (Hence XY is still M.) Given the maximum row and column norms of X and Y respectively, we know by the Cauchy-Schwarz inequality that the maximum entry in M (in absolute value) is at most r. Finally, we scale M by $1/r$ to get a matrix with values in $[-1, -1/r] \cup [1/r, 1]$ that has a decomposition $M = XY$ where each row of X and each column of Y has ℓ_2 norm exactly 1.

Now for each input x, Alice has a vector $v_x = X[x, *]$ of norm 1 and for each input y, Bob has a vector $v_y = Y[*, y]$ of norm 1. They want to distinguish between the cases $\langle v_x, v_y \rangle \in [-1, -1/r]$ and $\langle v_x, v_y \rangle \in [1/r, 1]$, outputting -1 in the former case and 1 in the latter case. They wish to give the correct output with probability around $\frac{1}{2} + 1/r$. We will use randomized rounding [GW95] in order to accomplish this.

Let k be such that $v_x, v_y \in \mathbb{R}^k$ and let $a = \langle v_x, v_y \rangle$. Alice and Bob use public coins to choose a uniformly random unit vector w_r in $\mathbb{R}^k$. Alice sends Bob $\mathsf{sign}(\langle v_x, w_r \rangle)$. If it matches $\mathsf{sign}(\langle v_y, w_r \rangle)$, Bob outputs 1. Else Bob outputs -1. To analyze the probability of being correct, consider the component of w_r in the span of v_x and v_y. The signs only depend on this component. In this two-dimensional space, v_x and v_y have angle $cos^{-1}(a)$ between them. The perpendicular to the component of w_r has probability $2cos^{-1}(a)/2\pi$ of landing between them and making the two signs different. Hence the probability Bob outputs 1 is

$$\begin{aligned} 1 - cos^{-1}(a)/\pi &= 1 - \frac{\pi/2 - \sin^{-1}(a)}{\pi} \\ &= \frac{1}{2} + \frac{\sin^{-1}(a)}{\pi}. \end{aligned}$$

For all $a \in [-1, 1]$, $\frac{\sin^{-1}(a)}{a} \in [1, \frac{\pi}{2}]$. So for $a \in [1/r, 1]$, Bob outputs 1 with probability at least $\frac{1}{2} + 1/r$ and for $a \in [-1, -1/r]$ Bob outputs -1 with probability at least $\frac{1}{2} + 1/r$. Since the cost of the protocol is 2 bits, we get an upper bound of $\mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F) \leq 2 + \log(r)$. □

One other measure of note here before we wrap up this section on communication measures is that of the ℓ_1 norm of the Fourier coefficients (see Definition 3.1.3) of F, which we shall hereby refer to as the spectral norm of F, or $\left\|\widehat{F}\right\|_1$. Note that having a small protocol in no way guarantees small spectral norm. For instance, the function that computes the $\mathbb{F}_2$ inner product of the first and second halves of Alice's input requires only 1 bit to communicate to Bob, but has a large spectral norm. However, Grolmusz observed [Gro97] that all functions with small spectral norms seemed to have small randomized communication protocols. Buoyed by his observation in a previous paper [Gro96] that the number-on-forehead communication complexity with $\log n$ players[7] is at most polynomial in the logarithm of the ℓ_1 norm, Grolmusz makes the following conjecture.

[7] We will not be explaining any of those terms in this thesis.

Conjecture 3.2.28: Grolmusz's Conjecture [Gro97]

There exists a universal constant α such that for any total communication function F, we have that $\mathsf{R}^{\mathsf{cc}}_{1/3}(F) \leq O\left(\log^{\alpha}\left(\left\|\widehat{F}\right\|_1\right)\right)$.

This conjecture has been proven false in this thesis in Chapter 4.

Grolmusz referred to this as a randomized analogue of the Log-Rank Conjecture. This was not unjustified since, as will be implied by Grolmusz's Theorem (Theorem 3.4.2), this conjecture is essentially true if the Log-Approximate-Rank Conjecture is.

Lemma 3.2.29 (LARC implies Grolmusz's conjecture)**.** *For any function* $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$,

$$R_{1/3}(F) \leq \log^{O(1)} \mathsf{rank}_{1/3}(F) \implies R_{1/3}(F) \leq (\log \left\|\widehat{F}\right\|_1 + \log n)^{O(1)}.$$

Proof. Let $w = \left\|\widehat{F}\right\|_1$. Theorem 3.4.2 implies the existence of a function $G = \sum_{S\subseteq[2n]} c_S \chi_S$ such that $|G(x) - F(x)| \leq 1/3$ for all $x \in \{0,1\}^n \times \{0,1\}^n$ and $\mathsf{spar}(G) = O(\left\|\widehat{F}\right\|_1^2 n)$.
Next, note that for any $S \subseteq [2n]$, the function $c_S\chi_S$ is a matrix of rank at most 1. By the sub-additivity of rank, $\log \mathsf{rank}(G)$, and thus $\log \mathsf{rank}_{1/3}(F)$, is at most $O(\log \left\|\widehat{F}\right\|_1 + \log n)$. □

Grolmusz was only able to show a quadratic upper bound on the randomized communication complexity of a function in terms of its spectral norm.

3.3 Measures for Query Functions

3.3.1 Deterministic

Recall from Observation 2.2.2 that a cost-c deterministic parity decision tree is a depth c binary tree which has, among others, the following properties.

- Each leaf corresponds to an affine space of codimension at most c.
- The affine spaces at the leaves partition the whole input space.
- The output of the decision tree on any input is the label of the leaf that the input lands in.

Akin to the rectangle partition number, we can define the large affine space partition number lower bound.

Large Affine Space Partition

Definition 3.3.1 (Large Affine Space Partition Number)**.** *Let* $\mathsf{LargeAffSpace}_1(f)$ *be defined as the minimum number* k *for which there exist disjoint affine spaces of codimension* $\leq k$*, say* $\{S_i\}_{i\in[t]}$ *such that* $\bigcup_{i\in[t]} S_i = f^{-1}(1)$*. We similarly define* $\mathsf{LargeAffSpace}_0(F)$ *with* $f^{-1}(0)$ *instead.*

Note that the constraint that the codimension is at most k implies that the number of affine spaces is at most 2^k, since they form a partition.

Lower Bound 11: Large Affine Space Partition Number

$\mathsf{D}^{\oplus}(f) \geq \mathsf{LargeAffSpace}_1(f)$.

Similar to the rectangle partition number, this lower bound is quadratically tight. This is easy to see given the following observation, wherein a linear form ℓ being *set* in an affine space S means that $\forall x_1, x_2 \in S, \langle \ell, x_1\rangle = \langle \ell, x_2\rangle$.

Observation 3.3.2. *If* S_1 *and* S_2 *are disjoint affine spaces, then there is a linear form* ℓ *such that* ℓ *is set in both* S_1 *and* S_2*.*

Before we formally prove the observation, we will see why it implies that the lower bound is quadratically tight. We give an upper bound of $\mathsf{D}^{\oplus}(f) \leq \mathsf{LargeAffSpace}_1(f)^2$. Simply choose an affine space of the partition and query all the constraints that it sets. If the input is in the affine space, output 1. Else, every subspace in the partition has a linear constraint set, and so they either become empty or have codimension $\leq \mathsf{LargeAffSpace}_1(f) - 1$. Hence we can continue this at most $\mathsf{LargeAffSpace}_1(f) - 1$ more times. If the input is never in a 1-affine space, then output 0. The number of queries made is at most $\mathsf{LargeAffSpace}_1(f)^2$.

Proof. Let $\ell_{1,1}, \ell_{1,2}, \ldots, \ell_{1,k_1}$ be a basis of linear forms set by S_1 to $a_{1,1}, a_{1,2}, \ldots, a_{1,k_1}$ respectively. Let $\ell_{2,1}, \ell_{2,2}, \ldots, \ell_{2,k_2}$ be a basis of linear forms that are set by S_2 to $a_{2,1}, a_{2,2}, \ldots, a_{2,k_2}$ respectively. If any $\ell_{1,i}$ is the same as $\ell_{2,j}$ the observation is true and we are done. Otherwise we create the affine subspace $S_1 \cap S_2$ iteratively by adding the linear constraints in the order listed above. Since $S_1 \cap S_2 = \emptyset$, at some point a linear constraint $\ell_{2,i} = a_{2,i}$ is added that makes the affine subspace empty. This implies that $\ell_{2,i}$ is dependent on $\{\ell_{1,j}\}_{j\in[k_1]} \cup \{\ell_{2,j}\}_{j\in[i-1]}$. This dependency must involve some constraints from S_1. Writing down this dependency, we get that there are sets $I_1 \subseteq [k_1], I_2 \subseteq [i]$ such that $\sum_{j\in I_1} \ell_{1,j} = \sum_{j\in I_2} \ell_{2,j}$. This proves the observation. □

Sparsity, Spectral Norm and Parity Kill Number

A less combinatorial and more algebraic measure is that of Fourier sparsity. (We will simply refer to this as sparsity.) Recall the definition of Fourier coefficients (Definition 3.1.3).

Definition 3.3.3 (Sparsity)**.** *Define the sparsity of a function $f : \{0,1\}^n \to \mathbb{R}$ as follows.*

$$\mathsf{spar}(f) := \left|\{S \subseteq [n] \mid \hat{f}(S) \neq 0\}\right|.$$

The Fourier representation of an affine space is easy to describe. Let A be an affine space of codimension k defined as $\bigwedge_{t\in[k]} \oplus_{j\in S_t} x_j = a_t$. The indicator function of A is represented by the polynomial

$$p_A(x) = \prod_{t\in[k]} \left(\frac{1 + (-1)^{a_t}\chi_{S_t}(x)}{2}\right).$$

From the above, we see that the indicator functions of affine spaces of codimension k have sparsity 2^k. So for f computed by a depth d parity decision tree.

$$\mathsf{spar}(f) \leq 2^{2d}.$$

This gives us a simple lower bound.

Lower Bound 12: Sparsity

$\mathsf{D}^{\oplus}(f) \geq \frac{\log \mathsf{spar}(f)}{2}$.

Conjecture 3.3.4: The Log-Rank Conjecture for XOR functions

There exists a universal constant α such that for any total function f, we have that $\mathsf{D}^{\oplus}(f) \leq O(\log^{\alpha}(\mathsf{spar}(f)))$.

As its name may have implied, this conjecture has very intricate connections with the Log-Rank Conjecture. For any function $f : \{0,1\}^n \to \mathbb{R}$, we have that $\mathsf{spar}(f) = \mathsf{rank}(f \circ \mathsf{XOR})$ (see Theorem 3.5.2). We know that $2\mathsf{D}^{\oplus}(f) \geq \mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR})$ and Hatami, Hosseini and Lovett [HHL18] showed that $\mathsf{D}^{\oplus}(f) \leq \tilde{O}(\mathsf{D}^{\mathsf{cc}}(f \circ \mathsf{XOR})^6)$. Hence the above conjecture is equivalent to the Log-Rank Conjecture restricted to the class of XOR functions.

Another fundamental measure of functions is that of spectral norm.

Definition 3.3.5 (Spectral norm)**.** *Define the spectral norm of a function $f : \{0,1\}^n \to \mathbb{R}$ as follows.*

$$\left\|\widehat{f}\right\|_1 := \sum_{S\subseteq[n]} |\hat{f}(S)|.$$

From the Fourier representation of affine spaces given above, we see that the spectral norm of the indicator function of an affine space is 1. So a function f computed by a depth k PDT must satisfy $\left\|\widehat{f}\right\|_1 \leq 2^k$. More generally, a function f computed by a PDT with t leaves must satisfy $\left\|\widehat{f}\right\|_1 \leq t$. We denote this measure of the number of leaves required in a PDT computing f as $\mathsf{D}^{\oplus,\mathsf{leaf}}(f)$. This measure has been used before. In fact it was conjectured that any Boolean function with small $\left\|\widehat{f}\right\|_1$ can be written as a small sum/difference of indicator

functions of affine spaces [STV17]. The best known upper bound for this conjecture comes from constructing a parity decision tree with a small number of leaves.

It is easy to see that $\left\|\widehat{f}\right\|_1 \leq \sqrt{\mathsf{spar}(f)}$. To see so, one merely constructs the vector v of absolute values of non-zero Fourier coefficients of f, and the all-1 vector w of the same dimension as v. The dimension of these vectors is $\mathsf{spar}(f)$. The sum of the entries of v is $\left\|\widehat{f}\right\|_1$. The Cauchy-Schwarz inequality states that $\langle v, w\rangle^2 \leq \langle v, v\rangle\langle w, w\rangle$. This amounts to $\left\|\widehat{f}\right\|_1^2 \leq 1 \cdot \mathsf{spar}(f)$, since the sum of squares of Fourier coefficients is at most 1 for Boolean functions (Theorem 3.1.4).

As we will see in Section 3.5, it will turn out that $\left\|\widehat{f}\right\|_1 = \gamma_2(f \circ \mathsf{XOR})$. As $\log \gamma_2(F)$ was a lower bound for $\mathsf{P}^{\mathsf{EQcc}}(F)$, similarly we have seen that $\log \left\|\widehat{f}\right\|_1$ is a lower bound for $\log \mathsf{D}^{\oplus,\mathsf{leaf}}(f)$. However, there is another model of PDTs that is equivalent to $\log \mathsf{D}^{\oplus,\mathsf{leaf}}$, but is a better analogue of $\mathsf{P}^{\mathsf{EQcc}}$.

Let us strengthen the model of PDTs by allowing the decision algorithm to query affine spaces. A node labeled with an affine space S will direct the algorithm to its right child if $x \in S$, and to the left child otherwise. The minimum depth of a tree, over all affine space decision trees computing f, is denoted as $\mathsf{D}^{\wedge\oplus}(f)$.

Theorem 3.3.6. $\mathsf{D}^{\wedge\oplus}(f) \in \left[\Omega\left(\frac{\log \mathsf{D}^{\oplus,\mathsf{leaf}}(f)}{\log(n+1)}\right), O(\log \mathsf{D}^{\oplus,\mathsf{leaf}}(f))\right]$

Proof. Any depth d affine space decision tree can be converted to a parity decision tree with $n^{O(d)} = 2^{O(d \log n)}$ leaves. Simply replace each affine space query S with the parity decision tree T_S computing whether $x \in S$. T_S has co-dim(S) 0-leaves and one 1-leaf. Every 0-leaf of T_S is replaced with a copy of the left child of the S-query node, and the 1-leaf is replaced with the right child of the S-query node. Continue these substitutions until only parity queries are left in the tree. Any leaf ℓ of the new tree can be specified by specifying, for each node of the original tree, which of the at most $n+1$ leaves of the node's replacement tree should be taken to reach ℓ. This yields at most $(n+1)^d$ leaves.

One can also convert a parity decision tree with t leaves into an affine space decision tree of depth $O(\log t)$. This follows a well known technique of balancing trees [Spi71]. This technique is applicable when it is easy to test whether the input will pass through a specific node v. Since we start with parity decision trees, every node v corresponds to an affine space. An affine space query can easily check whether the input will pass through v.

The technique is simple. An unbalanced node is defined as a node whose right subtree has more than twice as many leaves as its left subtree, or vice versa. If such a node does not exist in the tree, then any node at depth d has at most a $(2/3)^d$ fraction of the tree's leaves under it, and so the depth of the tree is at most $\log_{3/2} t$. To balance the tree, we start at the root. We ensure that the root is balanced, and then recursively

balance the subtrees under it. In case the root v is unbalanced, we balance it as follows.

- Let t_v be the number of leaves in the subtree rooted at v.
- Starting from v, repeatedly move to the child that has a higher number of leaves under it, breaking ties arbitrarily. Stop when you reach a node w with between $t_v/3$ and $2t_v/3$ leaves under it. You will reach such a node since the number of leaves must reach 1 eventually and they decrease by at most a factor of $1/2$ each time.
- Replace the subtree rooted at v with the following subtree that computes the same function.
  - The root is the affine space query of whether the input reaches w.
  - The right child of the root is the subtree rooted at w.
  - The left child of the root is the subtree rooted at v, but with the subtree rooted at w removed. Note that this also makes w's parent useless, and it can be replaced with w's sibling.

This substitution leaves the number of leaves unchanged, yet the node v has been replaced with a root node that is balanced. The subtrees of the root node are made purely of parity queries, so we can recursively balance those subtrees as well. □

3.3.2 Parity Kill Number

Another measure of interest is the parity kill number, defined as follows.

Definition 3.3.7 (Parity Kill Number)**.** *The* parity kill number *of f is defined as*

$$\mathsf{C}^{\oplus}_{\mathsf{min}}(f) \triangleq \min\{\textit{co-dim}(S) | S \textit{ is an affine subspace on which } f \textit{ is constant}\}.$$

It is clear that if T is a parity decision tree computing f, then every leaf of T must be at depth at least $\mathsf{C}^{\oplus}_{\mathsf{min}}(f)$. On the other hand, by querying $\mathsf{C}^{\oplus}_{\mathsf{min}}(f)$ parities, one can reduce the problem to one with at most half the original sparsity. (See Theorem 3.3.8 below.) If one can show that there is a constant c such that for every f, $\mathsf{C}^{\oplus}_{\mathsf{min}}(f) \leq \log^c \mathsf{spar}(f)$, this would imply that $\mathsf{D}^{\oplus}(f) \leq \log^{c+1} \mathsf{spar}(f)$, and it would be equivalent to proving Conjecture 3.3.4.

Theorem 3.3.8 ($\mathsf{C}^{\oplus}_{\mathsf{min}}$ Queries Reduces Sparsity [STV17])**.** *Let f be monochromatic on the affine space defined by the constraints $\langle w_1, x\rangle = a_1, \langle w_2, x\rangle = a_2, \ldots, \langle w_k, x\rangle = a_k$. Then for any $b_1, b_2, \ldots, b_k$, f restricted to the affine space $\langle w_1, x\rangle = b_1, \langle w_2, x\rangle = b_2, \ldots, \langle w_k, x\rangle = b_k$ has sparsity at most $\mathsf{spar}(f)/2$.*

Proof. Consider the function f' defined as the restriction of f to the affine space defined by the constraints $\langle w_1, x\rangle = b_1, \langle w_2, x\rangle = b_2, \ldots, \langle w_k, x\rangle = b_k$. f' is a Boolean function whose domain is an $n-k$-dimensional space. Let $v_1, \ldots, v_{n-k}$ be an extension of the w's to a basis of the whole n-dimensional space. The Fourier basis of the space of functions on an $n-k$-dimensional space spanned by $v_1, \ldots, v_{n-k}$ is the set of parity functions of the form $\{\chi_v\}$ where v is in the span of $v_1, \ldots, v_{n-k}$. Each $\chi_S(x)$ can be written as $\chi_v(x)\chi_w(x)$ for some v and w in the spans of $v_1, \ldots, v_{n-k}$ and $w_1, \ldots, w_k$ respectively. Hence

$$\begin{aligned} f(x) &= \sum_S \hat{f}(S)\chi_S(x) \\ &= \sum_v \sum_w \hat{f}(v+w)\chi_v(x)\chi_w(x) \\ &= \sum_v \left(\sum_w \hat{f}(v+w)\chi_w(x) \right) \chi_v(x) \end{aligned}$$

and $\hat{f'}(v) = \sum_w \hat{f}(v+w)\chi_w(x)$, where $\chi_w(x)$ is a constant depending only on $b_1, \ldots, b_k$. We can thus see that the Fourier coefficients of f' (let us refer to them as the "new" coefficients) are linear combinations of Fourier coefficients of f (the "old" coefficients). No "old" coefficient appears in the linear combinations of two different "new" coefficients. We know the Fourier spectrum of f when restricted to the affine subspace $\langle w_1, x\rangle = a_1, \langle w_2, x\rangle = a_2, \ldots, \langle w_k, x\rangle = a_k$. It has at most one non-zero Fourier coefficient, whose value is 1 if it exists. This means that for every non-zero "old" coefficient, there must have been at least one other non-zero "old" coefficient that combined with it in order to change the value of the "new" coefficient to either 0 or 1.
Note that which "old" coefficients combine together does not depend on the values of $a_1, \ldots, a_k$. So even in the function f', every non-zero "old" coefficient combines with another non-zero "old" coefficient. Hence the number of non-zero "new" coefficients is at most half the number of "old" coefficients. □

Given the above theorem, a PDT of depth $(\max_{\text{affine subspace } A} \mathsf{C}^{\oplus}_{\mathsf{min}}(f|_A)) \cdot \log \mathsf{spar}(f)$ follows.

> The upper bound in terms of nonnegative rank, when applied to a function $f \circ \mathsf{XOR}$, yields a protocol much like the one above.

The best known upper bound on $\mathsf{C}^{\oplus}_{\mathsf{min}}(f)$ is $\mathsf{C}^{\oplus}_{\mathsf{min}}(f) \leq O(\left\|\widehat{f}\right\|_1)$ [TWXZ13]. Since $\left\|\widehat{f}\right\|_1 \leq \mathsf{spar}(f)$, an upper bound of $\log^{O(1)} \left\|\widehat{f}\right\|_1$ would imply the Log-Rank Conjecture for XOR functions.

Conjecture 3.3.9: The Parity Kill Number Conjecture [TWXZ13]

$\mathsf{C}^{\oplus}_{\mathsf{min}}(f) \leq \log^{O(1)} \left\|\widehat{f}\right\|_1$.

This conjecture has been proven false in this thesis in Chapter 4.

3.3.3 Randomized

We now move on to randomized parity decision trees. Recall that a cost-c randomized parity decision tree is a distribution on depth c binary trees, each of which having, among others, the following properties.

- Each leaf corresponds to an affine space of codimension at most c.
- The affine spaces at the leaves partition the whole input space.
- The output of the protocol on any input is the label of the leaf that the input lands in.

The probability of a randomized parity decision tree outputting a 1 is a convex combination of the outputs of the constituent deterministic parity decision trees. Since each cost-k deterministic parity decision tree computes a function f with $\left\|\widehat{f}\right\|_1 \leq 2^k$, it follows that the probability of an RPDT outputting a 1 is also computed by a function of spectral norm at most 2^k.

If $f : \{0,1\}^n \to \{0,1\}$ is computed by a cost-k randomized parity decision tree with error $\leq \epsilon$, then the probability of outputting a 1 on an input z should also be within ϵ of $f(z)$. Hence f is pointwise close to a function with spectral norm $\leq 2^k$. This gives us a lower bound in terms of the approximate spectral norm.

Definition 3.3.10 (Approximate Spectral Norm)**.** *For a function f the ϵ-approximate spectral norm of F, which we denote as $\left\|\widehat{f}\right\|_{1,\epsilon}$, is defined as*

$$\min_{g:||f-g||_\infty \leq \epsilon} \left\|\widehat{g}\right\|_1 .$$

Lower Bound 13: Approximate Spectral Norm

$\mathsf{R}^{\oplus}_{\epsilon}(f) \geq \log \left\|\widehat{f}\right\|_{1,\epsilon}$.

Note that a similar lower bound does not immediately follow for approximate sparsity. In fact logarithm of the approximate sparsity is *not* a lower bound for RPDTs. For instance, take the AND function on n bits. It is well known to have a constant size randomized parity decision tree involving random hashes. On the other hand, it has approximate sparsity at

least $\Omega(n)$. (This can be derived from Theorem 3.2.15 and the fact that the $\mathsf{AND} \circ \mathsf{XOR}$ function, equivalent to the equality function, has deterministic communication complexity $n+1$.) However, it turns out that $\Omega(\log \mathsf{spar}_{1/3}(f) - \log n)$ is a lower bound on the randomized parity decision tree complexity. The proof of this is via Theorem 3.4.2, which shows that approximate sparsity and approximate spectral norm are related.

> A reason why approximate sparsity is not by itself a lower bound, unlike approximate rank, is that approximate rank only gives a lower bound for *private* coin communication protocols. Randomized PDTs do not have a notion of privacy in their random coins, so it should be the public coin lower bounds in communication complexity that have analogues here.

The weakly unbounded error PDT complexity of a function f is defined analogously to the corresponding communication complexity measure: $\mathsf{R}^{\oplus}_{\mathsf{weak},<\frac{1}{2}}(F)$ is the minimum, over all randomized PDTs that output the correct answer with probability $\geq \frac{1}{2} + \delta$ on every input, of the depth of the tree plus $\log(1/\delta)$.

We define the margin complexity of f as an approximate spectral norm measure, the way γ_2^∞ is defined given γ_2. We now think of f as a function outputting 1 or -1.

Definition 3.3.11 (Polynomial Margin)**.** *For a function $f : \{0,1\}^n \to \{-1,1\}$, $\mathsf{margin}(f)$ is defined as $\min_{g:1\leq g(z)f(z)} \|\widehat{g}\|_1$. That is, it is the minimum sum of the absolute values of coefficients of any real polynomial g that computes f in the sense that $f(z)g(z) \geq 1$ for all z.*

> The polynomial margin of f is actually the inverse of the minimum margin that a sign-representing polynomial for f of *unit* weight (sum of absolute values of coefficients) can achieve. It is easy to see that the above definition gives the same quantity.

Theorem 3.3.12. $\log(\mathsf{margin}(f)) + 2 \geq \mathsf{R}^{\oplus}_{\mathsf{weak},<\frac{1}{2}}(f) \geq \log(\mathsf{margin}(f)) - 1$.

Proof. Given a $g : \{0,1\}^n \to \mathbb{R}$ that minimizes the expression in the definition of polynomial margin, we can come up with an efficient RPDT. Sample $S \subseteq [n]$ according to the distribution that is proportional to $|\widehat{g}(S)|$. Output $\chi_S(z) sgn(\widehat{g}(S))$. The expected value of the output is

$$\sum_S \frac{\widehat{g}(S)}{\|\widehat{g}\|_1} \chi_S(z) = \frac{g(z)}{\|\widehat{g}\|_1}.$$

So on every input, the correct answer is output with probability $\geq \frac{1}{2} + \frac{1}{2\|\widehat{g}\|_1}$, and this protocol has a weakly unbounded error cost of $1 + \log(2\|\widehat{g}\|_1) \leq 2 + \log(\mathsf{margin}(f))$.

The lower bound on $\mathsf{R}^{\oplus}_{\mathsf{weak},<\frac{1}{2}}(f)$ follows from the statement $\mathsf{D}^{\oplus}(f) \geq \log \left\|\widehat{f}\right\|_1$ in exactly the same way as the lower bound on $\mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}$ follows from the statement $\mathsf{D}^{\mathsf{cc}}(F) \geq \log \gamma_2(F)$ (see proof of Theorem 3.2.27 and its preceding analysis). □

★ 3.4 A Bridge Between Counting and Weighing

Here we present various theorems connecting measures that we have seen. Of them, the first four are of a very similar nature. Indeed, their proofs share the same framework. This chapter merely reproduces these known proofs but phrased in this new framework. The four are as follows. (Query functions are on n bits and communication functions are on $n+n$ bits.)

- Grolmusz's Theorem: $\mathsf{spar}_\delta(f) \leq O(\left\|\widehat{f}\right\|_{1,\epsilon}^2 n/(\delta-\epsilon)^2)$
- Newman's Theorem: $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},\delta}(F) \leq \mathsf{R}^{\mathsf{cc}}_{\epsilon}(F) + \lceil \log(2n) + 2\log(1/(\delta-\epsilon))\rceil$
- $\mathsf{rank}_\delta(F) \leq O(\gamma_2^{1/(1-\epsilon)}(F)^2(1-\epsilon)^2 n/(\delta-\epsilon)^2)$
- $\mathsf{rank}^+_\delta(F) \leq O(\mathsf{srect}^1_\epsilon(F)^2 n/(\delta-\epsilon)^2)$

We will also see a fifth connection, stating that $\mathsf{srect}^1_{2\delta}(F) \leq \mathsf{rank}^+_\delta(F)$.

The former four follow the following format. On the left hand size of the inequality, we want to bound the complexity of a target function by the minimum "count" of simple functions needed to approximate the target function (the relevant measure in Newman's theorem will be clear from the proof). On the right hand side we have that our target function is approximated as a low-weight combination of the simple functions. Then we follow the following steps.

1. This low-weight combination of the simple functions allows us to approximate the target function as a distribution over simple functions.
2. We then use Hoeffding's Lemma to say that for any for any fixed input x, with (very) high probability the average of a few samples from the distribution should approximate the target function. Since we are representing it with only a few samples, this means that the "count" of simple functions being used is small. (To get the optimal parameters in the theorem relating γ_2^α and rank_δ, we will have to resort to a different concentration bound that does not assume boundedness.)
3. We finally use a union bound to say that with non-zero probability the average of a few samples from the previous step actually approximates the function at *all* points.

Lemma 3.4.1 (The "Essentially Hoeffding" Framework for Low-Count Approximations)**.** *Let $\delta > \epsilon \geq 0$ and $f : \{0,1\}^n \to \{0,1\}$ be ϵ-approximated by a real function $g : \{0,1\}^n \to \mathbb{R}$. Let $\mathcal{H}$ be a distribution over a set of functions such that for all x, $g(x) = \mathbb{E}_{\mathbf{h}\sim\mathcal{H}}[\mathbf{h}(x)]$, with each h bounded in $[-r,r]$. Then there is a $g' : \{0,1\}^n \to \mathbb{R}$ that δ-approximates f such that $g' = \frac{1}{t}\sum_{i\in[t]} h_i$, where $t \leq O(r^2 n/(\delta-\epsilon)^2)$.*

Proof. Fix an $x \in \{0,1\}^n$. Let us take t samples $h_1, h_2, \ldots, h_t$ from the distribution of $\mathbf{h}$. Hoeffding's Lemma (Lemma 3.1.1) says that $\Pr_{h_1,\ldots,h_t}[|\,\mathbb{E}_{\mathbf{h}}[\mathbf{h}(x)] - \frac{1}{t}\sum_{i\in[t]} h_i| \geq \delta - \epsilon] \leq 2\exp(-\frac{2(\delta-\epsilon)^2 t^2}{4tr^2})$.

We can get this probability below 2^{-n} if we set $t = O(r^2 n/(\delta-\epsilon)^2)$. Then by a union bound, we get that with probability greater than 0,

$$\forall x \in \{0,1\}^n \; |\mathop{\mathbb{E}}_{\mathbf{h}}[\mathbf{h}(x)] - \frac{1}{t}\sum_{i\in[t]} h_i| \leq \delta - \epsilon$$

and hence $\frac{1}{t}\sum_{i\in[t]} h_i$ is a $(\delta-\epsilon)$-approximation to g and a δ-approximation to f. □

We now prove the four connections using the above framework.

Grolmusz's Theorem had its roots in a paper by Bruck and Smolensky [BS90], and was strengthened by Grolmusz [Gro97] and Zhang [Zha14]. We write it in slightly more generality here.

Theorem 3.4.2 (Grolmusz's Theorem [BS90, Gro97, Zha14])**.** *For any $f : \{0,1\}^n \to \{0,1\}$ and $\delta > \epsilon \geq 0$,*

$$\mathsf{spar}_\delta(f) \leq O(\left\|\widehat{f}\right\|_{1,\epsilon}^2 n/(\delta-\epsilon)^2).$$

Proof. Let $g : \{0,1\}^N \to \mathbb{R}$ be a function witnessing $\left\|\widehat{f}\right\|_{1,\epsilon}$. Consider the functions $h_\alpha(x) = \|\widehat{g}\|_1 \operatorname{sign}(\widehat{g}(\alpha))\chi_\alpha(x)$, with the distribution over them being the following: h_α is chosen with probability $|\widehat{g}(\alpha)|/\|\widehat{g}\|_1$.

Clearly each h_α is bounded in $[-\|\widehat{g}\|_1, \|\widehat{g}\|_1]$. Furthermore, $\mathbb{E}_{\mathbf{h}}[h_\alpha(x)] = \sum \widehat{g}(\alpha)\chi_\alpha(x) = g(x)$.

Using Lemma 3.4.1, we get a function g' that is a sum of at most $O(\|\widehat{g}\|_1^2 n/(\delta-\epsilon)^2)$ monomials that δ-approximates f. Since $\left\|\widehat{f}\right\|_{1,\epsilon} = \|\widehat{g}\|_1$, the theorem follows. □

We now see Newman's Theorem.

Theorem 3.4.3 (Newman's Theorem [New91])**.** *Let $\delta > \epsilon \geq 0$ and $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ such that $\mathsf{R}^{\mathsf{cc}}_\epsilon(F) = c$. Then $\mathsf{R}^{\mathsf{cc}}_{\mathsf{pri},\delta}(F) \leq c + \lceil \log(2n) + 2\log(1/(\delta-\epsilon))\rceil$.*

Proof. Let Π be a cost-c protocol that computes F to within ϵ error, and let $p : \{0,1\}^n \times \{0,1\}^n \to \mathbb{R}$ be the function outputting the probability that Π outputs 1 on an input. $p\epsilon$-approximates F.

Π gives a distribution $\mathcal{D}$ over cost-c deterministic protocols such that $\mathbb{E}_{D\sim\mathcal{D}}[D(x,y)] = p(x,y)$. Each D is a function outputting values in $\{0,1\}$.

Hence Lemma 3.4.1 says that there is a function p' that $\delta-\epsilon$-approximates p such that p' is the average of $t = O(2n/(\delta-\epsilon)^2)$ cost-c deterministic protocols $h_1, \ldots, h_t$.

To end the proof, we note that there is a private coin randomized protocol Π' that has acceptance probability p'. Alice samples a random number $i \in [t]$ and sends it to Bob. Alice and Bob then run the deterministic protocol h_i. It is easy to see that it has acceptance probability p' and hence error $\leq \delta$, and the cost of the protocol is $\leq c + \log(2n) + 2\log(1/(\delta-\epsilon))$.

Note that this applies to more than just randomized communication complexity. For instance, it also provides a way of converting an efficient quantum protocol that uses public

randomness (and not entanglement) into an efficient quantum protocol in which the parties do not share any randomness/entanglement beforehand. □

Our next theorem shows that the approximate γ_2 norm cannot be much smaller than approximate rank, although the exact versions could be *far* smaller. This was proven by Lee and Shraibman [LS09a], but the proof we give here goes via the proof of the Johnson-Lindenstrauss Theorem [JL01] as proven by Indyk and Motwani [IM98].

Theorem 3.4.4. *[[LS09a]] Let $\delta > \epsilon \geq 0$ and $F : \{0,1\}^n \times \{0,1\}^n \to \{-1,1\}$ such that $\gamma_2^{1/(1-\epsilon)}(F) = c$. Then $\mathsf{rank}_\delta(F) \leq O(c^2(1-\epsilon)^2 n/(\delta-\epsilon)^2)$.*

Proof. The assumption in the theorem gives us a matrix representing F with certain properties as given in the definition of γ_2^α. We scale it by $1-\epsilon$ to get a matrix $M \in \mathbb{R}^{2^n \times 2^n}$ with values in $[-1,-1+\epsilon] \cup [1-\epsilon, 1]$ that sign agrees with the matrix of F and has the following property. There exists a natural number $d \in \mathbb{N}$, vectors $\{\phi_x\}_{x\in\{0,1\}^n}$ and $\{\psi_y\}_{y\in\{0,1\}^n}$, all in $\mathbb{R}^d$, such that $M[x,y] = \langle \phi_x, \psi_y \rangle$ and $\max_x \|\phi_x\| = \max_y \|\psi_y\| = \sqrt{c(1-\epsilon)}$.

This already gives us $\mathsf{rank}_\epsilon(F) \leq d$, but that could be very large. Instead we can use the following distribution of rank-1 matrices.

Sample a random vector $\mathbf{v} \in \mathbb{R}^d$ by sampling each coordinate to be uniformly at random from $\{-1,1\}$. Construct the vector $\mathbf{v}_\mathcal{X} \in \mathbb{R}^{2^n}$ by setting the xth coordinate to be $\langle \mathbf{v}, \phi_x \rangle$. Similarly construct the vector $\mathbf{v}_\mathcal{Y} \in \mathbb{R}^{2^n}$ by setting the yth coordinate to be $\langle \mathbf{v}, \psi_y \rangle$. Construct the rank-1 matrix $\mathbf{h}$ as the outer product of $\mathbf{v}_\mathcal{X}$ and $\mathbf{v}_\mathcal{Y}$. This matrix can be seen as a function from $\{0,1\}^n \times \{0,1\}^n$ to $\mathbb{R}$.

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{h}}[h[x,y]] &= \mathop{\mathbb{E}}_{\mathbf{v}}\left[\left(\sum_{i\in[d]} v_i\phi_{x,i}\right)\left(\sum_{i\in[d]} v_i\psi_{y,i}\right)\right] \\
&= \sum_{i\in[d]} \mathop{\mathbb{E}}_{v}[v_i^2\phi_{x,i}\psi_{y,i}] + \sum_{i\neq j\in[d]} \mathop{\mathbb{E}}_{v}[v_iv_j\phi_{x,i}\psi_{y,j}] \\
&= \sum_{i\in[d]} \phi_{x,i}\psi_{y,i} = \langle \phi_x, \psi_y \rangle = M[x,y]
\end{aligned}$$

Each entry of h is bounded in $[-c^2(1-\epsilon)^2, c^2(1-\epsilon)^2]$, since $\langle \mathbf{v}, \phi_x \rangle \leq \|\phi_x\|_1 \leq \|\phi_x\|_2^2 \leq c(1-\epsilon)$ and similarly $\langle \mathbf{v}, \psi_y \rangle \leq c(1-\epsilon)$. Using Lemma 3.4.1, we can conclude that there is a matrix M' that δ-approximates F with rank at most $t = O(c^4(1-\epsilon)^4 n/(\delta-\epsilon)^2)$.

While the above is a nice result, it can be made tighter. Our function h was bounded in $[-c^2, c^2]$. However it has a very thin tail and is concentrated within $[-O(c), O(c)]$. Using some better analysis, we will see that the final bound can be improved to $O(c^2(1-\epsilon)^2 n/(\delta-\epsilon)^2)$.

Towards this, it is useful to make each entry of $\mathbf{v}$ an independent normal random variable with mean 0 and variance 1 (as done by [IM98]). Note that $\mathbb{E}[v_i^2] = 1$ and v_i is distributed identically to $-v_i$, so this change of random variable does not change the expectation of $\mathbf{h}$.

Also observe that $\mathbf{h}[x, y] = \langle \mathbf{v}, \phi_x \rangle \langle \mathbf{v}, \psi_y \rangle$ can be rewritten as

$$\frac{\langle \mathbf{v}, \phi_x + \psi_y \rangle \langle \mathbf{v}, \phi_x + \psi_y \rangle - \langle \mathbf{v}, \phi_x \rangle \langle \mathbf{v}, \phi_x \rangle - \langle \mathbf{v}, \psi_y \rangle \langle \mathbf{v}, \psi_y \rangle}{2}.$$

Such a rewriting is useful for us because of the simple random variables that arise from it. $\langle \mathbf{v}, \phi_x \rangle$ is a sum of weighted normal distributions and is hence a normal distribution of mean 0 and variance $\phi_{x,1}^2 + \phi_{x,2}^2 + \cdots + \phi_{x,d}^2 = \|\phi_x\|_2^2$. It can hence be thought of as $\|\phi_x\|_2 \cdot \mathcal{N}(0, 1)$.

Thus $\langle \mathbf{v}, \phi_x \rangle \langle \mathbf{v}, \phi_x \rangle$ is distributed as $\|\phi_x\|_2^2 \cdot \mathcal{N}(0, 1)^2$. The sum of squares of k independent standard (i.e. mean 0, variance 1) normal distributions is called a χ^2 distribution with k degrees of freedom. It has a strong concentration bound (see [LM00]), namely for such a random variable X and $\gamma \leq 1$,

$$\Pr\left[X \notin k(1 \pm \gamma)\right] \leq \exp(-O(k\gamma^2)).$$

This replaces Hoeffding for us, and we see that

$$\Pr\left[\frac{1}{t}\sum_{i \in [t]} \langle \mathbf{v_i}, \phi_x \rangle^2 \notin \|\phi_x\|_2^2 (1 \pm \gamma)\right] \leq \exp(-O(t\gamma^2)).$$

Let $\mathbf{v^{(1)}}, \ldots, \mathbf{v^{(t)}}$ be random variables distributed as per the distribution of $\mathbf{v}$ above. Setting $\gamma = \frac{(\delta - \epsilon)}{3c(1-\epsilon)}$ and $t = O(\frac{c^2(1-\epsilon)^2 n}{(\delta-\epsilon)^2})$, and observing that $\|\phi_x\|_2^2 \leq c(1 - \epsilon)$, we can conclude that

$$\Pr\left[\frac{1}{t}\sum_{i \in [t]} \langle \mathbf{v^{(i)}}, \phi_x \rangle^2 \notin \|\phi_x\|_2^2 \pm (\delta - \epsilon)/3\right] < 2^{-2n-1}. \tag{3.1}$$

We have similar statements for ψ_y and $\phi_x + \psi_y$. A simple application of the union bound shows the existence of $v^{(1)}, \ldots, v^{(t)}$ such that the above statement is simultaneously satisfied for all ϕ_x, all ψ_y and all $\phi_x + \psi_y$ (there are at most $2^n + 2^n + 2^{2n}$ statements).

This suffices for our theorem, since this implies that for all x, y,

$$\begin{aligned}
\left|\frac{1}{t}\sum_{i \in [t]} \langle v^{(i)}, \phi_x \rangle \langle v^{(i)}, \psi_y \rangle - \langle \phi_x, \psi_y \rangle\right| &\leq \left|\frac{1}{t}\sum_{i \in [t]} \langle v^{(i)}, \phi_x \rangle^2 - \|\phi_x\|_2^2\right|/2 \\
&+ \left|\frac{1}{t}\sum_{i \in [t]} \langle v^{(i)}, \psi_y \rangle^2 - \|\psi_y\|_2^2\right|/2 \\
&+ \left|\frac{1}{t}\sum_{i \in [t]} \langle v^{(i)}, \phi_x + \psi_y \rangle^2 - \|\phi_x + \psi_y\|_2^2\right|/2 \\
&\leq \frac{(\delta - \epsilon)/3 + (\delta - \epsilon)/3 + (\delta - \epsilon)/3}{2} < (\delta - \epsilon)
\end{aligned}$$

and so the δ-approximate rank is at most t. □

We now move on to comparing approximate nonnegative rank and the smooth rectangle bound.

Theorem 3.4.5 ([KMSY14]). *For any $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ and $\delta > \epsilon \geq 0$, $\mathsf{rank}^+_\delta(F) \leq O(\mathsf{srect}^1_\epsilon(F)^2 n/(\delta-\epsilon)^2)$*

Proof. Let $\mathsf{srect}^1_\epsilon(F) = c$. Recall from the definition that this implies the existence of a set of rectangles $\mathcal{R}$ with nonnegative weights $\{w_R\}_{R\in\mathcal{R}}$ such that

- $\sum_R w_R = c$,
- for all $x, y \in F^{-1}(1)$, $\sum_{R \ni (x,y)} w_R \in [1-\epsilon, 1]$ and
- for all $x, y \in F^{-1}(0)$, $\sum_{R \ni (x,y)} w_R \in [0, \epsilon]$.

Hence the function $p(x,y) = \sum_{R\in\mathcal{R}} w_R R(x,y)$ is an ϵ-pointwise approximation of F where the function R is the indicator function of R. Each function $h_R(x,y) = c \cdot R(x,y)$ is bounded in $[0,c]$. Let us consider the distribution over the functions $\{h_R\}_{R\in\mathcal{R}}$ that chooses h_R with probability w_R/c. Then $\mathbb{E}_{\mathbf{h}}[\mathbf{h}(x,y)] = \sum w_R R(x,y) = p(x,y)$.

Using Lemma 3.4.1, the proof is complete. □

The above shows that the approximate nonnegative rank is not much larger than the smooth rectangle bound. We finish this section with the result that it is not much smaller either.

Theorem 3.4.6 ([KMSY14]). *For any $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ and $\delta > 0$, $\mathsf{srect}^1_{2\delta}(F) \leq \mathsf{rank}^+_\delta(F)$*

Proof. Let $\mathsf{rank}^+_\delta(F) = c$. This means that there is a matrix M that is δ-close to F, and a decomposition of M as a sum of at most c nonnegative rank 1 matrices, $M_1, M_2, \ldots, M_c$.

Central to the proof is the claim that any nonnegative rank 1 matrix whose maximum entry is at most 1 can be written as a subconvex combination of rectangles. Note that by the definition of nonnegative rank, a matrix M has nonnegative rank 1 if and only if there exist vectors u and v such that $M = u \otimes v$.

Claim 3.4.7. *Let u and v be nonnegative vectors in $\mathbb{R}^n$ such that $u \otimes v$ has its maximum entry at most 1. Then there exist rectangles $R_1, R_2, \ldots, R_d$ with nonnegative weights $w_1, w_2, \ldots, w_d$ such that $u \otimes v = \sum_{i=1}^d w_i R_i$ and $\sum_{i=1}^d w_i \leq 1$.*

Proof. Since $\max_j u_j \cdot \max_j v_j$ is at most 1, u and v can be appropriately scaled without changing $u \otimes v$ to also ensure that $\max u_j$ and $\max v_j$ are at most 1.

Let the entries of u be, in increasing order, $r_1, r_2, \ldots, r_k$. Define sets $A_1, A_2, \ldots, A_k \subseteq [n]$ as $A_i \triangleq \{j \in [n] | u_j \geq r_i\}$. Similarly define $s_1, \ldots, s_{k'}$ to be the entries of v and the sets $B_1, \ldots, B'_k$ are defined analogously. Note that $\sum_{i\in[k]} (r_i - r_{i-1}) \mathbb{1}_{A_i} = u$, where $r_0 = 0$.

We can now represent $u \otimes v$ as

$$\left(\sum_{i\in[k]}(r_i - r_{i-1})\mathbb{1}_{A_i}\right) \otimes \left(\sum_{i\in[k']}(s_i - s_{i-1})\mathbb{1}_{B_i}\right) = \sum_{(i,j)\in[k]\times[k']}(r_i - r_{i-1})(s_j - s_{j-1})R_{i,j}$$

where $R_{i,j} = A_i \times B_j$. The sum of the weights of the rectangles is $\sum_{i\in[k]}(r_i - r_{i-1})\sum_{j\in[k']}(s_j - s_{j-1}) = r_k s_{k'} = \max_i u_i \max_i v_i \leq 1$. □

Let M be scaled by $1/(1+\delta)$ so that its maximum entry is at most 1. It now approximates F to error $1 - (1-\delta)/(1+\delta) < 2\delta$ and each M_i has their maximum entry at most 1.

We can now use Claim 3.4.7 to write each M_i as a sum of weighted rectangles, with the weights adding up to at most 1. Adding up these for each M_i, we represent M as a sum of weighted rectangles, with the sum of the weights being at most c. This concludes the proof of the theorem.

Note that if approximate nonnegative rank also imposed that the entries of the approximating matrix be bounded in $[-1, 1]$, then we would have no loss in the error parameter. □

3.5 Lifting from Polynomial Measures to Matrix Measures

In this section, we will see that many query measures that we have seen for f lift to communication measures for $f \circ \mathsf{XOR}$.

The first two measures that we will look at are sparsity and spectral norm, which lift to rank and trace norm, the latter defined as follows.

Definition 3.5.1. *The trace norm of F, denoted $\|F\|_{tr}$, is the sum of the singular values of the communication matrix of F. It can be defined as $\|F\|_{tr} = \mathsf{trace}\left(\sqrt{MM^T}\right)$, where M is the communication matrix of F.*[8]

We will not be proving that the trace norm is a norm. We prove the following.

Theorem 3.5.2.

1. $\mathsf{rank}(f \circ \mathsf{XOR}) = \mathsf{spar}(f)$.
2. $\gamma_2(f \circ \mathsf{XOR}) = \|f \circ \mathsf{XOR}\|_{tr} / 2^n = \left\|\widehat{f}\right\|_1$.

Proof. It is known using dual norms (see [LMSS07], section 3) that $\gamma_2(M) \geq \|M\|_{tr} / 2^n$ for all $2^n \times 2^n$ matrices M. It is also easy to see that $\gamma_2(f \circ \mathsf{XOR}) \leq \left\|\widehat{f}\right\|_1$, since γ_2 is a norm and $\gamma_2(g \circ \mathsf{XOR}) = 1$ for any monomial g. Hence $\gamma_2(f \circ \mathsf{XOR})$ is sandwiched between $\|f \circ \mathsf{XOR}\|_{tr} / 2^n$ and $\left\|\widehat{f}\right\|_1$. If we show that $\|f \circ \mathsf{XOR}\|_{tr} / 2^n = \left\|\widehat{f}\right\|_1$, this would prove

[8] Since MM^T is a real symmetric positive semidefinite matrix, this is the same as the sum of the entries of the diagonal matrix D obtained by diagonalizing MM^T.

that all three terms are equal. Both $\|f \circ \mathsf{XOR}\|_{tr} / 2^n = \left\|\widehat{f}\right\|_1$ and $\mathsf{rank}(f \circ \mathsf{XOR}) = \mathsf{spar}(f)$ follow from Theorem 3.5.3. □

Theorem 3.5.3 (Folklore)**.** *Let $f : \{0,1\}^n \to \mathbb{R}$, and $F \triangleq f \circ \mathsf{XOR}$. For each $S \subseteq [n]$ the vector v_S, which is the "truth table" of χ_S, is an eigenvector of the communication matrix of F with eigenvalue $2^n \hat{f}(S)$.*

Proof. Let v_S be the vector in $\{-1,1\}^n$ defined as $v_S[x] = \chi_S(x)$. Let M be the communication matrix of F. Note that

$$\begin{aligned}
(Mv_S)[x] &= \sum_{y \in \{0,1\}^n} M[x,y] v_S[y] \\
&= \sum_{y \in \{0,1\}^n} f(x \oplus y) \chi_S(y) \\
&= \sum_{y \in \{0,1\}^n} \left(\sum_{T \subseteq [n]} \hat{f}(T) \chi_T(x) \chi_T(y) \right) \chi_S(y) \\
&= \sum_{T \subseteq [n]} \chi_T(x) \hat{f}(T) \left(\sum_{y \in \{0,1\}^n} \chi_T(y) \chi_S(y) \right) \\
&= \chi_S(x) \hat{f}(S) 2^n + \sum_{T \neq S} \chi_T(x) \hat{f}(T) \cdot 0.
\end{aligned}$$

Since this gives us 2^n orthogonal eigenvectors, this completely defines the spectrum of the eigenvalues of the matrix. □

When considering approximate measures, one might run into some trouble. Let the function g be the most sparse function that is pointwise ϵ-close to f. So $\mathsf{spar}_\epsilon(f) = \mathsf{spar}(g)$. Therefore $g \circ \mathsf{XOR}$ would be pointwise close to $f \circ \mathsf{XOR}$ and we can conclude that $\mathsf{rank}_\epsilon(f \circ \mathsf{XOR}) \leq \mathsf{rank}(g \circ \mathsf{XOR}) = \mathsf{spar}(g)$. However there may be another matrix M that is close to the matrix of $f \circ \mathsf{XOR}$ with even less rank so the approximate rank of $f \circ \mathsf{XOR}$ might actually be smaller than the approximate sparsity of f. We show, however, that the lifting theorem is *approximately* true. To prove this, we first use the fact that the lifting theorem for approximate spectral norm is *exactly* true, by which we mean the following theorem, a proof of which is outlined by Lee and Shraibman [LS09b, Remark 7.2]. In the following, $\gamma_{2,\epsilon}(M)$ refers to the minimum γ_2 norm among matrices that are entrywise ϵ-close to M.

Theorem 3.5.4. *Let $f : \{0,1\}^n \to \mathbb{R}$. Let M be a matrix that is pointwise ϵ-close to the matrix of $f \circ \mathsf{XOR}$. There exists a function g that is pointwise ϵ-close to f such that $\|g \circ \mathsf{XOR}\|_{tr} \leq \|M\|_{tr}$ and $\gamma_2(g \circ \mathsf{XOR}) \leq \gamma_2(M)$. Hence $\left\|\widehat{f}\right\|_{1,\epsilon} = \|f \circ \mathsf{XOR}\|_{tr,\epsilon} / 2^n = \gamma_{2,\epsilon}(f \circ \mathsf{XOR})$.*

Proof. Let us create the matrix $M_{sym}[x,y] = \frac{1}{2^n}\sum_{z\in\{0,1\}^n} M[x\oplus z, y\oplus z]$. Consider a term in the summation corresponding to z. Denoting this matrix as M_z, we can see that M_z is M but with an involution (a permutation that is its own inverse) applied to the rows and columns (row x maps to row $x\oplus z$ and similarly for the columns). Hence, $M_z = P_z M P_z$ (note that $P_z = P_z^{-1} = P_z^T$). Now $\|M_z\|_{tr} = \mathsf{trace}\left(\sqrt{M_z M_z^T}\right)$. However, if MM^T is diagonalized as VDV^{-1}, then note that $M_z M_z^T$ is diagonalized as $(P_zV)D(P_zV)^{-1}$. Hence $\|M_z\|_{tr} = \|M\|_{tr}$, and since it is a norm, $\|M_{sym}\|_{tr} \le \|M\|_{tr}$.
It is even easier to see that $\gamma_2(M_z) = \gamma_2(M)$, since rearranging rows and columns does not change the maximum row norm or the maximum column norm. Again, since γ_2 is a norm, $\gamma_2(M_{sym}) \le \gamma_2(M)$.
$M_{sym}[x,y] = \frac{1}{2^n}\sum_{(x',y') \text{ s.t. } x'\oplus y' = x\oplus y} M[x',y']$. Since each of the $M[x',y']$ in the sum is ϵ-close to $F(x\oplus y) = f(z)$, so is $M_{sym}[x,y]$. Since all (x,y) such that $x\oplus y = z$ have the same $M_{sym}[x,y]$, M_{sym} is actually $g\circ\mathsf{XOR}$ for some g that is ϵ-close to f. □

Note that the above proof would also work when the notion of approximation is of the type used in the definition of γ_2^α (Definition 3.2.26). As a direct consequence, we get the following *exact* relation between polynomial margin (defined identically to the approximate spectral norm $\|f\|_1^\infty$) and γ_2^∞.

★ **Theorem 3.5.5.** *Let $f:\{0,1\}^n\to\{-1,1\}$. Then $\mathsf{margin}(f) = \gamma_2^\infty(f\circ\mathsf{XOR})$.*

★ **Theorem 3.5.6.** *Let $f:\{0,1\}^n\to\mathbb{R}$ and $\epsilon>\delta$. Then $\mathsf{spar}_\epsilon(f) \ge \mathsf{rank}_\epsilon(f\circ\mathsf{XOR})$ and*

$$\mathsf{rank}_\delta(f\circ\mathsf{XOR}) \ge \Omega\left(\frac{(\epsilon-\delta)^2\mathsf{spar}_\epsilon(f)}{n\cdot(\max_x|f(x)|+\delta)^2}\right).$$

Proof. It is easy to see from Theorem 3.5.3 that $\mathsf{spar}_\epsilon(f) \ge \mathsf{rank}_\epsilon(f\circ\mathsf{XOR})$. However, combining Grolmusz' Theorem (Theorem 3.4.2) with Theorem 3.5.4, we have

$$\mathsf{spar}_\epsilon(f) \le O\left(\frac{\left\|\widehat{f}\right\|_{1,\delta}^2 n}{(\epsilon-\delta)^2}\right) \le O\left(\frac{\gamma_{2,\delta}(f\circ\mathsf{XOR})^2 n}{(\epsilon-\delta)^2}\right).$$

Now $\gamma_2(M) \le \sqrt{\mathsf{rank}(M)}\max_{i,j}|M[i,j]|$ (Theorem 3.2.10). Setting M to be the matrix witnessing $\mathsf{rank}_\delta(f\circ\mathsf{XOR})$, we conclude that

$$\begin{aligned}\mathsf{rank}_\delta(f\circ\mathsf{XOR}) = \mathsf{rank}(M) &\ge \frac{\gamma_2(M)^2}{(\max_x|f(x)|+\delta)^2} \ge \frac{\gamma_{2,\delta}(f\circ\mathsf{XOR})^2}{(\max_x|f(x)|+\delta)^2}\\ &\ge \Omega\left(\frac{(\epsilon-\delta)^2\mathsf{spar}_\epsilon(f)}{n\cdot(\max_x|f(x)|+\delta)^2}\right).\end{aligned}$$

□

We do not see any necessity for the $\mathsf{rank}_{1/3}$ lower bound to be a multiplicative factor of n smaller than $\mathsf{spar}_{1/3}$. It would be interesting to see whether this lifting is also as tight as approximate spectral norm's lifting.

Conjecture 3.5.7: Sparsity Lifting

Let $f : \{0,1\}^n \to \{0,1\}$. Then $\mathsf{rank}_{1/3}(f \circ \mathsf{XOR}) = \Theta(\mathsf{spar}_{1/3}(f))$.

We can now combine the fact that $\mathsf{margin}(f) = \gamma_2^\infty(f \circ \mathsf{XOR})$ with the facts that $\log \mathsf{margin}(f) = \mathsf{R}^{\oplus}_{\mathsf{weak},<\frac{1}{2}}(f) + \Theta(1)$ (Theorem 3.3.12) and $\log \gamma_2^\infty(F) = \mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(F) + \Theta(1)$ (Theorem 3.2.27) to get the following lifting theorem.

Theorem 3.5.8. *Let $f : \{0,1\}^n \to \{0,1\}$. Then $\mathsf{R}^{\oplus}_{\mathsf{weak},<\frac{1}{2}}(f) = \mathsf{R}^{\mathsf{cc}}_{\mathsf{weak},<\frac{1}{2}}(f \circ \mathsf{XOR}) + \Theta(1)$.*

★ 3.6 Quantum Communication is Upper Bounded by γ_2^∞

We provide new upper bounds on quantum communication complexity, along similar lines to the square root of approximate rank upper bound given by Gál and Syed [GS19]. Let $\mathsf{Q}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F)$ denote the quantum communication complexity of F when the parties do not share entangled qubits before the protocol starts. Let $\mathsf{Q}^{\mathsf{cc}}_{\mathsf{ent},1/3}$ denote the quantum communication complexity when they do share entanglement. Gál and Syed showed that $\mathsf{Q}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\alpha^2\sqrt{\mathsf{rank}^\alpha(F)}\log(\mathsf{rank}^\alpha(F))$ for all $\alpha > 1$. [9] Their algorithm was inspired by looking at γ_2^α, and by the fact that it is at most $\alpha\sqrt{\mathsf{rank}^\alpha(F)})$.

We can use the Johnson-Lindenstrauss Theorem along with a phased amplitude amplification algorithm to extend this to $\tilde{O}(\gamma_2^\infty(F))$ for quantum communication with entanglement, with an extra factor of $\log n$ for quantum communication without entanglement. There is an upper bound of $O(\gamma_2^\infty(F)^2)$ on the randomized communication complexity by just repeatedly running a small advantage protocol and amplifying the advantage by taking the majority. One would imagine that the quantum protocol can use amplitude amplification to achieve the same goal without the quadratic loss. That is almost true, but there is a problem in using amplitude amplification. This is remedied by using a method similar to quantum fingerprinting as done by Gál and Syed [GL14]. We first go over what amplitude amplification and the variant of quantum fingerprinting are, and then move on to our proofs.

Theorem 3.6.1 (Amplitude amplification [BHMT02])**.** *Let $p \in (0, \frac{1}{2}]$. Let $\mathcal{A}$ be a quantum algorithm operating on a 1-qubit register ans and another register R, that transforms the state*

[9] Recall that similar to the definition of γ_2^α, $\mathsf{rank}^\alpha(F)$ is the minimum rank among matrices M that sign-agree with the sign matrix of F and have values in $[-\alpha,-1] \cup [1,\alpha]$. It is essentially $\mathsf{rank}_\epsilon(F)$, where ϵ is around $\frac{1}{2} - \frac{1}{2\alpha}$.

$|0\rangle_{ans}|0\rangle_R$ to the state $|\psi\rangle = \cos(\theta)|0\rangle_{ans}|v_0\rangle_R + \sin(\theta)|1\rangle_{ans}|v_1\rangle_R$, with v_0 and v_1 being unit vectors and with the promise that $\cos^2(\theta) \in [0, \frac{1}{2} - p] \cup [\frac{1}{2} + p, 1]$. Then there is a quantum algorithm using $O(1/p)$ applications of $\mathcal{A}$ and $\mathcal{A}^{-1}$ that decides with probability at least $2/3$ whether $\cos^2(\theta) \in [0, \frac{1}{2} - p]$ or $\cos^2(\theta) \in [\frac{1}{2} + p, 1]$.

Proof. Before we begin, let us analyze the condition on θ in the theorem statement. We can assume that $\cos(\theta)$ and $\sin(\theta)$ are both positive (negative signs can be pushed into v_0 and v_1). This restricts θ to be in $[0, \pi/2]$. Since the maximum slope of $\cos^2(\theta)$ is 1, $\cos^2(\theta) \in [0, \frac{1}{2} - p]$ implies that $\theta \in [\pi/4 + p, \pi/2]$ and $\cos^2(\theta) \in [\frac{1}{2} + p, 1]$ implies that $\theta \in [0, \pi/4 - p]$.

Let $\mathcal{S}_{ans}$ be the unitary operation that maps a basis vector of the form $|0\rangle_{ans}|i\rangle_R$ to itself and maps a basis vector of the form $|1\rangle_{ans}|i\rangle_R$ to $-|1\rangle_{ans}|i\rangle_R$. Let $\mathcal{S}_0$ be the unitary operation that maps the basis vector $|0\rangle_{ans}|0\rangle_R$ to $-|0\rangle_{ans}|0\rangle_R$ and maps any other basis vector to itself.

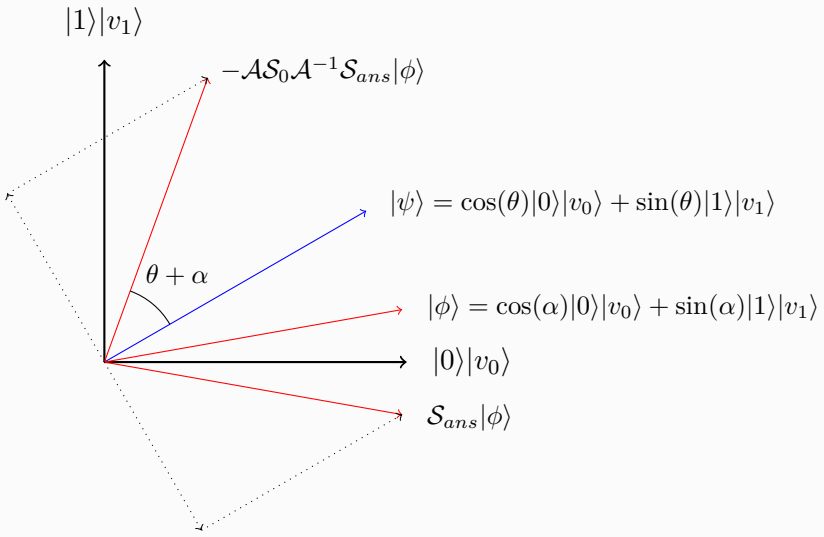


Figure 3.1: One step of amplitude amplification. The dotted lines illustrate how negating the component orthogonal to $|\psi\rangle$ is equivalent to reflecting about $|\psi\rangle$.

For any vector of the form $\sin(\alpha)|1\rangle|v_1\rangle + \cos(\alpha)|0\rangle|v_0\rangle$, we look at what the operation $-\mathcal{A}\mathcal{S}_0\mathcal{A}^{-1}\mathcal{S}_{ans}$ does (see Figure 3.1 for a visual guide). $\mathcal{S}_{ans}$ flips the signs of the components with $B = |1\rangle$, resulting in the vector $-\sin(\alpha)|1\rangle|v_1\rangle + \cos(\alpha)|0\rangle|v_0\rangle = \sin(-\alpha)|1\rangle|v_1\rangle + \cos(-\alpha)|0\rangle|v_0\rangle$. The operation $-\mathcal{A}\mathcal{S}_0\mathcal{A}^{-1}$ is best understood by looking at its action on $|\psi\rangle$ and its action on states orthogonal to $|\psi\rangle$. In the former case, $|\psi\rangle$ is mapped to $|0\rangle|0\rangle$ by $\mathcal{A}^{-1}$, its sign is flipped by $\mathcal{S}_0$ and then it is mapped back to $|\psi\rangle$ by $-\mathcal{A}$. For a state orthogonal to $|\psi\rangle$, $\mathcal{A}^{-1}$ maps it to a state orthogonal to $|0\rangle_{ans}|0\rangle_R$. $\mathcal{S}_0$ does nothing to it, and $-\mathcal{A}$ maps it back to what it originally was but with its sign flipped. Hence this amounts to a reflection around $|\psi\rangle$, and our

state $\sin(-\alpha)|1\rangle|v_1\rangle + \cos(-\alpha)|0\rangle|v_0\rangle$ becomes $\sin(\theta + (\theta - (-\alpha)))|1\rangle|v_1\rangle + \cos(\theta + (\theta - (-\alpha)))|0\rangle|v_0\rangle = \sin(2\theta + \alpha)|1\rangle|v_1\rangle + \cos(2\theta + \alpha)|0\rangle|v_0\rangle$.

This amplification of the magnitude of the component of ψ along the state $|0\rangle_{ans}$ is what is referred to as amplitude amplification. Its benefits are immediate. If $\cos^2(\theta)$ was equal to $\frac{1}{2} + \epsilon$ for a small ϵ (i.e. $\theta \approx \pi/4 + \epsilon$), then with $\pi/4\epsilon$ applications of $-\mathcal{A}\mathcal{S}_0\mathcal{A}^{-1}\mathcal{S}_{ans}$, the magnitude of the component along $|0\rangle_{ans}$ becomes close to 1. Classically we would have to run an ϵ advantage protocol $O(1/\epsilon^2)$ times to be sure whether the deviation from half was positive or negative.

However, this operation is not monotone. If you apply $-\mathcal{A}\mathcal{S}_0\mathcal{A}^{-1}\mathcal{S}_{ans}$ $\pi/4\epsilon$ times, and $\cos^2(\theta)$ happened to be much bigger than $\frac{1}{2} + \epsilon$, then the magnitude of the component along $|0\rangle_{ans}$ could be anywhere between 0 and 1. So the algorithm goes in phases, each phase checking for a different range of ϵ, as illustrated in Figure 3.2.

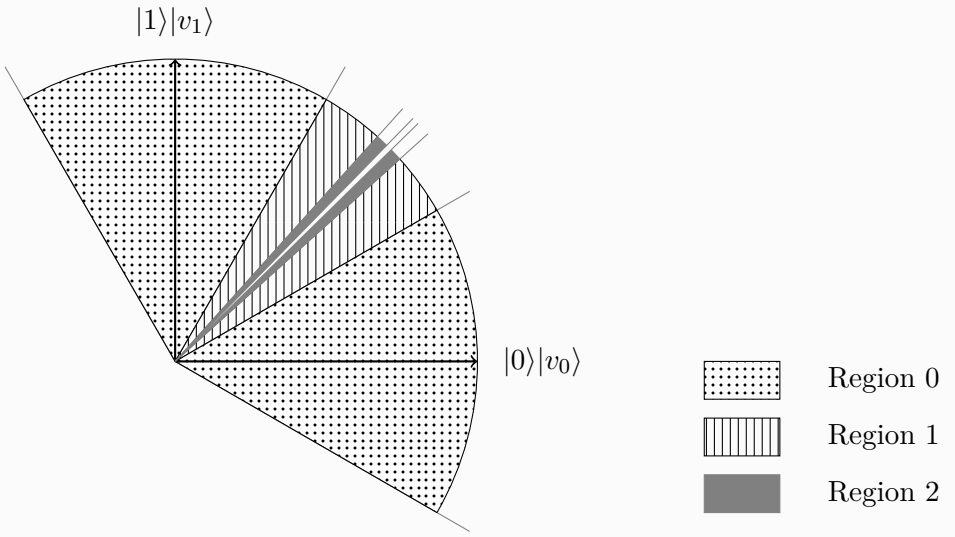


Figure 3.2: All states in region 0 are far enough from $\pi/4$ for a few measurements to be sufficient for differentiating between "larger than $\pi/4$" and "smaller than $\pi/4$". In phase 1, any $|\psi\rangle$ in region 1 gets mapped to a state in region 0. It is then measured, and this process is repeated a few times to bring down the probability of error. Similarly in phase i, any state that was originally in region i is mapped to a state in region 0. Within $O(1/p)$ regions, all states that satisfy Theorem 3.6.1's promise are covered.

We can now describe the quantum algorithm. Let i_{max} be $\lceil \log_5(\pi/12p) \rceil$. Let $\mathsf{amp}_i = \frac{5^i - 1}{2}$ and $\mathsf{rep}_i = 100(i_{max} + 1 - i)$. Execute the following for $i = 0, 1, \ldots, i_{max}$.

- Repeat the following experiment rep_i times.
  - Starting with $|0\rangle_{ans}|0\rangle_R$, apply the unitary $\mathcal{A}$.
  - Apply the unitary $-\mathcal{A}\mathcal{S}_0\mathcal{A}^{-1}\mathcal{S}_G$ amp_i times.

– Measure register ans.

- If the number of times the measurement returned 1 is greater than $.6\mathsf{rep}_i$, then output $\theta \geq \pi/4 + p$.
- If the number of times the measurement returned 1 is less than $.4\mathsf{rep}_i$ then output $\theta \leq \pi/4 - p$.
- Else move on to the next value of i.

Note that when $i = 0$, we are just constructing $|\psi\rangle$ and measuring ans. If $\theta > \pi/3$ the probability of measuring a 1 is $\geq 3/4$. Using Hoeffding's Inequality (Lemma 3.1.1), we see that the probability of measuring 1 less than $.6\mathsf{rep}_0$ times is $< e^{-2(3/4-.6)^2\mathsf{rep}_0} \leq (1/4)^{i_{max}+1}$. Similarly, for $\theta < \pi/6$ the probability of not outputting $\theta < \pi/4 - p$ is $\leq (1/4)^{i_{max}+1}$.
When $\theta \notin [0, \pi/6] \cup [\pi/3, \pi/2]$, we want to ensure that a θ which is greater than $\pi/4$ does not output $\theta < \pi/4 - p$. Here the probability of measuring 1 is at least $\frac{1}{2}$ and so the probability of getting a 1 less than $.4\mathsf{rep}_0$ times is $< e^{-2(.5-.4)^2\mathsf{rep}_0} \leq (1/4)^{i_{max}+1}$. Similarly a θ which is less than $\pi/4$ will output $\theta > \pi/4 - p$ with probability $\leq (1/4)^{i_{max}+1}$.
In phase i, θ gets transformed to $\theta + \frac{5^i-1}{2} \cdot 2\theta = 5^i\theta$ before measurement. In particular the interval $[\pi/4 + \frac{\pi}{5^i \cdot 12}, \pi/4 + \frac{\pi}{5^{i-1} \cdot 12}]$ gets transformed to the interval $[5^i\pi/4 + \frac{\pi}{12}, 5^i\pi/4 + \frac{5\pi}{12}]$, which is congruent to $[\pi/4 + \pi/12, \pi/4 + 5\pi/12] = [\pi/3, 2\pi/3]$ modulo π. Similarly the interval $[\pi/4 - \frac{\pi}{5^{i-1} \cdot 12}, \pi/4 - \frac{\pi}{5^i \cdot 12}]$ gets transformed to the interval $[-\pi/6, \pi/6]$. Measurements taken now will have the same guarantees as measurements made in phase 0.
However, rep_i changes with i, so the probabilities of failure in phase i become at most $(1/4)^{i_{max}-i+1}$ instead of the $(1/4)^{i_{max}+1}$ that we got in phase 0.
Note that every θ in the range $[0, \pi/4-p] \cup [\pi/4+p, \pi/2]$ is mapped to a $\theta \in [-\pi/6, \pi/6] \cup [\pi/3, 2\pi/3]$ in some phase. Let us refer to this phase as i_θ.
The probability that a θ outputs wrongly before phase i_θ is at most $\sum_{i=0}^{i_\theta - 1} (1/4)^{i_{max}-i+1}$. The probability that it does not output the correct answer in phase i_θ is at most $(1/4)^{i_{max}-i_\theta+1}$. A union bound over all the errors gives a total error probability of at most $1/4 + 1/4^2 + \cdots < 1/3$.

The number of times $\mathcal{A}$ or its inverse are used is at most

$$\begin{aligned}\sum_{i=0}^{i_{max}}(1+2\mathsf{amp}_i)\mathsf{rep}_i &= \sum_{i=0}^{i_{max}} 5^i \cdot 100(i_{max}+1-i) \\ &\leq 100\sum_{j=0}^{i_{max}} 5^{i_{max}-j}\cdot(j+1) \\ &= O(5^{i_{max}}) = O(1/p).\end{aligned}$$

The last line above follows from the line before it by noting that each successive term in the summation is at most $2/5$ times the previous term. □

Gal et al. [GS19] uses the following theorem that uses a technique similar to quantum fingerprinting [BCWdW01] that we'll refer to as quantum vector-embedding.

Theorem 3.6.2. *There is a quantum protocol without entanglement of cost* $\log \mathsf{rank}^{\alpha}(F)$ *that has advantage* $\Omega(1/\alpha\sqrt{\mathsf{rank}^{\alpha}(F)})$.

Proof. Given a unit vector $v \in \mathbb{R}^d$, the quantum computational model allows us to represent it in a $\lceil \log d \rceil$-qubit register R as $|\psi\rangle_R = \sum_{i\in[d]} v_i|i\rangle$. This representation doesn't allow us to recover v but it is useful for estimating the dot product of vectors. Suppose that Alice has v_1 and Bob has v_2. Here is how they would estimate $\langle v_1, v_2\rangle$. Alice initializes a 1-qubit register B in the state $|0\rangle_B$, and a $\lceil \log d \rceil$ qubit register V in the state $|\overline{0}\rangle_V$.

- Alice applies the Hadamard transform on B.
- Controlled on $B = |0\rangle$, Alice transforms $|\overline{0}\rangle_V$ to $|\phi_x\rangle_V$ where $|\phi_x\rangle = \sum_{i\in[d]} v_1[i]|i\rangle$. This means that

  $$|0\rangle_B|\overline{0}\rangle_V \mapsto |0\rangle_B|\phi_x\rangle_V,\ \forall i\ |1\rangle_B|i\rangle_V \mapsto |1\rangle_B|i\rangle_V.$$

  Which unitary she uses to do the transformation on V is irrelevant.
- Alice sends across B and V to Bob.
- Controlled on $B = |1\rangle$, Bob transforms $|\overline{0}\rangle_V$ to $|\phi_y\rangle_V$ where $|\phi_y\rangle = \sum_{i\in[d]} v_2[i]|i\rangle$. This means that

  $$|1\rangle_B|\overline{0}\rangle_V \mapsto |1\rangle_B|\phi_y\rangle_V,\ \forall i\ |0\rangle_B|i\rangle_V \mapsto |0\rangle_B|i\rangle_V.$$

  Which unitary he uses to do the transformation on V is irrelevant.
- Bob sends back B and V to Alice.

- Alice applies the Hadamard transform on B.

On the state $|0\rangle_B|\overline{0}\rangle_V$, the above protocol produces the state $\frac{1}{2}(|0\rangle_B|\phi_x + \phi_y\rangle_V + |1\rangle_B|\phi_x - \phi_y\rangle_V)$. The probability of measuring a 1 in register B in this state is $\frac{1}{4}\langle\phi_x - \phi_y \mid \phi_x - \phi_y\rangle = \frac{1}{4}(2 - 2\langle\phi_x \mid \phi_y\rangle) = \frac{1}{2} - \frac{1}{2}\langle v_1, v_2\rangle$.
Let $r = \mathsf{rank}^\alpha(F)$. Let M be the matrix guaranteed by the definition of rank^α that sign-agrees with the sign matrix of F, has entries in $[-\alpha, -1] \cup [1, \alpha]$ and has rank r. We know (see Observation 3.2.12) that M/α

- is pointwise-close to the sign matrix of F, taking values in $[-1, -1/\alpha] \cup [1/\alpha, 1]$.
- has a decomposition XY, where each row of X and column of Y is of dimension r and has length at most $\sqrt[4]{r}$.

We can add two extra dimensions to X and Y so that XY is still M/α but the length of every row of X and column of Y is *exactly* $\sqrt[4]{r}$. Alice and Bob scale X and Y by $1/\sqrt[4]{r}$ and use the above protocol on $X[x, *]$ and $Y[*, y]$ to estimate $M[x, y]$. This gives them a quantum circuit that involves $O(\log r)$ communication and on input (x, y) yields a state $\cos(\theta)|0\rangle|v_0\rangle + \sin(\theta)|1\rangle|v_1\rangle$, where $\cos^2(\theta)$ belongs to $[\frac{1}{2} + \frac{1}{2\alpha\sqrt{r}}, \frac{1}{2} + \frac{1}{2\sqrt{r}}]$ or $[\frac{1}{2} - \frac{1}{2\sqrt{r}}, \frac{1}{2} - \frac{1}{2\alpha\sqrt{r}}]$ depending on the value of $F(x, y)$. □

Given a protocol with such advantage, Alice and Bob would use amplitude amplification (see Theorem 3.6.1) to figure out whether $F(x, y)$ was 0 or 1. This involves them implementing the operations $\mathcal{S}_0, \mathcal{S}_{ans}, \mathcal{A}$ and $\mathcal{A}^{-1}$. Note that all the qubits are held by Alice at the end of the above protocol. Hence she can easily implement $\mathcal{S}_0$ and $\mathcal{S}_{ans}$. The operation $\mathcal{A}$ is the above protocol itself. The operation $\mathcal{A}^{-1}$ is the above protocol again, but in reverse where every unitary is replaced with its inverse. This is due to the simple observation that $(U_1U_2)^{-1} = U_2^{-1}U_1^{-1}$.

The final protocol would involve $O(\alpha\sqrt{\mathsf{rank}^\alpha(F)})$ applications of the quantum vector-embedding protocol, and so the total cost of the protocol is at most $O(\alpha\sqrt{\mathsf{rank}^\alpha(F)}\log \mathsf{rank}^\alpha(F))$.

We exhibit the following upper bounds.

Theorem 3.6.3. *For any function $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$,*

1. $\mathsf{Q}^{\mathsf{cc}}_{\mathsf{ent},1/3}(F) \leq O(\gamma_2^\infty(F)\log\gamma_2^\infty(F))$*, witnessed by a protocol using entanglement solely as public randomness, and*
2. $\mathsf{Q}^{\mathsf{cc}}_{\mathsf{pri},1/3}(F) \leq O(\gamma_2^\infty(F)\log\gamma_2^\infty(F) + \log n)$.

Proof. Let $r = \gamma_2^\infty(F)$. By the definition of γ_2^∞, we have a matrix M sign-agreeing with the sign matrix of F with values in the range $(-\infty, -1] \cup [1, \infty)$ such that $M = XY$,

where each row of X and column of Y have norm at most $\sqrt{r}$. As done in the proof of Theorem 3.6.2, we can make the norms *exactly* $\sqrt{r}$. This also means that any entry of M has absolute value at most r. Hence $\gamma_2^\infty(F) = \gamma_2^r(F)$.

Alice and Bob can try touse the decomposition XY to compute the function, but the dimensions of the rows of X and columns of Y can be as large as the approximate rank. In order to get shorter vectors, they can use Theorem 3.4.4 (the Johnson-Lindenstrauss-like theorem). This would result in a decomposition $X'Y'$ where each row of X' and column of Y' are of dimension $O(r^2 n)$, each row and column having norm at most $\sqrt{r}$ (see Equation (3.1)), and the inner product of any row and any column lying in $[-1, -1/2r] \cup [1/2r, 1]$ (by setting $\epsilon = 1 - 1/r$ and $\delta = 1 - 1/2r$). Note that the multiplicative factor of n only appears because we are using a union bound over $\approx 2^{2n}$ entries.

Since they only care about preserving the norms of ϕ_x, ψ_y and $\phi_x + \psi_y$ instead of *all* the $\approx 2^{2n}$ vectors, they can bring the dimension down to $O(r^2)$. (Recall from the proof of Theorem 3.4.4 that the vectors ϕ_x and ψ_y are obtained by scaling X and Y respectively, and have norms at most 1.) Their protocol would begin as follows.

- Use public randomness to sample the appropriate dimensional random vectors $\mathbf{v^{(1)}}, \ldots, \mathbf{v^{(t)}}$ for $t = O(\frac{r^2(1-\epsilon)^2}{(\delta-\epsilon)^2}) = O(r^2)$.
- Alice creates the vector $v_x = (\langle \mathbf{v^{(1)}}, \phi_x \rangle, \ldots, \langle \mathbf{v^{(t)}}, \phi_x \rangle)$.
- Bob creates the vector $v_y = (\langle \mathbf{v^{(1)}}, \psi_y \rangle, \ldots, \langle \mathbf{v^{(t)}}, \psi_y \rangle)$.

We can analyze these vectors as done in the analysis of Theorem 3.4.4. We can set the constant hidden in $t = O(r^2)$ so that with probability at least 0.99, $\langle v_x, v_y \rangle \in \langle \phi_x, \psi_y \rangle \pm (\delta - \epsilon)$, $\|v_x\|^2 \in \|\phi_x\|^2 \pm (\delta - \epsilon)$ and $\|v_y\|^2 \in \|\psi_y\|^2 \pm (\delta - \epsilon)$. Hence it is extremely likely that $\langle v_x, v_y \rangle \in [-1, -1/2r]$ if $F(x, y)$ were -1 and in the range $[1/2r, 1]$ if $F(x, y)$ were 1. Alice and Bob add extra coordinates so that the norm of v_x and v_y are 2 without chnaging $\langle v_x, v_y \rangle$.

We can now proceed with vector-embedding as done in the proof of Theorem 3.6.2, yielding a protocol of cost $O(\log r)$ (because of the dimension of the vectors v_x, v_y being poly(r)), and with advantage $\Omega(1/r)$ (because the vectors have constant norm and the magnitude of the inner product of the vectors is $\Omega(1/r)$). We can now use amplitude amplification to conclude part 1 of the theorem.

For part 2 of the theorem, we simply use Newman's Theorem (Theorem 3.4.3) on part 1. □

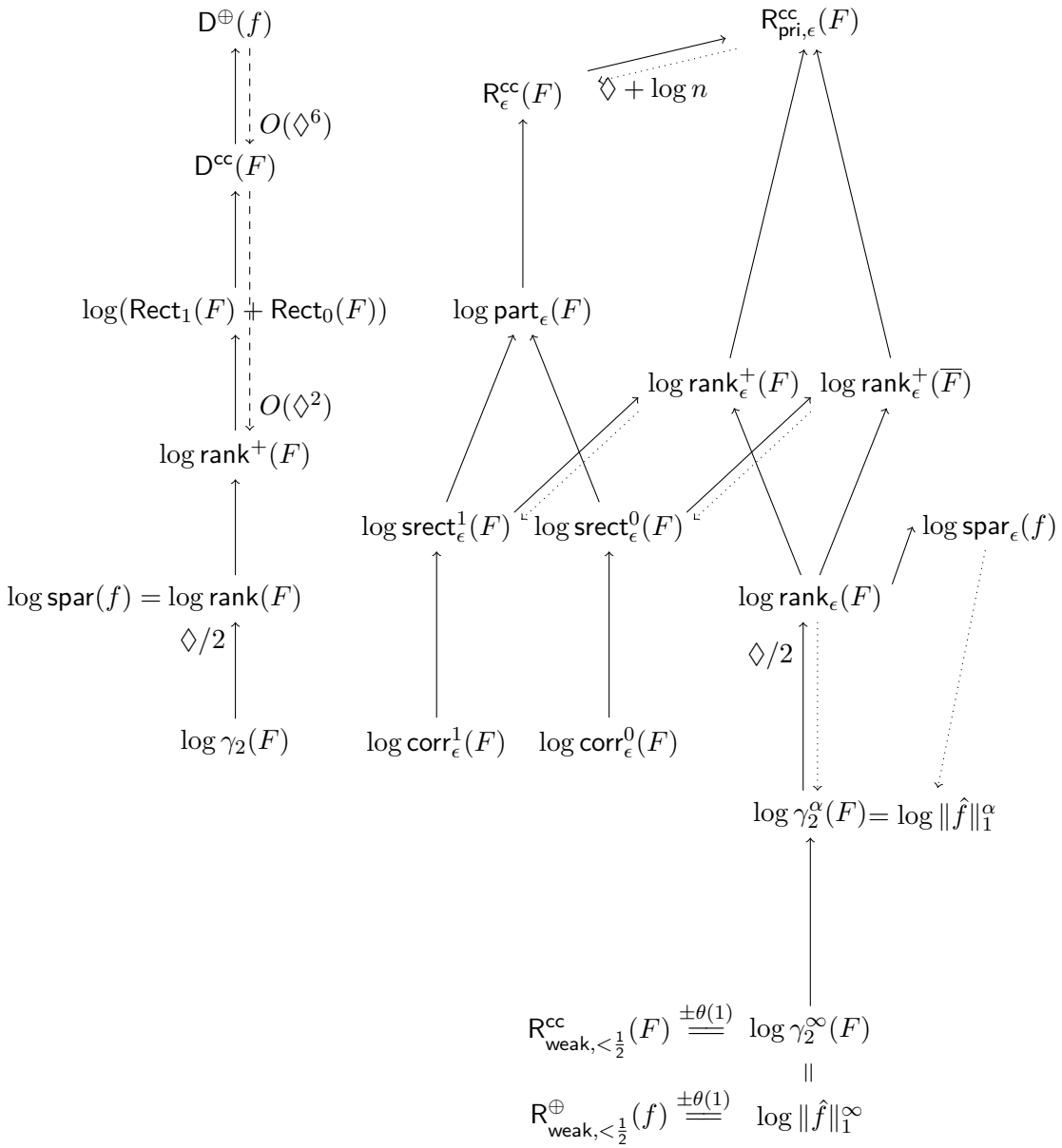


Figure 3.3: Relations between various measures of Boolean functions f and F. When a measure of f is compared with a measure of F, it is implicit that $F := f \circ \mathsf{XOR}$ (see Section 3.5).
$A \longrightarrow B$ indicates $A \leq B$.
$A \xrightarrow{g(\lozenge)} B$ indicates that not only is $A \leq B$, but $A \leq g(B)$.
$A \overset{g(\lozenge)}{\dashrightarrow} B$ indicates that although it may not be the case that $A \leq B$, it is still true that $A \leq g(B)$.
A dotted arrow is a special case of the above arrow and it indicates $A \leq 2B + \log n + O(1)$ (unless stated otherwise). It also hides differences in the error parameters of A and B. All the dotted relations follow the framework described in Section 3.4.

Chapter 4

Refuting the Log-Approximate-Rank Conjecture

This work was done in collaboration with Arkadev Chattopadhyay and Nikhil S. Mande. It has appeared at STOC '19 and was published in JACM. [CMS20]

In this chapter, we show that the logarithms of the approximate rank, approximate nonnegative rank, spectral norm (or γ_2 norm) are not well suited to characterize randomized parity decision tree complexity or randomized communication complexity. We also show that the parity kill number of a function with small spectral norm can be large. All these separations are witnessed by the function SINK or its lift $\mathsf{SINK} \circ \mathsf{XOR}$. Figure 4.1 shows various measures for $\mathsf{SINK} \circ \mathsf{XOR}$. Figure 4.2 shows most of the conjectures refuted by SINK or $\mathsf{SINK} \circ \mathsf{XOR}$. The intuition behind all this is the observation that rank and norms are subadditive, and their logarithms hence grow very slowly with respect to addition. On the other hand, we have no reason to believe that randomized communication complexity grows as slowly. This motivates us to look at the following class of functions designed to have small spectral norm as a result of being the summation of functions of small spectral norm.

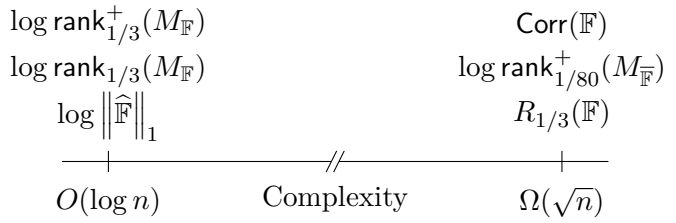


Figure 4.1: Various complexity measures of our function $\mathbb{F} := \mathsf{SINK} \circ \mathsf{XOR}$ on $2n$ bits. Prior to this work, no exponential separation between any pair of these measures was known for a total function.

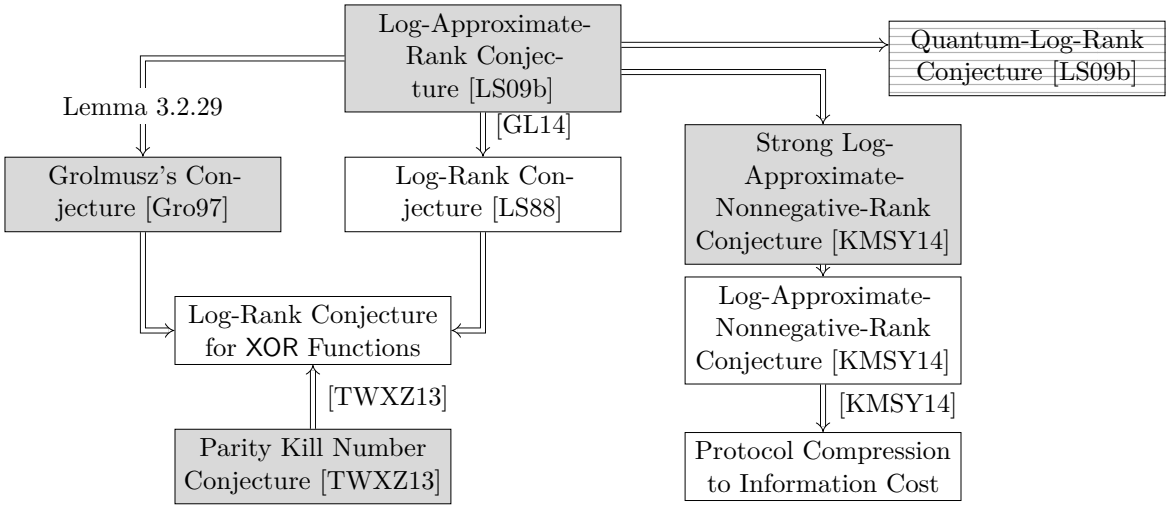


Figure 4.2: Implications between various interesting conjectures. We use the SINK function to disprove the shaded conjectures. The Quantum-Log-Rank Conjecture was subsequently falsified [ABT19, SdW19] using SINK, and the rest remain unresolved.

4.1 The Disjoint Subcube Functions

A subcube is a subset of the Boolean hypercube with the following structure.

Definition 4.1.1 (Subcube)**.** *A set $T \subseteq \{0,1\}^n$ is said to be a subcube if there exist coordinates $i_1, \ldots, i_k \in [n]$ and $a_1, \ldots, a_k \in \{0,1\}$ such that $T = \{x \in \{0,1\}^n | x_{i_1} = a_1, x_{i_2} = a_2, \ldots, x_{i_k} = a_k\}$. We call $\mathsf{fixed}(T) := \{i_1, \ldots, i_k\}$ the set of fixed coordinates in T.*

We can see that the spectral norm (Definition 3.3.5) of any subcube is at most 1.

Claim 4.1.2. *Let S be any subcube in $\{0,1\}^k$. Then $\left\|\widehat{(\mathsf{DS}_S)}\right\|_1 = 1$.*

Proof. It is easy to see that the following polynomial evaluates to 1 on all $x \in S$, and 0 otherwise. (Recall from Definition 3.1.3 that $\chi_{\{j\}}(x) = (-1)^{x_j}$.)

$$p_S(x) = \prod_{j \in \mathsf{fixed}(S)} \left(\frac{1 + (-1)^{b_j} \chi_{\{j\}}(x)}{2} \right),$$

where S fixes x_j to b_j. Expanding the above gives a sum of $2^{|\mathsf{fixed}(S)|}$ monomials, each with a coefficient of absolute value $2^{-|\mathsf{fixed}(S)|}$. Thus the spectral norm of p_S is 1. □

Now we define our class of Disjoint Subcube functions.

Definition 4.1.3 ($\mathsf{DS}_\mathcal{S}$)**.** *Consider a set $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ of m disjoint subcubes in $\{0,1\}^k$. Define*

$$\mathsf{DS}_\mathcal{S}(x) := \begin{cases} 1 & \text{if } x \in \bigcup_{i \in [m]} S_i \\ 0 & \text{otherwise.} \end{cases}$$

Claim 4.1.4. *Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be any set of m disjoint subcubes in $\{0,1\}^k$. Then,*

$$\left\| \widehat{(\mathsf{DS}_{\mathcal{S}})} \right\|_1 \leq m.$$

Proof. Since the subcubes are disjoint, the exact polynomial representation for $\mathsf{DS}_{\mathcal{S}}$ is

$$p_{\mathsf{DS}_{\mathcal{S}}} = \sum_{S_i \in \mathcal{S}} p_{S_i},$$

where p_{S_i} is the polynomial that evaluates to 1 for all $x \in S_i$ and 0 otherwise. Since the spectral norm of each p_{S_i} is 1 the spectral norm of $\mathsf{DS}_{\mathcal{S}}$ is at most m. □

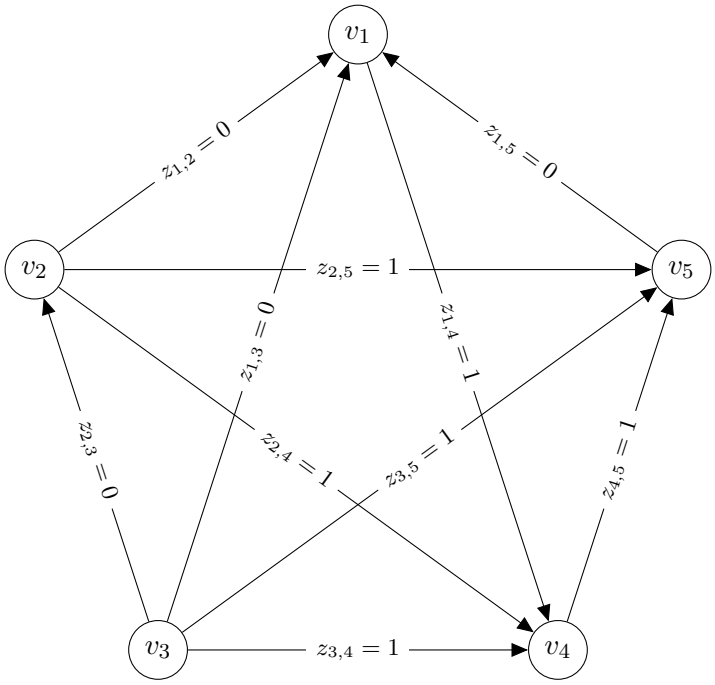
4.1.1 The Sink Function

We now define the function SINK that will be crucial for this chapter. The definition involves the concept of a tournament. A tournament on n vertices is a directed graph which is obtained by assigning directions to each edge in the undirected complete graph K_n.

Definition 4.1.5 (SINK)**.** *Consider a tournament on m vertices defined by the $\binom{m}{2}$ variables $x_{i,j}$ for $i < j \in [m]$ in the following way: $x_{i,j} = 1 \implies v_i \to v_j$ is the direction of the (v_i, v_j) edge, and $x_{i,j} = 0 \implies v_i \leftarrow v_j$ is the direction. The function* SINK *computes whether or not there is a sink in the graph. In other words,*

$$\mathsf{SINK}(x) = 1 \iff \exists\ i \in [m] \text{ such that all edges adjacent to } v_i \text{ are incoming.}$$

The following picture shows a 0-input of SINK.



We now note that SINK is a specific instance of DS.

Lemma 4.1.6. *Consider the function* SINK *on* $\binom{m}{2}$ *variables. There exists a set of disjoint subcubes* $\mathcal{S} = \{S_1, \ldots, S_m\}$ *in* $\{0,1\}^{\binom{m}{2}}$ *such that* $\mathsf{SINK} = \mathsf{DS}_{\mathcal{S}}$.

Proof. Label each of the $\binom{m}{2}$ coordinates by a unique (i,j) pair for $i < j \in [m]$. The description of $\mathsf{fixed}(S_i)$ is given below.

- $x_{i,j} = 0$ for all $j > i$.
- $x_{j,i} = 1$ for all $j < i$.

For each $i < j$, $S_i \cap S_j = \emptyset$ since $x_{i,j} = 0$ in S_i and $x_{i,j} = 1$ in S_j. Thus, the subcubes $S_1, \ldots, S_m$ are disjoint. Note that the function $\mathsf{DS}_{\mathcal{S}}$ is exactly the same as SINK, with $x \in S_i$ if and only if v_i is a sink when the edges are oriented according to x. □

Unless mentioned otherwise, we consider the SINK function to be on $\binom{m}{2}$ variables.

Projections

While analyzing the complexity of SINK, we will often use projections of the inputs. Let $X \in \{0,1\}^{\binom{m}{2}}$. To see how X orients the edges incident to a vertex v_i, let E_{v_i} be the set of $m-1$ input coordinates that correspond to the edges incident to v_i. We use the notation $X_{v_i} \in \{0,1\}^{m-1}$ to denote the input projected to the coordinates in E_{v_i}. Note that X_{v_i} decides whether or not v_i is a sink. By z_i, we refer to the $(m-1)$-bit string such that v_i is a sink if and only if $X_{v_i} = z_i$.

4.2 Simplicity of SINK

Building on the fact that SINK is an instance of a Disjoint Subcube function, we note the following ways in which SINK is a 'simple' function.

Theorem 4.2.1 (Simplicity of SINK)**.**

1. $\gamma_2(\mathsf{SINK} \circ \mathsf{XOR}) = \left\|\widehat{\mathsf{SINK}}\right\|_1 \leq m$.
2. $\mathsf{rank}_{1/3}(\mathsf{SINK} \circ \mathsf{XOR}) \leq \mathsf{spar}_{1/3}(\mathsf{SINK}) = O(m^4)$.
3. $\mathsf{rank}^{+}_{1/3}(\mathsf{SINK} \circ \mathsf{XOR}) = O(m^5)$.

Proof. **Proof of Part 1:** Theorem 3.5.2 states that $\gamma_2(f \circ \mathsf{XOR}) = \left\|\widehat{f}\right\|_1$. Since SINK is an instance of $\mathsf{DS}_{\mathcal{S}}$ where $|\mathcal{S}| = m$ (Lemma 4.1.6) and $\left\|\widehat{\mathsf{DS}_{\mathcal{S}}}\right\|_1 \leq |\mathcal{S}|$ (Claim 4.1.4), we know that $\left\|\widehat{\mathsf{SINK}}\right\|_1 \leq m$. Note that composing with XOR does not change the spectral norm: If $p_{\mathsf{SINK}}(x)$ is the polynomial computing SINK, then $p_{\mathsf{SINK}\circ\mathsf{XOR}}(x,y)$ is obtained by replacing every monomial $\chi_S(x)$ in $p_{\mathsf{SINK}}(x)$ with $\chi_S(x)\chi_S(y) = \chi_S(x \oplus y)$.

Proof of Part 2: Recall that Part 1 implies that $\left\|\widehat{\mathsf{SINK}}\right\|_1 \leq m$. Theorem 3.4.2 implies the existence of a function $f : \{0,1\}^{\binom{m}{2}} \to \mathbb{R}$ with sparsity $O(m^4)$ such that $|\mathsf{SINK}(x) - f(x)| \leq 1/3$ for all x. From Theorem 3.5.3, $f \circ \mathsf{XOR}$ has rank $O(m^4)$ and $|\mathsf{SINK} \circ \mathsf{XOR}(x,y) - f \circ \mathsf{XOR}(x,y)| \leq 1/3$ for all x, y.

Proof of Part 3: We show that the communication matrix of $\mathsf{SINK} \circ \mathsf{XOR}$ is pointwise close to a matrix M which can be written as the nonnegative sum of at most $O(m^5)$ nonnegative rank-1 matrices.

Note that $\mathsf{SINK} \circ \mathsf{XOR}(X, Y)$ can be written as an OR of 'Equalities': $\bigvee_{i=1}^{m} (X_{v_i} = Y_{v_i} \oplus z_i)$. Since at most one of these Equalities can fire for any input, it is in fact the sum of the m Equalities. It is well known that any Equality where each party gets $m-1$ bits can be solved to within error $1/6m$ with a public coin randomized communication protocol of cost $\log m + O(1)$.

> Alice and Bob use the public randomness to sample sets $S_1, S_2, S_t \subseteq [m-1]$ uniformly at random. Alice then sends Bob the t-bits string $\{\oplus_{S_j} x\}_{j \in [t]}$ to Bob, who responds with the output 1 if it is the same as $\{\oplus_{S_j} y\}_{j \in [t]}$ and the output 0 otherwise. If the strings are equal, then the output is 1 with probability 1. If they are different, let i be such that $x_i \neq y_i$. For every $S \subseteq [m-1] \setminus \{i\}$, $\oplus_S x = \oplus_S y$ if and only if $\oplus_{S \cup \{i\}} x \neq \oplus_{S \cup \{i\}} y$. Hence the probability that $\oplus_{S_j} x = \oplus_{S_j} y$ is half and the probability that the output is 1 is $1/2^t$. The total communication is $t + 2$ bits.

Newman's Theorem (Theorem 3.4.3) states that this can be converted to a private coin protocol, introducing an additional error of at most δ, with an additional communication cost of $\log m + 2\log(1/\delta) + O(1)$. Setting $\delta = 1/6m$, we get a $4\log m + O(1)$ cost private coin protocol with error $1/3m$. Lower bound 6 states that $R_\epsilon^{pri}(F) \geq \log \mathsf{rank}_\epsilon^+(M_F)$. So the matrix for any of the Equalities has $(1/3m)$-approximate nonnegative rank at most $O(m^4)$. By adding up these m approximating matrices, we get a matrix of nonnegative rank $O(m^5)$ that pointwise $1/3$-approximates $M_{\mathsf{SINK} \circ \mathsf{XOR}}$. □

4.3 SINK is hard: Parity Kill Number and RPDT Complexity

4.3.1 Affine Subspaces: Notation and Facts

We will identify $\{0,1\}^n$ with the vector space $\mathbb{F}_2^n$, where 0 and 1 are identified with the additive identity and multiplicative identity of $\mathbb{F}_2$, respectively. We can now talk of linear forms $\ell \in \{0,1\}^n$ that map an $x \in \{0,1\}^n$ to $\langle \ell, x \rangle := \sum_{i \in [n]} c_i x_i \in \{0,1\}$, where $\ell = (c_1, c_2, \ldots, c_n)$. We can also take the sums of linear forms to get other linear forms in their span. A linear equation $\langle \ell, x \rangle = a$ will sometimes be represented as $(\ell, a) \in \{0,1\}^{n+1}$ in order to avoid confusion. Note that a set of linear equations logically implies every equation in its span.

Definition 4.3.1 (Affine subspace)**.** *A set $T \subseteq \{0,1\}^n$ is said to be an affine subspace if there exist independent linear forms $\ell_1, \ldots, \ell_k$ and $a_1, \ldots, a_k \in \{0,1\}$ such that $T = \{x \in \{0,1\}^n | \langle \ell_i, x \rangle = a_i$ for all $i \in [k]\}$. k is called the co-dimension of T, denoted* co-dim(T)*. It can be seen that $|T| = 2^{n-k}$.*

Claim 4.3.2. *Let W be an affine subspace of $\{0,1\}^n$ defined by a system of equations with span $\mathcal{L}$. Let $S \subseteq [n]$. Let $\mathcal{L}_S \subseteq \mathcal{L}$ be the subset of equations that are supported completely by variables indexed within S. For any $y \in \{0,1\}^S$, the number of extensions of y in W is 0 if y violates a constraint in $\mathcal{L}_S$ and $2^{\dim(W)-(|S|-\dim(\mathcal{L}_S))}$ otherwise.*

Proof. Let B_S be a basis for $\mathcal{L}_S$. Extend B_S to get a basis B for $\mathcal{L}$.

- Let $T_S \subseteq \{0,1\}^S$ be the affine subspace where the equations $\mathcal{L}_S$ are satisfied. If $y \in \{0,1\}^S$ is not in T_S, then y must contradict one of the equations in $\mathcal{L}_S$. Since this is also an equation satisfied in W, no extension of y is in W.

- Consider the set $S_y \subseteq W$ defined as those elements of W which also satisfy the equations $\{\langle e_i, x\rangle = y_i\}_{i\in S}$. S_y is exactly the set of extensions of y within W. S_y already satisfies the equations in $\mathcal{L}_S$. Hence S_y is adding only $|S| - \dim(\mathcal{L}_S)$ more equations to W. If we show that these new equations are independent of B, or equivalently that $\{\langle e_i, x\rangle = y_i\}_{i\in S}$ is independent of $B \setminus B_S$, we can conclude that $|S_y| = 2^{\dim(W)-(|S|-\dim(\mathcal{L}_S))}$.

  Suppose $(B\setminus B_S)\cup\{\langle e_i, x\rangle = y_i\}_{i\in S}$ has a dependency. Then there must be a $B' \subseteq B\setminus B_S$ and an $S' \subseteq S$ such that the equation $\sum_{(\ell,a)\in B'}(\ell, a) = \sum_{i\in S'}(e_i, y_i)$. Hence $B \setminus B_S$ generates an element of $\mathcal{L}_S$ (since it cannot generate anything outside of $\mathcal{L}$), contradicting the fact that B is a basis, and finishing the proof.

□

We will also make use of the following corollary.

Corollary 4.3.3. *Let A and B be affine subspaces of $\{0,1\}^n$. If $\frac{|A\cap B|}{|B|} < \frac{|A|}{2^n}$, then $A \cap B = \emptyset$.*

Proof. The affine subspace A sets $n - \dim(A)$ independent linear forms. We can take a linear transformation of $\{0,1\}^n$ such that A now sets the first $n - \dim(A)$ coordinates. Note that since the linear transformation is a bijection, the quantity $|A \cap B|$ does not change. The quantity $|A \cap B|$ counts the number of extensions of the $n - \dim(A)$-bit string set by A that lie in B. The above claim implies that $|A \cap B|$ is either 0 or $2^{\dim(B)-(n-\dim(A)-c)}$ where $c \geq 0$ corresponds to $\dim(\mathcal{L}_S)$ in the statement of the claim. So $|A \cap B|$ is either 0 or at least $\frac{|B|\cdot|A|}{2^n}$, as the corollary claims. □

4.3.2 Large Parity Kill Number

The fraction of 1-inputs that SINK has grows exponentially small with m: For any vertex v, exactly $1/2^{m-1}$ fraction of inputs have v as a sink. Since these are disjoint events, the fraction of 1s is $m/2^{m-1}$. Despite the abundance of 0-inputs, we will show that it is hard to find a large affine subspace that is monochromatically 0. We will in fact show a stronger statement.

Lemma 4.3.4 (Sink Avoidance is Costly)**.** *Fix any $k \leq m$. Let W be an affine subspace such that the vertices $v_1, v_2, \ldots, v_k$ are not sinks in any input in W. Then co-dim$(W) \geq 2k/3$.*

Proof. Let W be an affine subspace of $\{0,1\}^{\binom{m}{2}}$, defined by a system of linear equations with span $\mathcal{L}$, such that for every input in W, none of $V = \{v_1, \ldots, v_k\}$ is a sink. Recall that z_i is defined as the $m-1$-bit string such that v_i is a sink if and only if x restricted to E_{v_i} is equal to z_i. So we have that for every $1 \leq i \leq k$, no extension of z_i appears in W. Then, by Claim 4.3.2, we know that for every $1 \leq i \leq k$, there exists at least one linear equation in $\mathcal{L}$ of the form $\langle \ell_i, x \rangle = a_i$, where ℓ_i is supported completely by the variables indexed within E_{v_i}, and a_i is such that $\langle \ell_i, z_i \rangle \neq a_i$. Let us call such a linear constraint a *v_i-constraint*.

Since $x_{i,j}$ is the only variable in $E_{v_i} \cap E_{v_j}$, we have $x_{i,j} = 0$ and $x_{i,j} = 1$ as the only candidates for equations that are simultaneously v_i-constraints and v_j-constraints. But, assuming that $i < j$, the equation $x_{i,j} = 0$ represents a $v_j \to v_i$ edge and is hence a v_j-constraint and not a v_i-constraint. Similarly $x_{i,j} = 1$ is not a v_j-constraint. Hence no constraint can simultaneously be a v_i-constraint and a v_j-constraint. Furthermore, $\mathcal{L}$ cannot have both the constraints $x_{i,j} = 0$ and $x_{i,j} = 1$, since that would make W empty (and not an affine subspace) since no x can satisfy them both.

Let $\{\langle \ell_v, x \rangle = a_v\}_{v \in \{v_1, \ldots, v_k\}}$ be a list of v_i-constraints in $\mathcal{L}$, one for each $i \in [k]$. From the above set, we define $L = \{\ell_v\}_{v \in \{v_1, \ldots, v_k\}}$. Note that L has k distinct elements. In what follows, each linear form will be seen as an element of $\mathbb{F}_2^n$. We argue that the dimension of $\mathrm{span}(L)$ is at least $2k/3$, to conclude that W has co-dimension at least $2k/3$.

Let $B \subseteq L$ be a basis of $\mathrm{span}(L)$, $|B| = b$. Let L_B denote $L \setminus B$. We make the two following simple claims which together easily imply our lemma.

Claim 4.3.5. *Let $\ell_r \in L_B$, such that $\ell_r = \ell_1 + \cdots + \ell_s$, where $\ell_i \in B$ for each $i \in [s]$. Then, the set $B \setminus \{\ell_1, \ldots, \ell_s\}$ spans every element in $L_B \setminus \{\ell_r\}$.*

Claim 4.3.6. *Let $\ell_r \in L_B$ and $B_0 \subseteq B$, such that $\ell_r \in \mathrm{span}(B_0)$. Then, $|B_0| \geq 2$.*

Before proving the above two claims, let us use them to establish our lemma. Pick any $\ell \in L_B$. Then, find the minimal $B_0 \subseteq B$ such that $\ell \in \mathrm{span}(B_0)$. By Claim 4.3.6, $|B_0| \geq 2$. Now, shrink B and L_B by deleting B_0 and ℓ from them, respectively. Then, by Claim 4.3.5, the shrunk B still spans the new L_B. Hence, we can repeat the above step to shrink L_B this way at least $k - b$ times before it becomes empty. At each step B shrinks in size by at least 2. Thus, $b \geq 2(k-b)$, yielding $b \geq \lceil 2k/3 \rceil$.

All that is left is to establish the two claims. Let us begin by proving Claim 4.3.5. First, consider the vertices $v_1, \ldots, v_s, v_r$, where ℓ_i is the linear form of the v_i-constraint. By our assumption, $\ell_1 + \cdots + \ell_s + \ell_r = 0$. The variable $x_{i,j}$ can be supported only by the constraints ℓ_i and ℓ_j. It cannot be supported in only one of them, since then it will appear in the sum which should be 0. So if $x_{i,j}$ is supported by ℓ_i and ℓ_i appears in the sum, then ℓ_j must also appear in the sum and it must also support $x_{i,j}$. Hence, consider the undirected graph G_r with vertex-set $\{v_1, \ldots, v_s\} \cup \{v_r\}$, where edge (v_i, v_j) is present iff $x_{i,j}$ is supported by ℓ_i (and hence by ℓ_j). We show, using the fact that B is a basis, that G_r is connected. Take any connected component G' of G_r. Consider $\sum_{i : v_i \in G'} \ell_i$. For every edge (v_i, v_j) in G', the

variable $x_{i,j}$ is added twice and so the sum will be 0. If there is more than one connected component, one of the components will not include v_r and will be contained in $\{v_1, \ldots, v_s\}$. This will contradict the fact that B is a basis. So there is only one connected component, $\{v_1, \ldots, v_s\} \cup \{v_r\}$. This also means that in any non-trivial linear sum of linear forms in L that is identically zero, ℓ_i participates for any $i \in \{1, \ldots, s\} \cup \{r\}$ iff each of $\ell_1, \ldots, \ell_s$ and ℓ_r participate (since every connected component that includes v_i must include all the others). Now take any $\ell \in L_B \setminus \{\ell_r\}$. Because B is a basis, there is a linear sum of elements just from B that equals ℓ. If any element from $\{\ell_1, \ldots, \ell_s\}$ participates in this, then by the above argument ℓ_r will also participate, yielding a contradiction as $\ell_r \notin B$.

Finally, we prove Claim 4.3.6. For the sake of contradiction, let $B_0 = \{\ell_j\}$. Then, $\ell_r = \ell_j$. But we know that the elements of L are distinct, so this is not possible. □

We now observe that the lower bound on the co-dimension obtained in Lemma 4.3.4 is tight.

Claim 4.3.7. *Let $m \geq 3$. Fix any $k \leq m$. There is an affine subspace W of co-dimension $\lceil 2k/3 \rceil$ such that the vertices $V = \{v_1, v_2, \ldots, v_k\}$ are not sinks in any input in W.*

Proof. Let k' be the largest multiple of 3 less than or equal to k. For vertex $v_{k'+1}$, should it exist, we set $x_{(1,k'+1)}$ so that the $(v_1, v_{k'+1})$ edge is directed out of $v_{k'+1}$. We do the same for $v_{k'+2}$. Now we only need to ensure that none of $v_1, \ldots, v_{k'}$ are sinks.

Group these k' vertices into triples $(v_1, v_2, v_3), \ldots, (v_{k'-2}, v_{k'-1}, v_{k'})$. Consider the space W obtained as the solution space to the constraints defined below. For each triple (v_i, v_{i+1}, v_{i+2}), add the following two constraints to the constraint list of W.

- $x_{(i,i+1)} + x_{(i,i+2)}$ is set so that exactly one of the two edges (v_i, v_{i+1}) and (v_i, v_{i+2}) is directed out of v_i.
- $x_{(i,i+1)} + x_{(i+1,i+2)}$ is set so that exactly one of the two edges (v_i, v_{i+1}) and (v_{i+1}, v_{i+2}) is directed out of v_{i+1}.

The two constraints above are simultaneously satisfied if and only if v_i, v_{i+1}, v_{i+2} forms a cycle. Hence the above constraints ensure that none of $v_1, \ldots, v_{k'}$ are sinks. The total number of constraints is $2\lfloor k/3 \rfloor + (k \mod 3) = \lceil 2k/3 \rceil$. □

We now conclude that SINK has large parity kill number. We know that $\left\| \widehat{\mathsf{SINK}} \right\|_1 \leq \log m$. By concluding that $\mathsf{C}^{\oplus}_{\mathsf{min}}(\mathsf{SINK}) \geq 2m/3$, we refute Conjecture 3.3.9. This also shows that we cannot hope to use PDT leaf complexity to get a generic upper bound on the number of affine spaces needed to represent a function of small spectral norm, and that Shpilka, Tal and Volk's result [STV17] is nearly optimal in this regard.

Theorem 4.3.8.

1. $\forall m > 2$, $\mathsf{C}^{\oplus}_{\mathsf{min}}(\mathsf{SINK}) = \lceil 2m/3 \rceil$.

2. Any deterministic PDT computing SINK *has at least* $2^{2m/3}$ *leaves.*

Proof of Theorem 4.3.8. **Proof of Part 1:** Let W be an affine subspace of $\{0,1\}^{\binom{m}{2}}$ such that every input in W has a sink. Since the number of such inputs is at most $2^{\binom{m}{2}} \cdot m/2^{m-1}$, this means W must must have co-dimension at least $m - 1 - \log m$.

Now let W be an affine subspace of $\{0,1\}^{\binom{m}{2}}$ such that every input in W has no sink. By Lemma 4.3.4 (set $k = m$), W must have co-dimension at least $2m/3$.

By Claim 4.3.7, we see that there in fact is a monochromatic affine subspace of co-dimension $\lceil 2m/3 \rceil$.

Proof of Part 2: Every leaf of the PDT is a monochromatic affine subspace of co-dimension at most the depth of the leaf. From Part 1, we know that every leaf in a PDT computing SINK has to be at depth at least $2m/3$. Hence the number of leaves in any PDT computing SINK is at least $2^{2m/3}$. □

4.3.3 Randomized Parity Decision Trees

For brevity's sake in this section we will use f to refer to the function SINK on $\binom{m}{2}$ variables.

Given a randomized parity decision tree Π of cost c that computes f to within error ϵ, and any distribution μ on $\{0,1\}^k$, there exists a deterministic parity decision tree Π' of cost at most c such that $\Pr_{x\sim\mu}[\Pi'(x) \neq f(x)] \leq \epsilon$. This is because the error Π makes when the input is sampled from μ is the expectation of the errors of its constituent deterministic parity decision trees.

Let μ be the distribution defined as $\mu(x) = 1/2|f^{-1}(0)|$ for $x \in f^{-1}(0)$ and $\mu(x) = 1/2|f^{-1}(1)|$ for $x \in f^{-1}(1)$. μ places exactly half mass on 0-inputs and half mass on 1-inputs.

Lemma 4.3.9. *Let $\epsilon \leq 1/8$ be any constant. Any deterministic parity decision tree Π of cost c for f with error probability ϵ under the input distribution μ induces an affine subspace W such that $\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ and* co-dim$(W) \leq c$.

Proof. A deterministic parity decision tree gives a partition of the universe into at most 2^c labelled affine subspaces, each of co-dimension at most c, where we assume the label is 1 iff the affine subspace has a larger mass over its 1-inputs than over its 0-inputs. This assumption can only decrease the error. We note the following about the affine subspaces in the partition.

- Since the error probability of Π is ϵ, it cannot have 1-affine subspaces covering more than $\frac{1}{2} + \epsilon$ mass under the distribution μ.
- Affine subspaces W which have error $\geq 4\epsilon\mu(W)$ can only make up $1/4$ mass, to keep the total error below ϵ.

Hence, if $\epsilon \leq 1/8$ is a constant, Π must induce a 0-affine subspace W such that $\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ and co-dim$(W) \leq c$. □

Lemma 4.3.10 (Smallness of Biased Affine Subspaces)**.** *Let $\epsilon \leq 1/20$. Any affine subspace W that has $\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ satisfies*

$$\text{co-dim}(W) \geq m/3.$$

Assuming the above lemmata it immediately follows that $\mathsf{R}^{\oplus}_{1/20}(f) \geq m/3$. Since one can repeat a protocol in order to decrease the error, $\mathsf{R}^{\oplus}_{1/20}(f) = O(\mathsf{R}^{\oplus}_{1/3}(f))$. Hence assuming Lemma 4.3.10, we have the following theorem.

Theorem 4.3.11. $\mathsf{R}^{\oplus}_{1/3}(\mathsf{SINK}) \geq \Omega(m)$.

Since we know $\mathsf{spar}_{1/3}(f) \leq O(m^4)$ and $\left\|\widehat{f}\right\|_1 \leq m$, this shows exponential separations between $\mathsf{R}^{\oplus}_{1/3}(f)$ and $\log \mathsf{spar}_{1/3}(f)$ or $\log \left\|\widehat{f}\right\|_1$. We now prove Lemma 4.3.10.

Smallness of Biased Affine Subspaces

We want to show that any 0-biased affine subspace under μ must have large co-dimension. We do this in two steps.

- We show that any 0-biased affine subspace must have a very small fraction of 1 inputs.
- We then show that any affine subspace with a very small fraction of 1 inputs must have large co-dimension.

We formally state these two steps below and then prove them.

Claim 4.3.12. *Let $\epsilon \leq 1/8$. An affine subspace W such that $\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ must satisfy*

$$\frac{|W \cap f^{-1}(1)|}{|W|} < 10\epsilon \frac{|f^{-1}(1)|}{2^{\binom{m}{2}}}.$$

Claim 4.3.13. *If an affine subspace W satisfies*

$$\frac{|W \cap f^{-1}(1)|}{|W|} < \frac{1}{2}\frac{|f^{-1}(1)|}{2^{\binom{m}{2}}},$$

then $\text{co-dim}(W) \geq m/3$.

Proof of Claim 4.3.12. Since $\mu(W) = \mu(W \cap f^{-1}(1)) + \mu(W \cap f^{-1}(0))$,

$$\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W) \Leftrightarrow \mu(W \cap f^{-1}(1)) \leq \frac{4\epsilon}{1-4\epsilon}\mu(W \cap f^{-1}(0)).$$

Note that

$$\mu(W \cap f^{-1}(1)) = \frac{1}{2}\frac{|W \cap f^{-1}(1)|}{|f^{-1}(1)|} \text{ and}$$
$$\mu(W \cap f^{-1}(0)) = \frac{1}{2}\frac{|W \cap f^{-1}(0)|}{|f^{-1}(0)|} \leq \frac{1}{2}\frac{|W|}{|f^{-1}(0)|} < \frac{1.1}{2}\frac{|W|}{2^{\binom{m}{2}}}.$$

$$\text{Therefore,} \quad \frac{|W \cap f^{-1}(1)|}{|f^{-1}(1)|} < \frac{5\epsilon}{1-4\epsilon}\frac{|W|}{2^{\binom{m}{2}}}.$$

Cross-multiplying and using the assumption that $\epsilon \leq 1/8$,

$$\frac{|W \cap f^{-1}(1)|}{|W|} < 10\epsilon\frac{|f^{-1}(1)|}{2^{\binom{m}{2}}}.$$

□

Proof of Claim 4.3.13. Let S be the set of inputs that represent a graph with a sink, i.e. $S = f^{-1}(1)$. Let $S_i \subset S$ be the set of inputs in which the graph represented has vertex v_i as a sink.

Consider the set I of all $i \in [m]$ such that

$$\frac{|W \cap S_i|}{|W|} < \frac{|S_i|}{2^{\binom{m}{2}}}.$$

Then, by the condition assumed in the claim

$$\frac{1}{2}\frac{|S|}{2^{\binom{m}{2}}} > \frac{|W \cap S|}{|W|} = \sum_{i=1}^{m}\frac{|W \cap S_i|}{|W|} \geq \sum_{i \in \bar{I}}\frac{|W \cap S_i|}{|W|} \geq |\bar{I}|\frac{|S|/m}{2^{\binom{m}{2}}} = |\bar{I}|\frac{1}{m}\frac{|S|}{2^{\binom{m}{2}}},$$

where the first equality follows from the disjointness of the S_is, and the second inequality follows from the definition of I and the fact that $|S_i| = |S|/m$. Hence $|I| > m/2$, and for all $i \in I$,

$$\frac{|W \cap S_i|}{|W|} < \frac{|S_i|}{2^{\binom{m}{2}}} = 2^{-(m-1)}.$$

Fix any $i \in I$. Define the distribution W_{v_i} by the following sampling procedure: Sample an input uniformly at random from W and project it to E_{v_i}. We have the following fact about W_{v_i}.

$$\Pr_{X \sim W_{v_i}}[X = z_i] = \frac{|W \cap S_i|}{|W|} < 2^{-(m-1)}.$$

But by Claim 4.3.2, the number of extensions of z_i inside W is either 0 or at least $2^{\dim(W)-(m-1)}$. That is, $\Pr_{X \sim W_{v_i}}[X = z_i]$ is either 0 or at least $2^{-(m-1)}$. Given our fact about W_{v_i}, it must be 0.

So for all $i \in I$, v_i is never a sink in W. Since $|I| \geq m/2$, Lemma 4.3.4 implies that the co-dimension of W is at least $m/3$. □

Proof of Lemma 4.3.10. By chaining together Claim 4.3.12 and Claim 4.3.13 we get that if $\epsilon \leq 1/20$, then

$$\mu(W \cap f^{-1}(1)) < 4\epsilon\mu(W) \implies \text{co-dim}(W) \geq m/3.$$

□

4.4 SINK∘XOR is hard: Randomized Communication Complexity

Before we begin proving the lower bound, we mention some statements regarding entropy which will be used in our proofs.

Definition 4.4.1 (Entropy)**.** *Let X be a discrete random variable. The entropy $H(X)$ is defined as*

$$H(X) := \sum_{s \in \mathrm{supp(X)}} \Pr[X = s] \log \frac{1}{\Pr[X = s]}.$$

Fact 4.4.2 (Folklore)**.** $|\mathrm{supp}(X)| = k \implies H(X) \leq \log k$*, with equality if and only if X is uniform.*

Definition 4.4.3 (Relative Entropy)**.** *Let ν, μ be distributions over a finite set S of outcomes. The relative entropy (or Kullback-Liebler divergence) $d_{KL}(\nu \| \mu)$ is defined as*

$$d_{KL}(\nu \| \mu) := \sum_{s \in S} \nu(s) \log \frac{\nu(s)}{\mu(s)}.$$

Lemma 4.4.4 (Pinsker's Inequality)**.** *For two distributions ν, μ over the same set of outcomes,*

$$d_{KL}(\nu \| \mu) \geq \frac{1}{2 \ln 2} \|\nu - \mu\|_1^2.$$

The following claim appears in [Gav16].

Claim 4.4.5 (Two faraway distributions cannot both have near-maximum entropy, [Gav16])**.** *If ν_1 and ν_2 are two distributions in $\{0,1\}^n$, then $\min\{H(\nu_1), H(\nu_2)\}) \leq n - \|\nu_1 - \nu_2\|^2/(8 \ln 2)$.*

We reproduce a proof for completeness.

Proof. Let u be the uniform distribution over $\{0,1\}^n$ and d be $\|\nu_1 - \nu_2\|_1$. By the triangle inequality, $\|\nu_1 - u\|_1 + \|\nu_2 - u\|_1 \geq d$. Without loss of generality, assume $\|\nu_1 - u\|_1 \geq d/2$. From Pinsker's inequality (Lemma 4.4.4) we know that $d^2/4 \leq 2 \ln 2 \, d_{KL}(\nu_1 \| u)$. But

$$d_{KL}(\nu_1 \| u) = \sum_x \nu_1(x) \log \frac{\nu_1(x)}{2^{-n}} = \sum_x \nu_1(x) \left(\log \nu_1(x) + \log \frac{1}{2^{-n}} \right) = -H(\nu_1) + n.$$

Thus, $d^2 \leq 8 \ln 2 (n - H(\nu_1))$, or $H(\nu_1) \leq n - \|\nu_1 - \nu_2\|^2/(8 \ln 2)$. □

We next recall a lemma that has proved to be quite useful in counting combinatorial structures using 'the entropy method'.

Lemma 4.4.6 (Shearer's Lemma)**.** *Let $X = (X_1, \ldots, X_n)$ be a random variable. If P is a random variable (independent of X) distributed on subsets of the coordinates $[n]$, such that for every $i \in [n]$, $\Pr[i \in P] \geq t$, then $\mathbb{E}[H(X_P)] \geq tH(X)$ where X_P is the random variable $(X_i : i \in P)$.*

In this section we use F to refer to the function SINK ∘ XOR on $\binom{m}{2} + \binom{m}{2}$ variables in order to cut down on `Overfull \hbox`es.

The communication lower bound we show on SINK ∘ XOR will be via the Corruption bound (Lower bound 9). In particular we will be using the combinatorial version of it (Theorem 3.2.25).

We define the distribution ν as follows. $\nu(x) = 1/2|F^{-1}(0)|$ for $x \in F^{-1}(0)$ and $\nu(x) =$

$1/2|F^{-1}(1)|$ for $x \in F^{-1}(1)$. ν places exactly half mass on 0-inputs and half mass on 1-inputs.

Lemma 4.4.7 (Corruption Bound, analogous to Lemma 4.3.10)**.** *Let $\epsilon \leq 1/12$. Any rectangle R that has $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R)$ satisfies*

$$\nu(R) \leq 2^{-m(1/2-40\epsilon)^2/(64\ln 2)}.$$

Since ν is balanced, by Theorem 3.2.25 the above lemma implies that $\mathsf{corr}^0_{\epsilon/2}(F) \geq \frac{1}{2}(\frac{1}{2} - \epsilon)2^{m(1/2-40\epsilon)^2/(64\ln 2)}$. This in turn implies that $\mathsf{R}^{\mathsf{cc}}_{1/320}(F) \geq m(1/2-1/4)^2/(64\ln 2) - O(1) \geq \Omega(m)$. Since repeating a protocol is an efficient way to reduce error, $\mathsf{R}^{\mathsf{cc}}_{1/320}(F) = O(\mathsf{R}^{\mathsf{cc}}_{1/3}(F))$. Hence assuming Lemma 4.4.7, we have the following theorem.

Theorem 4.4.8. $\mathsf{R}^{\mathsf{cc}}_{1/3}(\mathsf{SINK} \circ \mathsf{XOR}) \geq \Omega(m)$.

Since we know $\mathsf{rank}_{1/3}(F) \leq O(m^4)$, $\mathsf{rank}^+_{1/3}(F) \leq O(m^5)$, $\left\|\widehat{F}\right\|_1 \leq m$ and $\gamma_2(F) \leq m$, this shows exponential separations between $\mathsf{R}^{\oplus}_{1/3}(f)$ and the above measures. Indeed, the separation is even between corruption and the above measures. Furthermore, since $\mathsf{rank}^+_\epsilon(\overline{F}) \geq \mathsf{srect}^0_{2\epsilon}(F)$ (Theorem 3.4.6) and $\mathsf{srect}^0_{2\epsilon}(F)$ is by definition (Lower bound 8) an upper bound on $\mathsf{corr}^0_{2\epsilon}(F)$, it follows that $\mathsf{rank}^+_{1/80}(\overline{F}) \geq \Omega(m)$. This answers a question posed by Lee [Lee12] who asked whether there can be a separation between approximate rank and approximate nonnegative rank.

We now prove Lemma 4.4.7. We want to show that any 0-biased rectangle under ν must have small ν-mass. We prove this in three steps.

- We show that any 0-biased rectangle must have a very small fraction (under the uniform distribution) of 1-inputs.
- We then show that any rectangle with a very small fraction of 1-inputs must be small.
- We finish the proof by showing that any 0-biased small rectangle must have small ν mass.

We formally state these three steps below and then prove them.

Claim 4.4.9 (Analogous to Claim 4.3.12)**.** *Let $\epsilon \leq 1/8$. A rectangle R such that $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R)$ must satisfy*

$$\frac{|R \cap F^{-1}(1)|}{|R|} < 10\epsilon \frac{|F^{-1}(1)|}{2^{2\binom{m}{2}}}.$$

Claim 4.4.10 (Analogous to Claim 4.3.13)**.** *If a rectangle $R = A \times B$ satisfies*

$$\frac{|R \cap F^{-1}(1)|}{|R|} < 10\epsilon \frac{|F^{-1}(1)|}{2^{2\binom{m}{2}}},$$

then $\min\{|A|, |B|\} \leq 2^{\binom{m}{2} - m(1/2-40\epsilon)^2/(64\ln 2)}$.

Claim 4.4.11. *If a rectangle R satisfies $\nu(R \cap F^{-1}(1)) \leq \nu(R)/3$, then $\nu(R) \leq |R|/2^{2\binom{m}{2}}$.*

The proof of Claim 4.4.9 is syntactically equivalent to the proof of Claim 4.3.12.

Proof idea of Claim 4.4.10: This proof goes via the following intuition.

1. The rectangle R has a very small fraction of sinks relative to its size.
2. Hence for many vertices, R has a very small fraction of those vertices as sinks.
3. Any vertex v that is a sink very rarely in R must have its E_v projections on Alice's side and Bob's side quite "different" from each other.
4. Hence, either Alice's or Bob's projections must be small.
5. All these projections being small for Alice, say, shows that A must be really small, thus completing the proof via Shearer's lemma.

We now formalize this intuition.

Proof of Claim 4.4.10. We mimic here the first few steps of the proof of Claim 4.3.13 for the corresponding lower bound on RPDT's. Let S be the set of inputs (x, y) such that $x \oplus y$ represents a graph with a sink, i.e. $S = F^{-1}(1)$. Let $S_i \subset S$ be the set of inputs in which the graph represented has vertex v_i as a sink.

Consider the set I of all $i \in [m]$ such that

$$\frac{|R \cap S_i|}{|R|} \leq 20\epsilon \frac{|S_i|}{2^{2\binom{m}{2}}}.$$

Then, by the condition assumed in the claim

$$10\epsilon \frac{|S|}{2^{2\binom{m}{2}}} > \frac{|R \cap S|}{|R|} = \sum_{i=1}^{m} \frac{|R \cap S_i|}{|R|} \geq \sum_{i \in \bar{I}} \frac{|R \cap S_i|}{|R|} > |\bar{I}| 20\epsilon \frac{|S|/m}{2^{2\binom{m}{2}}} = |\bar{I}| \cdot \frac{1}{m} \cdot 20\epsilon \frac{|S|}{2^{2\binom{m}{2}}},$$

where the second inequality follows from definition of I and the fact that $|S_i| = |S|/m$. Hence $|I| \geq m/2$, and for all $i \in I$,

$$\frac{|R \cap S_i|}{|R|} \leq 20\epsilon \frac{|S_i|}{2^{2\binom{m}{2}}} = 20\epsilon 2^{-(m-1)}.$$

Here, the proof departs from the corresponding $\mathsf{R}^{\oplus}$ lower bound. Fix any $i \in I$. Define the distribution A_{v_i} by the following sampling procedure. Sample an input uniformly at random from A and project it to E_{v_i}. Similarly define B_{v_i}. Define $B'_{v_i} = B_{v_i} \oplus z_i$.

We now show that the distributions A_{v_i} and B'_{v_i} are far apart, and hence one of them has a loss in entropy by Claim 4.4.5. We use the notation $\alpha \in_U S$ to denote that α is drawn uniformly at random from S.

$$\begin{aligned}
20\epsilon 2^{-(m-1)} &\geq \frac{|R \cap S_i|}{|R|} \\
&= \Pr_{X,Y \in_U R}[X_{v_i} \oplus Y_{v_i} = z_i] \\
&= \mathop{\mathbb{E}}_{X \in_U A}\left[\mathop{\mathbb{E}}_{Y \in_U B}\left[\mathbb{1}_{X_{v_i} \oplus Y_{v_i} = z_i}\right]\right] \\
&= \mathop{\mathbb{E}}_{X \in_U A}\left[\Pr_{Y \in_U B}[X_{v_i} \oplus Y_{v_i} = z_i]\right] \\
&= \mathop{\mathbb{E}}_{X \sim A_{v_i}}\left[\Pr_{Y \sim B_{v_i}}[X \oplus Y = z_i]\right] \\
&= \mathop{\mathbb{E}}_{X \sim A_{v_i}}\left[\Pr_{Y \sim B'_{v_i}}[X = Y]\right].
\end{aligned}$$

Let

$$T = \left\{ x \in \mathrm{supp}(A_{v_i}) \,\middle|\, \Pr_{Y \sim B'_{v_i}}[Y = x] \leq 40\epsilon 2^{-(m-1)} \right\}.$$

By Markov's inequality, $A_{v_i}(T) \geq 1/2$. But T is defined such that $B'_{v_i}(T) \leq 40\epsilon 2^{-(m-1)} \cdot |\mathrm{supp}(A_{v_i})| \leq 40\epsilon$. Hence,

$$\begin{aligned}
&\|A_{v_i} - B'_{v_i}\|_1 \geq 1/2 - 40\epsilon \\
\implies &(1/2 - 40\epsilon)^2 \leq 8 \ln 2 \cdot (m - 1 - \min\{H(A_{v_i}), H(B'_{v_i})\}) && \text{(by Claim 4.4.5)} \\
\implies &\min\{H(A_{v_i}), H(B'_{v_i})\} \leq m - 1 - (1/2 - 40\epsilon)^2/(8 \ln 2).
\end{aligned}$$

Note that the distributions B_{v_i} and B'_{v_i} are the same distribution but for a relabelling of the elements in its support. Hence $H(B_{v_i}) = H(B'_{v_i})$.

Either Alice's side or Bob's side hence experiences a loss in entropy for at least half the projections in I. Without loss of generality, we assume it is Alice's side. Since $|I| \geq m/2$, the expected entropy of A_{v_i} (when uniformly sampling $i \in [m]$) is at most $m - 1 - 1/4 \cdot (1/2 - 40\epsilon)^2/(8 \ln 2)$.

Note that each coordinate in Alice's input appears in exactly 2 out of the m projections. We now apply Shearer's lemma (Lemma 4.4.6) with $X \in_U A$ and P uniform over $\{E_{v_i}\}_{i \in [m]}$. We have $t = 2/m$ and $\mathbb{E}[H(X_P)] \leq m - 1 - (1/2 - 40\epsilon)^2/(32 \ln 2)$. Hence we can conclude that

$$H(X) \leq \frac{m}{2} \cdot \left(m - 1 - (1/2 - 40\epsilon)^2/(32 \ln 2)\right).$$

Since X is uniform over A, we also have

$$|A| \leq 2^{(m/2)\left(m - 1 - (1/2 - 40\epsilon)^2/(32 \ln 2)\right)} = 2^{\binom{m}{2} - m(1/2 - 40\epsilon)^2/(64 \ln 2)}.$$

□

Proof of Claim 4.4.11.

$$\begin{aligned}
\frac{|R|}{2^{2\binom{m}{2}}} &\geq \frac{|R \cap F^{-1}(0)|}{2^{2\binom{m}{2}}} \\
&\geq .9\frac{|R \cap F^{-1}(0)|}{|F^{-1}(0)|} \\
&= 1.8\nu(R \cap F^{-1}(0)) && \text{(by definition of } \nu\text{)} \\
&\geq \nu(R). && \text{(by assumption } \nu(R \cap F^{-1}(0)) \geq 2\nu(R)/3\text{)}
\end{aligned}$$

□

Proof of Lemma 4.4.7. By chaining together Claim 4.4.9, Claim 4.4.10 and Claim 4.4.11, we get that if $\epsilon \leq 1/12$ and $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R)$, then

$$\nu(R) \leq \frac{|R|}{2^{2\binom{m}{2}}} \leq 2^{-m(1/2-40\epsilon)^2/(64\ln 2)}.$$

□

4.4.1 A Variant of SINK

$\overline{\mathsf{SINK} \circ \mathsf{XOR}}$ demonstrated a function which had small approximate rank but large approximate nonnegative rank. However, this function's complement has small approximate nonnegative rank. In this subsection, we define a variant of $\mathsf{SINK} \circ \mathsf{XOR}$ which still has small approximate rank, but the approximate nonnegative rank of both this function and its complement are large.

Define the function $\mathsf{VARSINK} : \{0,1\}^{1+\binom{m}{2}} \to \{0,1\}$ as follows. We interpret the last $\binom{m}{2}$ variables exactly the way we did for SINK. The output of the function is given below, where b is the first bit and x is the remaining $\binom{m}{2}$ bits.

$$\mathsf{VARSINK}(b, x) = b \oplus \mathsf{SINK}(x).$$

Let $M_{0,i}(b, x)$ be the $0-1$ indicator that is 1 if and only if $b = 0$ and v_i is a sink in x. Similarly, let $M_{1,i}(b, x)$ be 1 if and only if $b = 1$ and v_i is a sink in x. Let $M_1(b, x)$ be 1 if and only if $b = 1$. Note that each of $M_{0,i}$ and $M_{1,i}$ and M_1 is a subcube. Also

$$\mathsf{VARSINK}(b, x) = \sum_{i=1}^{m} M_{0,i}(b, x) + M_1(b, x) - \sum_{i=1}^{m} M_{1,i}(b, x).$$

Hence $\left\|\widehat{\mathsf{VARSINK}}\right\|_1 \leq 2m + 1$ and so $\mathsf{rank}_{1/80}(M_{\mathsf{VARSINK} \circ \mathsf{XOR}}) \leq O(m^4)$.

Now, note that if we set $b = 0$, $\overline{\mathsf{VARSINK}}(0, x) = \overline{\mathsf{SINK}}(x)$. So

$$\mathsf{rank}^{+}_{1/80}(M_{\overline{\mathsf{VARSINK} \circ \mathsf{XOR}}}) \geq 2^{\Omega(m)}.$$

If we set $b = 1$, $\mathsf{VARSINK}(1, x) = \overline{\mathsf{SINK}}(x)$. And hence

$$\mathsf{rank}^{+}_{1/80}(M_{\mathsf{VARSINK} \circ \mathsf{XOR}}) \geq 2^{\Omega(m)}.$$

Chapter 5

Extensions based on SINK: Subspace Designs

This work was done in collaboration with Arkadev Chattopadhyay and Ankit Garg. A preprint is available online. [CGS20]

In this chapter, we look at some questions raised by the refutation of the Log-Approximate-Rank Conjecture in the previous chapter. We start by analyzing our SINK counterexample and see whether it has any relevance to the Log-Rank and Log-Approximate-Nonnegative Rank Conjectures.

5.1 On Extensions to LRC and LANRC

As described in the previous chapter, SINK is a disjoint union of a few subcubes. Each subcube composed with XOR is easy to compute via randomized communication, yet SINK $\circ$ XOR is hard. If the Log-Approximate-Rank Conjecture were true, then the communication complexity of SINK $\circ$ XOR would be polylogarithmic in the number of subcubes, and hence it is false. Similarly the Log-Rank Conjecture would imply that the disjoint union of easy functions must be easy, and so one could disprove it by providing a disjoint union of easy functions that is hard. However, such a function is not possible.

Theorem 5.1.1 (Implicit in [Yan91])**.** *Let $F = \sum_{i\in[m]} F_i$, where (1) the sets $F_i^{-1}(1)$ are disjoint, and (2) each F_i has deterministic communication complexity at most c. Then the deterministic communication complexity of F is at most $O((c + \log m)^2)$.*

Proof. We know from Lower bound 1 that each $F_i^{-1}(1)$ can be partitioned into at most 2^c rectangles. hence $F^{-1}(1)$ can be partitioned into at most $m2^c$ rectangles. Then by Theorem 3.2.2, the deterministic communication complexity of F is at most $O(\log(m2^c)^2)$. $\square$

We now look at the Log-Approximate-Nonnegative-Rank Conjecture. In this case, what would suffice to disprove the conjecture is a function f such that (1) $f^{-1}(1)$ is a disjoint union of m_1 subcubes, (2) $f^{-1}(0)$ a disjoint union of m_0 subcubes, and (3) $F := f \circ \mathsf{XOR}$ is hard for randomized communication complexity (larger than $\mathsf{polylog}(m_0, m_1)$). However such an f is also impossible if we want F to have complexity $n^{\Omega(1)}$. It is implicit in a work of Ehrenfeucht and Haussler [EH89] that such a function f does not exist. Indeed, their result implies that such an f does not exist even under the milder restriction that f and $\overline{f}$ can be represented as a union of a small number of *not necessarily pairwise disjoint* subcubes. We explicitly reprove this here, condensing their proof to only pertain to our question.

Lemma 5.1.2. *Let $f : \{0,1\}^n \to \{0,1\}$ and let $\mathcal{S}$ and $\mathcal{T}$ be sets of subcubes such that $f^{-1}(1) = \bigcup_{S\in\mathcal{S}} S$ and $f^{-1}(0) = \bigcup_{S\in\mathcal{T}} S$. Then $\mathsf{R}^{\oplus}_{(}f) \leq \tilde{O}((\log(|\mathcal{S}|) + \log(|\mathcal{T}|))^2 \log n))$.*

Note that this theorem implies that for any f satisfying points (1) and (2) above, $R_{1/3}(f \circ \mathsf{XOR}) \leq \tilde{O}((\log(m_0) + \log(m_1))^2 \log n))$.

Proof. It was proved in [EH89] that f has a decision tree with a few leaves. We start with reproducing this proof and then we show that this implies a small-cost RPDT.

Since $\bigcup_{S\in\mathcal{S}\cup\mathcal{T}} S = \{0,1\}^n$, it follows that one of these subcubes, say S, which is without loss of generality in $\mathcal{S}$, has size at least $2^n/(|\mathcal{S}| + |\mathcal{T}|)$. Let $\mathsf{fixed}(S)$ be the set of coordinates that S fixes. Since S is large, $|\mathsf{fixed}(S)|$ must be at most $\log(|\mathcal{S}| + |\mathcal{T}|)$. Now S is disjoint from every subcube in $\mathcal{T}$ since S contains only 1-inputs and the subcubes in $\mathcal{T}$ contain only 0-inputs. For S to be disjoint from a subcube T, there must be an $i \in \mathsf{fixed}(S) \cap \mathsf{fixed}(T)$ such that S sets x_i differently from how T sets x_i. Let $i = \arg\max_{i\in\mathsf{fixed}(S)} |\{T \in \mathcal{T} \mid x_i \text{ is set differently in } S \text{ and } T\}|$. By the pigeonhole principle, the ith bit must be set differently in at least $|\mathcal{T}|/\log(|\mathcal{S}| + |\mathcal{T}|)$ subcubes.

We create a decision tree for f by querying x_i. The answer to the query either disagrees with x_i's setting in S (in which case we know for sure that $x \notin S$) or it agrees with the setting (in which case we know $|\mathcal{T}|/\log(|\mathcal{S}| + |\mathcal{T}|)$ subcubes that x does not belong to). In the latter case, we call the answer a "shrinking" answer and we say that $\mathcal{T}$ "shrunk". (If we had assumed that the large subcube S was in $\mathcal{T}$ instead of $\mathcal{S}$, then it would be $\mathcal{S}$ that "shrunk".) Having queried x_i and received the answer, the function $f' : \{0,1\}^{n-1} \to \{0,1\}$ we are now computing satisfies $f^{-1}(1) = \bigcup_{S\in\mathcal{S}'} S$ and $f^{-1}(0) = \bigcup_{S\in\mathcal{T}'} S$ where either $|\mathcal{S}'| \leq |\mathcal{S}| - 1$ or $|\mathcal{T}'| \leq |\mathcal{T}|\,(1 - 1/\log(|\mathcal{S}| + |\mathcal{T}|))$. In either case, $\log(|\mathcal{S}| + |\mathcal{T}|) \geq \log(|\mathcal{S}'| + |\mathcal{T}'|)$. We recursively build the decision tree.

Note that the maximum height of the decision tree is at most n, since there are only n variables. Also note that every internal query in the tree has a designated "shrinking" answer and a "non-shrinking" answer, with the "shrinking" answer either "shrinking" $\mathcal{S}$ or $\mathcal{T}$. We can specify a leaf of the tree by specifying the path from the root to the leaf, each edge in the path being specified as either the "shrinking" edge or the "non-shrinking" edge. But, as

the following calculation shows, the number of "shrinking" edges in a path can be at most $(\log(|\mathcal{T}|) + \log(|\mathcal{S}|)) \log(|\mathcal{S}| + |\mathcal{T}|) + 2$.

The number of times $\mathcal{T}$ can "shrink" is at most $\lceil \log(|\mathcal{T}|) \log(|\mathcal{S}| + |\mathcal{T}|) \rceil$, since shrinking it that many times will reduce its size to at most

$$|\mathcal{T}| \left(1 - \frac{1}{\log(|\mathcal{S}| + |\mathcal{T}|)}\right)^{\log(|\mathcal{T}|) \log(|\mathcal{S}|+|\mathcal{T}|)} \leq |\mathcal{T}| e^{-\log(|\mathcal{T}|)} < 1$$

at which point the function must evaluate to 1.

Similarly $\mathcal{S}$ can "shrink" at most $\lceil \log(|\mathcal{S}|) \log(|\mathcal{S}| + |\mathcal{T}|) \rceil$ times. So the number of leaves is at most the number of n bit strings with at most $(\log(|\mathcal{T}|) + \log(|\mathcal{S}|)) \log(|\mathcal{S}| + |\mathcal{T}|) + 2$ 1s. This is at most $(n+1)^{(\log(|\mathcal{T}|)+\log(|\mathcal{S}|)) \log(|\mathcal{S}|+|\mathcal{T}|)+2}$, or $2^{O((\log(|\mathcal{T}|)+\log(|\mathcal{S}|))^2 \log n)}$.

We now show that a decision tree with N leaves can be simulated by an RPDT of depth $\tilde{O}(\log N)$. This follows by balancing the decision tree (see for example [KN97], Lemma 2.8). If, for every node v, one can efficiently compute whether the input will reach v, then one can efficiently find the leaf that the input will reach. Starting from the root, go down the tree while ensuring that the number of leaves below your node is at least $2N/3$. When this is no longer possible, one of the children of your node, say v, will have between $2N/3$ and $N/3$ leaves. Compute whether the input reaches v. If it does, consider the subtree rooted at v and recursively find out which of the at most $2N/3$ leaves the input will reach. If it does not reach v, consider the original tree with the subtree rooted at v removed, and recursively find out which of the at most $2N/3$ leaves the input will reach. This recursion must stop within $\lceil \log_{3/2} N \rceil$ steps.

In the case of a decision tree, whether or not an input will reach v is computed by a conjunction. It is well known that a randomized parity decision tree can compute a conjunction with error ϵ using at most $\lceil \log \epsilon \rceil$ queries. By setting $\epsilon = \lceil \log_{3/2} N \rceil / 3$, we get a randomized parity decision tree with error $\leq 1/3$ and cost $O(\log N \log \log N)$.

Hence $\mathsf{RPDT}_{1/3}(f) \leq \tilde{O}((\log(|\mathcal{T}|) + \log(|\mathcal{S}|))^2 \log n)$. □

A nice combinatorial consequence of the above is that if $\{0,1\}^n$ has a partition into c subcubes, then there is decision tree that refines this partition and has at most $2^{\log^3 c \log n}$ leaves. One can create such a tree by considering the function $g : \{0,1\}^n \to \{0,1\}^{\lceil \log c \rceil}$ that maps every input to the subcube it lies in. Each of the $\lceil \log c \rceil$ bits of the output is computed by a function whose 1-inputs and 0-inputs are both partitioned into at most c subcubes. Hence each of them has a decision tree with $2^{\log^2 c \log n}$ leaves. By composing these functions, we get a decision tree with $(2^{\log^2 c \log n})^{\lceil \log c \rceil}$ leaves.

However, this above theorem is specifically about subcubes. We need not restrict ourselves to subcubes. Affine subspaces (which we refer to simply as subspaces) of $\mathbb{F}_2^n$ are also as capable as subcubes for our purposes. Can we use subspaces instead of subcubes to get better results? For instance, the above theorem is not yet known for subspaces.

Conjecture 5.1.3: Affine Subspace Partition

Let $\mathcal{P}$ be a partition of $\{0,1\}^n$ into c affine subspaces. Then there is a parity decision tree that refines this partition and has only $2^{\mathsf{polylog}(c,n)}$ leaves.

The switch to subspaces has more potential than this. For instance, there is evidence that it can lead to stronger counterexamples of the Log-Approximate-Rank Conjecture, and also further close the gap between approximate rank and randomized communication complexity. The search for more counterexamples to the LARC is also important because any total function that exponentially separates randomized and quantum communication complexity must also refute the LARC.

To elaborate on how subspaces are used to provide evidence for stronger refutations of the LARC, we first look at some preliminaries about subspaces and then move on to the object du chapitre, Subspace Designs.

5.2 Subspace Preliminaries

5.2.1 Notation

Given a subspace $S \subseteq \mathbb{F}_2^n$, we use $\mathsf{dim}(S)$ to denote its dimension and $\mathsf{codim}(S)$ to denote its codimension i.e. $n - \mathsf{dim}(S)$. Given the standard bilinear form $\langle \cdot, \cdot \rangle$ on $\mathbb{F}_2^n$, we can define the dual space of S as the set $\{\ell \in \mathbb{F}_2^n \mid \forall x \in S \ \langle \ell, x \rangle = 0\}$. It is a subspace of dimension $n - \mathsf{dim}(S)$ and its dual space is S.

Given a subspace S of dimension k, fix a basis $L = \{\ell_1, \ldots, \ell_{n-k}\}$ of its dual space. For every point $a \in \mathbb{F}_2^{n-k}$, we can define the set $S_a^L = \{x \in \mathbb{F}_2^n \mid \forall i \in [n-k] \ \langle \ell_i, x \rangle = a_i\}$. These are called affine shifts, or cosets, of S. Sets of the kind S_a^L are also called affine subspaces. Each coset of S also has size 2^k. We can also define a coset map of S with respect to a basis of its dual space as

$$\mathsf{coset}_S^L(x) = (\langle \ell_1, x \rangle, \ldots, \langle \ell_{n-k}, x \rangle).$$

It is easy to see that the choice of basis for the dual space does not affect the set of cosets of S. It merely affects the string $a \in \mathbb{F}_2^{n-k}$ that is used to refer to a specific coset. Hence we will refer to the coset map as coset_S, and we may choose an arbitrary basis of the dual space of S in order to interpret the coset map.

From here on, we will use $\{0,1\}$ to refer to $\mathbb{F}_2$. The values 0 and 1 represent the additive and multiplicative identity of $\mathbb{F}_2$.

5.2.2 Basic facts about subspaces

Here we mention two facts about subspaces that will be useful.

Lemma 5.2.1 (Disjoint Subspaces)**.** *Let S be a subspace of $\{0,1\}^n$ of dimension d_1. Let T be a subspace of $\{0,1\}^n$ of dimension d_2 chosen uniformly at random. Then $\Pr_T[S \cap T = \{0\}] \geq 1 - n2^{d_1+d_2-n}$.*

Proof. Let us generate T by choosing d_2 vectors $\{v_1, \ldots, v_{d_2}\}$, each vector independent of the previous ones, in order to form a basis for T. The subspace S intersects T trivially if and only if for all $i \in [d_2]$, $v_i \notin \text{span}(\{v_j\}_{j<i} \cup S)$. We call these events $E_1, \ldots, E_{d_2}$. When choosing v_i to add to the basis for T, there are $2^n - 2^{i-1}$ choices, since $|\text{span}(\{v_j\}_{j<i})| = 2^{i-1}$. Conditioned on $E_1, \ldots, E_{i-1}$, we also know that $|\text{span}(\{v_j\}_{j<i} \cup S)| = 2^{i-1+d_1}$. The probability of E_i occurring is

$$\frac{|\left(\{0,1\}^n \setminus \text{span}(\{v_j\}_{j<i})\right) \setminus \text{span}(\{v_j\}_{j<i} \cup S)|}{|\{0,1\}^n \setminus \text{span}(\{v_j\}_{j<i})|} = \frac{|\{0,1\}^n \setminus \text{span}(\{v_j\}_{j<i} \cup S)|}{|\{0,1\}^n \setminus \text{span}(\{v_j\}_{j<i})|}.$$

We can then calculate the probability of $S \cap T = \emptyset$ as

$$\begin{aligned}\Pr\left[\bigcap_{i\in[d_2]} E_i\right] &= \prod_{i=1}^{d_2} \Pr\left[E_i \mid E_1, \cdots, E_{i-1}\right] = \prod_{i=1}^{d_2} \frac{2^n - 2^{d_1+i-1}}{2^n - 2^{i-1}} \\ &\geq \left(1 - \frac{2^{d_1+d_2}}{2^n}\right)^{d_2} \geq 1 - \frac{d_2}{2^{n-d_1-d_2}}.\end{aligned}$$

□

Lemma 5.2.2. *Let V and W be affine subspaces of $\{0,1\}^n$ satisfying*

$$\frac{|V \cap W|}{|W|} < \frac{|V|}{2^n}.$$

Then $V \cap W = \emptyset$.

Proof. Let $\{\langle v_i, x\rangle = a_i\}_{i\in[k]}$ be the constraints defining the affine subspace W. We define the affine subspaces $W_0, W_1, \cdots, W_k$ as follows. The constraints for W_j are $\{\langle v_i, x\rangle = a_i\}_{i\in[j]}$. Clearly $W_0 = \{0,1\}^n$ and $W_k = W$.

Now let us assume that $|V \cap W_i| \neq 0$ and is hence an affine subspace. The set $V \cap W_{i+1}$ is the same affine subspace with the added constraint $\langle v_{i+1}, x\rangle = a_{i+1}$.

- If this constraint was already implied by the constraints in $V \cap W_i$, then $|V \cap W_{i+1}| = |V \cap W_i|$.
- If this constraint is incompatible with the constraints in $V \cap W_i$, then $|V \cap W_{i+1}| = 0$.
- If this constraint was independent of the constraints in $V \cap W_i$, then $|V \cap W_{i+1}| = |V \cap W_i|/2$.

Hence $|V \cap W_k|$ is either 0 or is at least $|V \cap W_0|/2^k$. On the other hand, $|W|/2^n = 1/2^k$. Since $V \cap W_k = V \cap W$ and $V \cap W_0 = V$, we can rewrite this as

$$V \cap W \neq \emptyset \implies \frac{|V \cap W|}{|V|} \geq \frac{|W|}{2^n}.$$

□

We also will use a corruption lower bound in the context of Randomized PDTs. This is slightly simpler than the more general communication complexity corruption bound.

Lemma 5.2.3 (Corruption, RPDT version)**.** *Let $f : \{0,1\}^n \to \{0,1\}$. Let μ be a distribution on $\{0,1\}^n$ such that $\mu(f^{-1}(0)) = \frac{1}{2}$. Let $\epsilon \leq 1/8$. Then an ϵ-error cost-c RPDT computing f implies the existence of an affine subspace W such that*

- *$\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ and*
- *$\mathsf{codim}(W) \leq c$.*

Proof. Note that an ϵ-error cost-c RPDT T computing f implies that for any distribution μ over the inputs of f, there is an RPDT whose expected error, $\mathbb{E}_{T,x\sim\mu}[|T(x) - f(x)|]$, is at most ϵ. Since T is a distribution over deterministic parity decision trees, there is a deterministic parity decision tree whose expected error is also at most ϵ.

Suppose that a subspace such as the one posited in the lemma statement did not exist. Then for any cost-c parity decision tree T, we may compute the error made as follows. Note that the set of inputs that reach any specific leaf forms an affine subspace of codimension at most c, with each pair of such affine subspaces being disjoint. Let $\mathcal{L}$ be the set of these affine subspaces corresponding to the leaves of T that are labelled 0. Then $\sum_{V\in\mathcal{L}} \mu(V) \geq \frac{1}{2} - \epsilon$, since otherwise T would be outputting 1 on more than an ϵ mass of 0-inputs. But then $\sum_{V\in\mathcal{L}} \mu(V \cap f^{-1}(1)) \geq \sum_{V\in\mathcal{L}} 4\epsilon\mu(V) \geq 4\epsilon(\frac{1}{2} - \epsilon) \geq 2\epsilon - 4\epsilon^2 > \epsilon$. So on more than an ϵ mass of 1-inputs, T outputs 0. Hence the tree T is erring on a larger than ϵ mass of inputs and we have a contradiction. □

5.3 Subspace Designs

Definition 5.3.1 (Subspace Design)**.** *An n-dimensional (s,h)-subspace design is a set of subspaces $\{S_1, S_2, \ldots, S_m\}$ of $\{0,1\}^n$ such that for all subspaces T of dimension at most s, at most h of the m subspaces intersect T non-trivially.*

We call a set of subspaces $\{V_1, V_2, \ldots, V_m\}$ of $\{0,1\}^n$ an n-dimensional (s,h)-dual subspace design if their duals form an (s,h)-subspace design. Dual subspace designs have an alternate characterization based on the notion of independent subspaces.

Definition 5.3.2 (Independent Subspaces)**.** *Subspaces $S, T \subseteq \{0,1\}^n$ are independent if their coset maps are independent. That is, let L_S and L_T be arbitrary bases for the dual spaces of S and T. For a variable x chosen uniformly at random from $\{0,1\}^n$, consider the random variables $\mathsf{coset}_S(x)$ and $\mathsf{coset}_T(x)$. S and T are independent if and only if these two random variables are independent. More formally, for every $a \in \mathbb{F}_2^{\mathsf{codim}(S)}, b \in \mathbb{F}_2^{\mathsf{codim}(T)}$, we want that $\Pr[\mathsf{coset}_S(x) = a \wedge \mathsf{coset}_T(x) = b] = \Pr[\mathsf{coset}_S(x) = a] \Pr[\mathsf{coset}_T(x) = b] = 2^{-\mathsf{codim}(S)-\mathsf{codim}(T)}$.*

In particular this implies that every coset of S intersects with every coset of T.

We now state the alternate characterization of dual subspace designs.

Claim 5.3.3. *The set $\{V_1, V_2, \cdots, V_m\}$ of $\{0,1\}^n$ is an n-dimensional (s,h)-dual subspace design if and only if for all subspaces W of* codimension *at most s, at least $m-h$ of the m subspaces are* independent *from W.*

This claim follows from the following lemma relating trivial subspace intersections and independent subspaces.

Lemma 5.3.4 (Independent Subspaces)**.** *Subspaces S and T of $\mathbb{F}_2^n$ are independent if and only if the dual space of S and the dual space of T intersect trivially (i.e. only at the point $0 \in \mathbb{F}_2^n$).*

Proof. Let V and W be the dual spaces of S and T respectively. If V and W intersected at a non-zero point $\ell \in \mathbb{F}_2^n$, then consider bases L_S and L_T for V and W respectively, wherein ℓ is the first element of L_S and also the first element of L_T. The coset maps of S and T with this choice of L_S and L_T cannot be independent since for all $x \in \mathbb{F}_2^n$, the first entries of $\mathsf{coset}_S^{L_S}(x)$ and $\mathsf{coset}_T^{L_T}(x)$ will always agree.

For the other direction, let L_S and L_T be arbitrary bases for V and W respectively. We will show that if V and W intersect trivially, then the coset maps are independent. Assuming V and W intersect trivially, this means that $\mathrm{span}(L_S) \cap \mathrm{span}(L_T) = \{0\}$. Hence $L = L_S \cup L_T$ is an independent set of size $\mathsf{dim}(V) + \mathsf{dim}(W)$. Hence for a uniformly random point $x \in \mathbb{F}_2^n$ and any $a \in F_2^{\mathsf{codim}(S)}, b \in \mathbb{F}_2^{\mathsf{codim}(T)}$, $\Pr[\mathsf{coset}_S^{L_S}(x) = a \wedge \mathsf{coset}_T^{L_T}(x) = b] = 2^{-\mathsf{dim}(V)-\mathsf{dim}(W)} = \Pr[\mathsf{coset}_S^{L_S}(x) = a] \Pr[\mathsf{coset}_T^{L_T}(x) = b]$. □

A useful corollary of Claim 5.3.3 is that an (s,h)-dual subspace design also forms a hitting set for the set of all affine subspaces of codimension at most s. We will use this fact to lower bound the randomized parity decision tree complexity of unions of subspaces.

Corollary 5.3.5. *Let $\{V_1, V_2, \cdots, V_m\}$ be an n-dimensional (s,h)-dual subspace design. For all affine subspaces W of* codimension *at most s, at least $m-h$ of the m subspaces intersect with W.*

Proof. This follows from Claim 5.3.3 and the fact that if two subspaces S and T are independent, then S will intersect any affine shift of T non-trivially. □

5.3.1 RPDT Hardness of Subspace Designs

Theorem 5.3.6. *Let $\mathcal{V}$ be an n-dimensional (s,h)-dual subspace design of size m.*

Let f be the function defined as $f^{-1}(1) = \bigcup_{V\in\mathcal{V}} V$. We now show that $\mathsf{R}^{\oplus}_{\epsilon}(f) \geq s$ as long as $\epsilon < \frac{m-h}{8m}\frac{|f^{-1}(0)|}{2^n}$.

We will be instantiating this with dual subspace designs such that $h \ll m$ and $|f^{-1}(0)|/2^n \approx 1$. In such an instantiation, the condition on ϵ is just that it is a constant less than $1/8$.

Proof. Consider the distribution μ defined over the inputs of f as follows.

- Sample $z \sim_{\mathsf{unif}} \{0,1\}$.
- If $z = 0$, output a uniformly random input from $f^{-1}(0)$.
- Otherwise, sample $V \sim_{\mathsf{unif}} \mathcal{V}$.
- Output a uniformly random input from V.

Assuming that f is computed by an ϵ-error cost-c RPDT, Lemma 5.2.3 implies the existence of a subspace W such that

- $\mu(W \cap f^{-1}(1)) \leq 4\epsilon\mu(W)$ and
- $\mathsf{codim}(W) \leq c$.

Assume we have a W such that $\mu(W\cap f^{-1}(1)) \leq 4\epsilon\mu(W)$. This means that $\mu(W\cap f^{-1}(1)) \leq \frac{4\epsilon}{1-4\epsilon}\mu(W \cap f^{-1}(0))$. We also know the following from the definition of μ.

$$\mu(W \cap f^{-1}(1)) = \frac{1}{2}\cdot\frac{1}{|\mathcal{V}|}\sum_{V\in\mathcal{V}}\frac{|W\cap V|}{|V|}$$
$$\mu(W \cap f^{-1}(0)) = \frac{1}{2}\cdot\frac{|W\cap f^{-1}(0)|}{|f^{-1}(0)|} \leq \frac{1}{2}\cdot\frac{|W|}{|f^{-1}(0)|}$$

Putting these together, we get that

$$\frac{1}{|\mathcal{V}|}\sum_{V\in\mathcal{V}}\frac{|W\cap V|}{|V|} \leq \frac{4\epsilon}{1-4\epsilon}\frac{|W|}{|f^{-1}(0)|}.$$

Now if $\epsilon < \frac{m-h}{8m}\frac{|f^{-1}(0)|}{2^n} \leq \frac{1}{8}$, then $\frac{4\epsilon}{1-4\epsilon} < \frac{m-h}{m}\frac{|f^{-1}(0)|}{2^n}$. This implies that less than $m-h$ subspaces of $\mathcal{V}$ can satisfy $\frac{|W\cap V|}{|V|} \geq \frac{|W|}{2^n}$, and hence more than h of them *must* satisfy $\frac{|W\cap V|}{|V|} < \frac{|W|}{2^n}$. This means that $W \cap V = \emptyset$ (Lemma 5.2.2). In other words, W is an affine subspace that managed to evade more than h subspaces of $\mathcal{V}$. But by Corollary 5.3.5, if W is of codimension at most s, then it is disjoint from at most h subspaces of $\mathcal{V}$. So W must be of codimension more than s.

Hence the codimension of W, and thereby the cost of the RPDT, is at least s. □

Remark 5.3.7. *The above proof would also work for any union of affine subspaces which forms a hitting set for the set of all large affine subspaces the way that the dual subspace design does.*

5.4 A Mere Cubic Gap Between Approximate Sparsity and Randomized PDT Complexity

In this section, we instantiate Theorem 5.3.6 with random subspaces to get a mere cubic gap between RPDT complexity and approximate sparsity. It is known that there are efficient probabilistic constructions of subspace designs. In fact, efficient explicit deterministic constructions are also known [GK16, GXY17]. We go through a probabilistic construction here, and use it to show our main theorem.

Theorem 5.4.1. *Let $m = 100n$. Let $V_1, V_2, \ldots, V_m$ be subspaces of $\{0,1\}^n$ chosen independently and uniformly at random from the set of subspaces of dimension $2n/5$. With probability $1 - o(1)$ the following two statements are true.*

- *$\mathcal{V} = \{V_1, \ldots, V_m\}$ forms an $(n/5, m/10)$-dual subspace design.*
- *Every pair of subspaces in $\mathcal{V}$ intersects trivially.*

Proof. Let W be a fixed affine subspace of $\{0,1\}^n$ of dimension $4n/5$. Let $\mathcal{V} = \{V_1, \cdots, V_m\}$ be subspaces of $\{0,1\}^n$ chosen independently and uniformly at random from the set of subspaces of dimension $2n/5$.

Since the duals of W and V_1 have dimension $n/5$ and $3n/5$ respectively, the probability that W and V_1 are independent is at least $1 - n2^{-n/5}$ (Lemma 5.2.1). This is independently true of W and each $V \in \mathcal{V}$. The probability that W is not independent with *at least* $m/10$ of the m subspaces is at most $\binom{m}{m/10}(n2^{-n/5})^{m/10}$.

Since the number of subspaces of dimension $4n/5$ is at most $(2^n)^{4n/5} = 2^{4n^2/5}$, the probability that there exists such a subspace W that is not independent with at least $m/10$ of the subspaces in $\mathcal{V}$ is at most $2^{4n^2/5}\binom{m}{m/10}(n2^{-n/5})^{m/10}$.

Setting $m = 100n$, this upper bound is at most $2^{.8n^2+100n+10n\log n-2n^2} = o(1)$.

Hence with high probability, $\mathcal{V}$ is an $(n/5, m/10)$-dual subspace design.

Let f be defined as in the theorem statement. Note that since V_1 and V_2 are random subspaces of dimension $2n/5$, the probability that they intersect only at 0 is at least $1 - n2^{-n/5}$. The probability that any two subspaces in $\mathcal{V}$ intersect at more than just 0 is at most $\binom{m}{2}n2^{-n/5} = o(1)$. □

Theorem 5.4.2 (Separation)**.** *Let $m = 100n$. Let $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ be a set of subspaces of $\{0,1\}^n$ chosen independently and uniformly at random from the set of subspaces of dimension $2n/5$. Let f be the function that outputs 1 on the set $\bigcup_{V\in\mathcal{V}} V$. With probability $1 - o(1)$ the following two statements are true.*

- *Randomized parity decision tree complexity of f is at least $\Omega(n)$.*
- *The spectral norm of f (sum of absolute values of its Fourier coefficients) is upper bounded by $O(n)$ and its approximate Fourier sparsity is upper bounded by $O(n^3)$.*

Hence there exist functions which have a merely cubic gap between approximate Fourier sparsity and RPDT complexity.

Proof. We know from Theorem 5.4.1 that with probability $1 - o(1)$ the set $\mathcal{V}$ forms an $(n/5, m/10)$-dual subspace design. We also can trivially lower bound $|f^{-1}(0)|/2^n$ by $1-m2^{-3n/5}$. Since $\mathcal{V}$ is an $(n/5, m/10)$-dual subspace design, we can conclude from Theorem 5.3.6 that for $\epsilon \leq 1/10$, $\mathsf{R}^{\oplus}_{\epsilon}(f) \geq n/5$.

We also know from Theorem 5.4.1 that with probability $1-o(1)$, every pair of subspaces from $\mathcal{V}$ intersects trivially. When this event holds, f can be represented as $\sum_{V\in\mathcal{V}} \mathbb{1}_V - (m-1)\mathbb{1}_{V_0}$ where $V_0 = \{0\}$ is the trivial subspace of dimension 0. Since the spectral norm of a subspace is equal to 1, the spectral norm of f is upper bounded by $m+m-1 < 2m$. Using Theorem 3.4.2, this also implies that $\mathsf{spar}_{\epsilon}(f) \leq O(m^2 n/\epsilon^2) = O(n^3)$ for any constant ϵ.

This concludes the proof of the merely cubic gap. □

5.5 On Extending this to Communication

In this section, we state a plausible conjecture that would imply a lower bound on the randomized communication complexity of XOR compositions of our functions.

In the RPDT lower bound, we showed that in order for an affine subspace to avoid most of the subspaces of a dual subspace design, the codimension of the affine subspace needs to be large. We could hope for a similar statement in the communication world: For a rectangle to put very little mass on most of the subspaces making up a dual subspace design (i.e., puts very little mass on inputs (x, y) such that $x \oplus y$ lies in the subspaces), the mass of the rectangle must be $2^{-\Omega(n)}$. One particularly neat conjecture that would imply that statement is the following, in which $\mathcal{U}_k$ denotes the uniform distribution over k elements.

Conjecture 5.5.1. *There exist constants $0 < \alpha < 1$, $\beta > 0$ and $k \geq 1$ such that the following holds. Let $\mathcal{V} = \{V_1, \ldots, V_m\}$ be an n-dimensional (s, h)-dual subspace design. Let B_i be the coset map of V_i. Let X be a random variable over $\{0,1\}^n$ such that $\|B_i(X) - \mathcal{U}_{2^{\mathsf{codim}(V_i)}}\|_1 \geq \alpha$ for more than kh values of $i \in [m]$. Then $H(X) \leq n - \beta s$.*

The merely cubic gap in the RPDT world used random subspaces. So for extending it to communication, it would be okay for us to bypass dual subspace designs and prove the theorem for random subspaces instead.

Conjecture 5.5.2. *There exist constants $0 < \alpha < 1, \beta > 0$ such that the following holds. Let $m = 100n$. Let $V_1, V_2, \ldots, V_m$ be random subspaces of $\{0,1\}^n$ of dimension $2n/5$, and*

let $B_1, B_2, \cdots, B_m$ be their coset maps. Let X be a random variable over $\{0,1\}^n$ such that $\|B_i(X) - \mathcal{U}_{2^{3n/5}}\|_1 \geq \alpha$ for at least $m/3$ values of $i \in [m]$. Then with high probability, $H(X) \leq n - \beta n$.

First of all note that the conjectures are true when X is the uniform distribution over an affine subspace. To see this, suppose X is the uniform distribution over an affine subspace W. $H(X) \geq n - s$ is the same as saying that $\mathsf{codim}(W) \leq s$. Then by Claim 5.3.3, for at least $m - h$ of the subspaces $V_1, \ldots, V_m$, V_i and the dual space of W are independent, which implies that $B_i(X)$ will be exactly uniform ($\mathcal{U}_{2^{\mathsf{codim}(V_i)}}$).

We discuss now why the Conjectures 5.5.1 and 5.5.2 appear to be a bit tricky to prove. While the conjectures are true for affine subspaces, the number of distributions (or even the number of subsets of $\{0,1\}^n$) is much larger (doubly exponential in n), so the conjectures are a leap of faith in this sense. But we haven't been able to come up with counterexamples and it would be very interesting to do so. The conceptual way to view the conjectures, e.g. Conjecture 5.5.2 to be concrete, is that if a random variable X has the property that when projected down to $2n/5$ bits in various ways it loses $\Omega(1)$ bits of entropy, then X overall loses $\Omega(n)$ bits of entropy. Shearer's lemma talks about these kind of statements. While in Shearer's lemma, the projections are onto subcubes, there are generalizations called Brascamp-Lieb inequalities which talk about more general projections (e.g. see [Chr13]). However, the Brascamp-Lieb inequalities can at best guarantee an $\Omega(n/k)$-bit entropy loss in X if there is an $\Omega(1)$-bit entropy loss while projecting X to k bits in various ways. What we want is much stronger. This is one difficulty.

The other difficulty is that a Fourier type approach doesn't seem to work either. One can control $\|B_i(X) - \mathcal{U}_{2^{\mathsf{codim}(V_i)}}\|_1$ by bounding the ℓ_2 distance and then trying to bound the Fourier coefficients of the distribution of X on the dual space of V_i. But this doesn't give any meaningful bound (if done in a naive way at least).

We now state the lower bound on the randomized communication complexity of a dual subspace design composed with XOR that we get assuming Conjecture 5.5.1. For a set of subspaces in n dimensions $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$, let $f_{\mathcal{V}}$ be the function on n bits that outputs 1 on inputs in $\cup_{V \in \mathcal{V}} V$.

Theorem 5.5.3. *Let us assume Conjecture 5.5.1 holds with constants α, β and k. Let $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ be an n-dimensional (s,h)-dual subspace design and define γ so that $|\cup_{V \in \mathcal{V}} V| = \gamma 2^n$. Let $F = f_{\mathcal{V}} \circ \mathsf{XOR}$. For $\epsilon < \frac{(1-\alpha)^2}{4} \frac{m-2kh}{8m}(1-\gamma)$, the ϵ-error randomized communication complexity of F is at least $\beta s + \log(1-\gamma)$.*

Proof. We first prove the statement with $\alpha = \frac{1}{2}$. Hence we only assume $\epsilon < \frac{1}{16} \frac{m-2kh}{8m}(1-\gamma)$

For any $V \in \mathcal{V}$, let $S_V = \{(x,y) \in \{0,1\}^n \times \{0,1\}^n \mid x \oplus y \in V\}$. Note that $|S_V| = 2^n |V|$ and $F^{-1}(1) = \cup_{V \in \mathcal{V}} S_V$. Consider the distribution ν defined over the inputs of F as follows.

- Sample $z \sim_{\mathsf{unif}} \{0,1\}$.

- If $z = 0$, output a uniformly random input from $F^{-1}(0)$.
- Otherwise, sample $V \sim_{\mathsf{unif}} \mathcal{V}$.
- Output a uniformly random input from S_V.

Assuming F is computed by an ϵ-error cost-c communication protocol, Lower bound 9 implies the existence of a rectangle R such that

- $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R)$ and
- $\nu(R) \geq 2^{-c-3}$.

Assume we have an R such that $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R)$. This means that $\nu(R \cap F^{-1}(1)) \leq \frac{4\epsilon}{1-4\epsilon}\nu(R \cap F^{-1}(0))$. We also know the following from the definition of ν.

$$\nu(R \cap F^{-1}(1)) = \frac{1}{2} \cdot \frac{1}{|\mathcal{V}|} \sum_{V \in \mathcal{V}} \frac{|R \cap S_V|}{|S_V|}$$
$$\nu(R \cap F^{-1}(0)) = \frac{1}{2} \cdot \frac{|R \cap F^{-1}(0)|}{|F^{-1}(0)|} \leq \frac{1}{2} \cdot \frac{|R|}{|F^{-1}(0)|}$$

Putting these together, we get that

$$\frac{1}{|\mathcal{V}|} \sum_{V \in \mathcal{V}} \frac{|R \cap S_V|}{|S_V|} \leq \frac{4\epsilon}{1-4\epsilon} \frac{|R|}{|F^{-1}(0)|}.$$

Now if $\epsilon < \frac{m-2kh}{128m} \frac{|F^{-1}(0)|}{2^{2n}} < 1/8$, then $\frac{4\epsilon}{1-4\epsilon} < \frac{m-2kh}{16m} \frac{|F^{-1}(0)|}{2^{2n}}$. This implies that less than $m - 2kh$ subspaces of $\mathcal{V}$ can satisfy $\frac{|R \cap S_V|}{|S_V|} \geq \frac{|R|}{16 \cdot 2^{2n}}$, and hence more than $2kh$ of them *must* satisfy $\frac{|R \cap S_V|}{|S_V|} < \frac{|R|}{16 \cdot 2^{2n}}$. Let us fix such a V.

Let coset_V denote the function $\mathsf{coset}_V^{L_V}$ for some fixed basis L_V of the dual space of V. Let $R = A \times B$. Then $\frac{|R \cap S_V|}{|R|}$ is the probability that, when x and y are sampled uniformly at random from A and B, $\mathsf{coset}_V(x) = \mathsf{coset}_V(y)$. Let A_V be the distribution of $\mathsf{coset}_V(x)$ and B_V be the distribution of $\mathsf{coset}_V(y)$. The condition $\frac{|R \cap S_V|}{|R|} < \frac{|S_V|}{16 \cdot 2^{2n}}$ can be rewritten as

$$\Pr_{x' \sim A_V, y' \sim B_V}[x' = y'] < \frac{|S_V|}{16 \cdot 2^{2n}} = \frac{1}{16 \cdot 2^{\mathsf{codim}(V)}}.$$

It follows that $A_V(S) < 1/4$ where $S = \{y' \mid B_V(y') \geq \frac{1}{4 \cdot 2^{\mathsf{codim}V}}\}$. However, $B_V(S)$ must be at least $3/4$, since $B_V(\overline{S}) \leq 1/4$.

Hence A_V and B_V have total variational distance at least $\frac{1}{2}$, and $\|A_V - B_V\|_1 \geq 1$. By the triangle inequality, $\max\{\|A_V - \mathcal{U}_{2^{\mathsf{codim}(V)}}\|_1, \|B_V - \mathcal{U}_{2^{\mathsf{codim}(V)}}\|_1\} \geq \frac{1}{2}$.

Hence, either there are more than kh subspaces that satisfy $\|A_V - \mathcal{U}_{2^{\mathsf{codim}(V)}}\| \geq \frac{1}{2}$ or there are more than kh subspaces that satisfy $\|B_V - \mathcal{U}\| \geq \frac{1}{2}$. Without loss of generality we assume the former. Now we use our conjecture. The conjecture implies that $H(A) \leq n - \beta s$. Hence $\frac{|R|}{2^{2n}} \leq 2^{-\beta s}$.

We now want to move from $|R|$ being small under the uniform distribution to R being small under ν. We know that $\nu(R \cap F^{-1}(1)) \leq 4\epsilon\nu(R) < \nu(R)/2$, so $\nu(R \cap F^{-1}(0)) \geq \nu(R)/2$. We also know from the definition of ν that

$$\nu(R \cap F^{-1}(0)) = \frac{|R \cap F^{-1}(0)|}{2|F^{-1}(0)|} \leq \frac{|R|}{2 \cdot 2^{2n}} \cdot \frac{2^{2n}}{|F^{-1}(0)|} \leq 2^{-\beta s - 1} \cdot \frac{1}{1-\gamma}.$$

So $\nu(R) \leq 2\nu(R \cap F^{-1}(0)) \leq 2^{-\beta s - 1 - \log(1-\gamma)}$. Hence the cost of the protocol is at least $\beta s + \log(1-\gamma) - 3$.

This concludes the proof with $\alpha = \frac{1}{2}$. We now explain how to modify the proof assuming the conjecture were true for other values of α. Here we revert to $\epsilon < \frac{(1-\alpha)^2}{4} \frac{m-2kh}{8m}(1-\gamma)$. The proof would go through as it does above, analyzing a rectangle $R = A \times B$.

- We would find more than $2kh$ subspaces V such that $\Pr[A_V = B_V] < \frac{(1-\alpha)^2}{4} \frac{|S_V|}{2^{2n}}$ as is done in the above proof.
- We would then set $S = \{y' \mid B_V(y') \geq \frac{1-\alpha}{2 \cdot 2^{\mathsf{codim}(V)}}\}$. This would mean that $A_V(S) \leq \frac{1-\alpha}{2}$ and $B_V(S) \geq 1 - \frac{1-\alpha}{2}$. Hence $\|A_V - B_V\|_1 \geq 2\alpha$, and one of A or B (wlog, A) satisfies $\|A_V - \mathcal{U}_{2^{\mathsf{codim}(V)}}\|_1 \geq \alpha$ for at least kh subspaces from the dual subspace design.
- The proof would continue as it does above, using the conjecture to conclude that the cost of the protocol would be at least $\beta s + \log(1-\gamma) - 3$, which is $\Omega(s)$ for constant γ.

□

Given this lower bound, we would want to apply it to get a merely cubic gap between randomized communication complexity and approximate rank along the lines of Theorem 5.4.1.

Corollary 5.5.4. *Let $\mathcal{V} = \{V_1, V_2, \ldots, V_m\}$ be an $(n/5, m/20k)$-dual subspace design with (1) $m = 200kn$, (2) each subspace having dimension $2n/5$ and (3) every pair of subspaces intersecting trivially. Let $F = f_{\mathcal{V}} \circ \mathsf{XOR}$. Then assuming Conjecture 5.5.1,*

- *The $1/10$-error randomized communication complexity of F is $\Omega(n)$.*
- $\mathsf{rank}_{1/10}(F) = O(n^3)$.

Proof. The size of $F^{-1}(1)$ would be at most $2^n \sum_{V \in \mathcal{V}} |V| \leq 2^{n+2n/5} m = o(2^{2n})$. We can then use Theorem 5.5.3 to get a lower bound of $\beta n/5$ when $\epsilon < \frac{(1-\alpha)^2}{4} \frac{m-2kh}{8m} \frac{|F^{-1}(0)|}{2^{2n}}$, which is a constant. Since we can use error reduction to go from error $1/10$ to any small constant error with only a constant blow-up in cost, the $1/10$-error randomized communication complexity is also $\Omega(n)$.

The ϵ-approximate rank of $f \circ \mathsf{XOR}$ is known to be at most the ϵ-approximate sparsity of f. As analyzed in Theorem 5.4.1, $\left\|\widehat{f_{\mathcal{V}}}\right\|_1 \leq 2m$ and $\mathsf{spar}_{1/10}(f_{\mathcal{V}}) \leq O(m^2 n) = O(n^3)$ and hence $\mathsf{rank}_{1/10}(F) \leq O(n^3)$. □

The existence of a dual subspace design as required in the previous corollary follows by changing Theorem 5.4.1 to set $m = 200kn$. The proof of the modified statement is syntactically identical to the proof of the original statement.

Chapter 6

Quantum First-Order Convex Optimization

This work was done in collaboration with Ankit Garg, Robin Kothari and Praneeth Netrapalli. It is to appear at ITCS '21. Currently a preprint is available online. [GKNS20]

Section 6.5 in this chapter does not appear in any publication.

Recall that in the introduction we defined the measure $C_{\mathcal{M}}(n,\epsilon) = \min_{\mathcal{A}\in Alg_{\mathcal{M},n,\epsilon}} cost(\mathcal{A})$ where $\mathcal{M} \in \{det, rand, quantum\}$ and

$$\begin{aligned} Alg_{\mathcal{M},n,\epsilon} = \{&\mathcal{A} \text{ is an } \mathcal{M} \text{ query algorithm making function value and subgradient queries } | \\ &\forall \text{ convex, 1-Lipschitz } f:\mathbb{R}^n \to \mathbb{R}, \text{ any subgradient oracle for } f, \\ &\mathcal{A} \text{ outputs an } \epsilon\text{-optimal point of } f \text{ within the unit ball}\}. \end{aligned}$$

In this chapter we mainly show lower bounds on $C_{quantum}(n,\epsilon)$. There are also some lower bounds on $C_{rand}(n,\epsilon)$ that are included for their simplicity and the fact that they are tight up to constant factors in a larger regime than seems to be previously proven in the literature. The most impactful theorem is perhaps that the classical gradient descent algorithm is optimal among quantum algorithms as well.

Before we get to the lower bounds we first discuss the task of convex optimization and why we only care about optimizing 1-Lipschitz functions in the unit ball.

6.1 A bit about convex optimization and gradient descent

Let's start with the unconstrained convex minimization problem for a convex function $f : \mathbb{R}^n \to \mathbb{R}$. Here we want to find an $x \in \mathbb{R}^n$ that's ϵ-close to minimizing the function f. More

precisely, if we let $x^* := \arg\min_{x\in\mathbb{R}^n} f(x)$, then our goal is to find any $x \in \mathbb{R}^n$ such that $f(x) - f(x^*) \leq \epsilon$.

To obtain algorithms that are very general, this problem is often studied in the setting of black-box optimization. Here we do not assume any particular structure of the function f (e.g., that f is a low-degree polynomial), and only assume that we have some efficient method of computing f by an algorithm or circuit. In other words, we view f as a black box.

If we only had access to a black-box computing f, this would be zeroth-order optimization. In first-order optimization, we additionally assume we can also compute the gradient of f, or more precisely, since the gradient may not exist, we assume we can compute some subgradient of f. We call this oracle the *first-order oracle* and denote it by $\mathcal{FO}(f)$. When queried at any point $x \in \mathbb{R}^n$, it returns some vector $g_x \in \mathbb{R}^n$ that satisfies for all $y \in \mathbb{R}^n$,

$$f(y) \geq f(x) + \langle g_x, y - x\rangle . \tag{6.1}$$

In this work we consider arbitrary convex functions that are not necessarily smooth,[1] and so we cannot assume that the gradient exists. Our goal is to solve the function minimization problem while minimizing the number of calls or queries to the black boxes for f and some subgradient of f.

One might wonder why we consider queries to f and the subgradient of f to cost the same. This assumption is justified in many practical situations because of the *cheap gradient principle* [GW08], which says that "the cost to evaluate the gradient ∇f is bounded above by a small constant times the cost to evaluate the function itself." This provably holds in many models of computation; E.g., for arithmetic circuits over $+$ and $\times$, it can be proved that the complexity of computing the gradient is at most 5 times the complexity of computing f [BS83]. The conversion of source code computing f to code computing ∇f can often be done automatically in many programming languages, and such methods are called *automatic differentiation* or *algorithmic differentiation* [GW08]. These same principles essentially carry over to the computation of subgradients [KL18]. In the quantum setting, there is additional motivation to assume that a function and its gradient cost roughly the same since we can obtain the gradient (or a subgradient) of a function from a black-box computing the function, as shown in a sequence of papers that make increasingly weaker assumptions on the function oracle [Jor05, GAW19, CCLW20, vAGGdW20].

Now that we have black-box access to f and $\mathcal{FO}(f)$, we also need a starting point $x_0 \in \mathbb{R}^n$ to begin our search for a minimum. We require this to be an input, and the complexity will depend on how close this is to x^*, since otherwise the interesting portion of the function where the minimum is achieved might be hiding in some small corner of $\mathbb{R}^n$ that we cannot efficiently locate with only black-box access. Since we can easily shift the function by a fixed vector,

[1] In the optimization literature, a *smooth function* is a function that is differentiable everywhere in its domain, so the gradient is well defined, and whose gradient has bounded Lipschitz constant.

without loss of generality we assume $x_0 = \vec{0}$ is the origin. Let the distance between $x_0 = \vec{0}$ and x^* be $R := \|x^*\|$.[2] For convenience, we will assume that R is part of the input as well, although this can be relaxed by binary searching for the correct value of R.

Finally, it is also reasonable that the complexity of our algorithms depend on how quickly f can change, since the value of f at some point only constrains its values at nearby points if the function does not change too rapidly. Let G be an upper bound on the Lipschitz constant of f, meaning that for any two points x, y, $\|f(x) - f(y)\| \leq G\|x - y\|$. We assume G is part of the input as well.

We are now ready to formally define the first-order convex minimization problem in the black-box setting. We use $B(x, R) := \{y : \|x - y\| \leq R\}$ to denote an ℓ_2-ball of radius R around x.

Problem 6.1.1 (First-order convex minimization)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ have Lipschitz constant at most G on $B(\vec{0}, R)$, and let*

$$x^* := \arg\min_{x \in B(\vec{0},R)} f(x). \tag{6.2}$$

Then given n, G, R, and $\epsilon > 0$, the goal is to output a solution $x \in B(\vec{0}, R)$ such that $f(x) - f(x^*) \leq \epsilon$ while minimizing the number of queries to f and $\mathcal{FO}(f)$.

For simplicity, we assume that these oracles output real numbers to arbitrarily many bits of precision. Note that we allow algorithms to query the function and gradient oracles at any point in $\mathbb{R}^n$ even though the domain we are minimizing over is $B(\vec{0}, R)$. Since the main results of this section are lower bounds, these only make our results stronger.

Although the problem seems to involve 4 parameters, the parameters G, R, and ϵ are not independent since we can rescale the input and output spaces of f and assume $G = 1$ and $R = 1$ without loss of generality. Given an f with Lipschitz constant G, being optimized in a ball of radius R, and with optimization accuracy ϵ, we can equivalently minimize $\hat{f}(x) := \frac{1}{GR} f(Rx)$ over the unit ball with optimization accuracy of $\frac{\epsilon}{GR}$. Hence the complexity is really only a function of n and GR/ϵ.

6.1.1 Classical algorithms for first-order convex minimization

Gradient descent, or in this case subgradient descent, is a simple algorithm that starts from a point x_0 and takes a small step (governed by a step size η) in the opposite direction of the subgradient returned at x_0. Intuitively this brings us closer to the minimum since we are stepping in the direction where f decreases the most.

We can now describe the performance of subgradient descent for Problem 6.1.1. Since this is a constrained optimization problem, we use the projected subgradient descent algorithm, which is subgradient descent with the added step of projecting the current vector back onto the ball $B(\vec{0}, R)$ after every step.

[2] If x^* is not unique, we can let R be the distance between x_0 and the closest x^* to it.

Theorem 6.1.2 (Complexity of projected subgradient descent)**.** *The projected subgradient descent algorithm solves Problem 6.1.1 using $O((GR/\epsilon)^2)$ queries to f and $\mathcal{FO}(f)$.*

Observe that the query complexity of this algorithm, i.e. the number of queries made by the algorithm, is independent of n.[3] This is quite surprising at first and partly explains why gradient descent and its variants are popular in high-dimensional applications. More generally, we call such algorithms *dimension-independent algorithms*.

There also exist dimension-dependent algorithms for Problem 6.1.1 that work well when n is small. For example, the center of gravity method [Bub15] solves this problem with $O(n \log(GR/\epsilon))$ queries, which is very reasonable when n is small (and the algorithm is very efficient in terms of ϵ). In this work we focus on dimension-independent algorithms and assume that n is polynomially larger than the other parameters in the problem.

When n is large, we cannot improve over projected subgradient descent (Theorem 6.1.2) using any deterministic or randomized algorithm. We reprove the (well known) optimality of this algorithm among deterministic and randomized algorithms. This result is presented in Section 6.3.

Theorem 6.1.3 (Randomized lower bound)**.** *For any G, R, and ϵ, there exists a family of convex functions $f : \mathbb{R}^n \to \mathbb{R}$ with $n = O((GR/\epsilon)^2)$, with Lipschitz constant at most G on $B(\vec{0}, R)$, such that any classical (deterministic or bounded-error randomized) algorithm that solves Problem 6.1.1 on this function family must make $\Omega((GR/\epsilon)^2)$ queries to f or $\mathcal{FO}(f)$ in the worst case.*

This lower bound on query complexity has been shown in several prior works [NY83, WS17, BJL+19], but we believe our proof is simpler and the dimension n required in our proof seems to be smaller than that in prior works. Note that while several expositions of gradient descent prove the lower bound for deterministic algorithms, very few sources establish a lower bound against randomized algorithms.

Our lower bound uses the following hard family of functions: For any $z \in \{-1, +1\}^n$, let $f_z(x_1, \ldots, x_n) = \max_{i \in [n]} z_i x_i$,[4] where $n = O(1/\epsilon^2)$. These functions are convex with Lipschitz constant 1. We show that finding an ϵ-approximate minimum within $B(\vec{0}, 1)$ requires $\Omega(n)$ queries to the oracles. We establish the lower bound by showing that with high probability, every query of a randomized algorithm only reveals $O(1)$ bits of information about the string z, but an ϵ-approximate solution to this problem allows us to reconstruct the string z, which has n bits of information.

[3] Of course, the time complexity of implementing this algorithm will be at least linear in n since each query to either oracle requires us to manipulate a vector of length n.

[4] We use $[n]$ to denote the set of positive integers less than or equal to n, i.e., $[n] := \{1, \ldots, n\}$.

6.1.2 Quantum algorithms for first-order convex minimization

We now turn to quantum algorithms for solving Problem 6.1.1. In the quantum setting, we have quantum analogues of these oracles. There is a straightforward generalization of any oracle to the quantum setting, which makes the classical oracle reversible and then allows queries in superposition to this oracle. This quantum generalization of the oracle is justified by the fact that if we had a classical circuit or algorithm computing a function f, then it is possible in a completely black-box manner to construct the quantum oracle corresponding to f, and this oracle would then support superposition queries. We discuss quantum oracles in more detail in Section 6.4, but for now it is sufficient to consider them as computing the same functions as the classical oracles, except that they can additionally be queried in superposition.

At first, it might seem that since gradient descent is a sequential, adaptive algorithm where each step depends on the previous one, there is little hope of quantum algorithms outperforming gradient descent. On the other hand, consider the hard family of functions described above that witnesses the classical randomized lower bound in Theorem 6.1.3. While this is hard for classical algorithms, we show in Section 6.3.1 that there is a quantum algorithm that solves the problem on this family obtaining a quadratic speedup over any classical algorithm (and in particular, over gradient descent).

Theorem 6.1.4 (Quantum algorithm for classically hard function family)**.** *There is a quantum algorithm that solves Problem 6.1.1 on the class of functions that appear in the classical lower bound of Theorem 6.1.3 using $O(GR/\epsilon)$ queries to the oracle for f.*

Notably, unlike most quadratic speedups in quantum computing, the source of this quadratic speedup is not Grover's algorithm or amplitude amplification. Theorem 6.1.4 uses Belovs' quantum algorithm for learning symmetric juntas, which is constructed by exhibiting a feasible solution to the dual semidefinite program of the negative-weights adversary bound [Bel14].

Now that we have shown a quadratic quantum speedup on a family of instances known to be hard for classical algorithms, there is some hope that quantum algorithms may provide some speedup for the general first-order convex minimization problem. Alas, our next result (established in Section 6.4), which is our main result, shows that this is not the case, and quantum algorithms cannot in general yield a speedup over classical algorithms for first-order convex minimization.

Theorem 6.1.5 (Quantum lower bound)**.** *For any G, R, and ϵ, there exists a family of convex functions $f : \mathbb{R}^n \to \mathbb{R}$ with $n = \widetilde{O}((GR/\epsilon)^4)$, with Lipschitz constant at most G on $B(\vec{0}, R)$, such that any quantum algorithm that solves Problem 6.1.1 with high probability on this function family must make $\Omega((GR/\epsilon)^2)$ queries to f or $\mathcal{FO}(f)$ in the worst case.*

Our lower bound uses ideas from the lower bound against parallel randomized algorithms recently established by Bubeck, Jiang, Lee, Li, and Sidford [BJL$^+$19].

At a high level, the hard family of functions used in the randomized lower bound does not work for quantum algorithms because although classical algorithms can only learn $O(1)$ bits of information per query, quantum algorithms can make queries in superposition and learn a little information about many bits simultaneously. We remedy this by choosing a new family of functions in which with high probability, no matter what query the quantum algorithm makes, the oracle's response is essentially the same. This allows us to control what the quantum algorithm learns per query, but now the instance is more complicated and the quantum algorithm learns $O(n)$ bits of information per query. Since the final output of the algorithm is a vector in $\mathbb{R}^n$, we cannot use the argument used before that simply compared the information learned per query to the total information that needs to be learned. Instead we use the venerable hybrid argument [BBBV97] to control what the quantum algorithm learns and show that it cannot find an ϵ-approximate solution to the minimization problem.

6.1.3 Related work

Classically, there is a long history of the study of oracle complexity (also known as black-box complexity or query complexity) for deterministic and randomized algorithms for non-smooth and smooth convex optimization. The setting considered in this paper, first-order convex optimization, where the algorithm has query access to the function value and the gradient, is very well studied. This topic is too vast to survey here, but we refer the reader to [NY83, Nes04, Nes18, Bub15] for more information about upper and lower bounds that can be shown in this setting.

There also has been work in the classical parallel setting, where in each round the algorithm is allowed to query polynomially many points and the goal is to minimize the number of rounds [Nem94, BS18, DG19, BJL$^+$19]. Our work is most closely related to this setting and borrows many ideas from these works. Although quantum algorithms and parallel classical algorithms are incomparable in power, the constructions used to thwart parallel classical algorithms in these papers also help with showing quantum lower bounds.

In the quantum setting, there has been some work on convex optimization in the oracle model. There is also work on quantum gradient descent not in the oracle model. For example, one situation studied is where the dimension n of the optimization space is very large and the vectors are encoded in quantum states of dimension $\log n$. See [RSW$^+$19, KP20] and the references therein for more information. Another setting is the work on semidefinite programming, an important special case of convex optimization, but these algorithms exploit the specific structure of semidefinite programs [BS17, vAGGdW17, BKL$^+$19, vAG19] and are not directly related to our work.

While in the classical setting, in general, a function value oracle is weaker than a gradient oracle, this is not the case in the quantum setting. Given a function value oracle, one can get a gradient oracle quite efficiently (with an $\widetilde{O}(1)$ overhead) [Jor05, GAW19, vAGGdW20,

CCLW20]. A similar result also holds for simulating a separation oracle given a membership oracle for convex bodies [vAGGdW20, CCLW20]. As discussed before, our focus in this paper is to see if quantum algorithms can outperform classical algorithms when given a function oracle and gradient oracle since in many relevant settings, gradient computation is cheap in the classical case as well.

The most related works are the papers by Chakrabarti, Childs, Li, and Wu [CCLW20] and van Apeldoorn, Gilyén, Gribling, and de Wolf [vAGGdW20]. These papers establish very similar results so we cover them together. These papers study the problem of black-box convex optimization, and their results are phrased in the slightly different language of membership and separation oracles, but this is not the main difference between their work and our work. Indeed, it is possible to recast our problem in their setting (see the discussion in the introduction in [vAGGdW20] for how to do this). The main difference is that their algorithms have complexities that depend on n, whereas we're working in the parameter regime where n is large and so we seek algorithms that are independent of n.

Specifically, [CCLW20] and [vAGGdW20] consider the problem of minimizing a linear function over a convex body given via a membership or separation oracle. A membership oracle for a convex body tells us whether a given point x is in the convex body and a separation oracle in addition when x is not in the body outputs a hyperplane that separates x from the convex body. Classically, the problem of outputting an ϵ-approximate solution can be solved with $O(n^2 \operatorname{polylog}(\cdot))$ queries to a membership oracle, where we are suppressing polylogarithmic dependence on several parameters (including ϵ). These two papers show a quantum algorithm that makes only $O(n \operatorname{polylog}(\cdot))$ membership queries. The key technical component of this is a construction of a separation oracle from a membership oracle with only polylogarithmic overhead. To do this, they first show how to obtain an approximate subgradient oracle from a function oracle with only polylogarithmic overhead.

There are also several lower bounds shown in these papers. In [vAGGdW20], the authors prove that quantum algorithms do not give any advantage over classical algorithms in the setting where we are not given a point inside the convex body to start with. This setting is not directly comparable to our setting, as far as we are aware. In the setting where we do know a point inside the convex body, which is very similar to our setting, [vAGGdW20, CCLW20] prove a lower bound of $\Omega(\sqrt{n})$, which is quadratically worse than their algorithm. While, in general, their results are incomparable to our results, one specific comparison to our results is that [CCLW20, Theorem 3.3] essentially shows a $\widetilde{\Omega}(\min\{GR/\epsilon, \sqrt{n}\})$ lower bound on the number of oracle calls to a function value oracle for the setting in Problem 6.1.1.[5] Note that this is quadratically worse than our tight lower bound (Theorem 6.1.5) in the dimension-independent setting (i.e., when the dimension n is large compared to GR/ϵ).

[5]This is equivalent to our setting, where we have a function value and gradient oracle, due to their results.

6.2 First-Order Convex Optimization Preliminaries

Recall that the subgradient of a function $f : \mathbb{R}^n \to \mathbb{R}$ at a point $x \in \mathbb{R}^n$ is any vector $g_x \in \mathbb{R}^n$ such that for all $y \in \mathbb{R}^n$, $\langle g_x, y - x\rangle \leq f(y) - f(x)$. If f is differentiable at x, then the subgradient is unique and is equal to the gradient of f at x, defined as

$$\nabla f(x) := \left(\frac{\partial f(x)}{\partial x_1}, \ldots, \frac{\partial f(x)}{\partial x_n}\right). \tag{6.3}$$

The Lipschitz constant of the function f is defined as the minimum G such that $f(x) - f(y) \leq G\|x - y\|$ for all x, y. It is easy to see that the maximum norm of a subgradient of f, the maximum being over all x and over all its subgradients, is equal to the Lipschitz constant. Here we state some other useful properties about subgradients that are easy to verify. (See [Nes04, Section 3.1.5].)

- The subgradients of a function f at a point x form a convex set.
- For a linear function $f(x) = \langle g, x\rangle + c$, the subgradient at any point is unique and is equal to g.
- For the function $f(x) = \|x\|$, the subgradient at a point $x \neq \vec{0}$ is unique and is equal to $x/\|x\|$. The subgradients of f at $\vec{0}$ are the vectors of norm at most 1.
- For a function $f(x) = \max_i\{f_i(x)\}$ with each f_i being convex, and a point x and index j such that $f(x) = f_j(x)$, the subgradients of f_j at x are valid subgradients of f at x.
- For functions f and f' with subgradients g and g' at x, the vector $a \cdot g + b \cdot g'$ is a valid subgradient of $a \cdot f + b \cdot f'$ at x.

Before we move on to the lower bounds, let us see the *upper* bound, that the deterministic query complexity of Problem 6.1.1 is $O((GR/\epsilon)^2)$, matching the (randomized) lower bound of Theorem 6.1.3. In particular, we describe how the well-known gradient descent algorithm, or more precisely a variant known as the projected subgradient descent algorithm, achieves this upper bound. We now restate Theorem 6.1.2 for convenience:

Theorem 6.1.2 (Complexity of projected subgradient descent)**.** *The projected subgradient descent algorithm solves Problem 6.1.1 using $O((GR/\epsilon)^2)$ queries to f and $\mathcal{FO}(f)$.*

Proof. Without loss of generality we assume $G = R = 1$. The projected subgradient descent algorithm is easy to describe. We start by setting the initial vector $x_0 = \vec{0}$. The algorithm then computes x_{t+1} from x_t using the formula

$$x_{t+1} = \mathcal{P}_{\mathcal{K}}(x_t - \eta \cdot g_{x_t}), \tag{6.4}$$

where $\eta > 0$ is the step size, a parameter of the algorithm that we must choose, and $\mathcal{P}_{\mathcal{K}}$ is the projector onto $B(\vec{0}, 1)$. After T steps, the algorithm outputs $\hat{x}_T := \frac{1}{T}\sum_{t=0}^{T-1} x_t$. To obtain the claimed upper bound we set the step size $\eta = \epsilon$.

Now we claim that for any $T \geq 1/\epsilon^2$, the output $\hat{x}_T$ satisfies:

$$f(\hat{x}_T) - f(x^*) \leq \epsilon. \tag{6.5}$$

We prove this using the potential function $\|x_t - x^*\|^2$. We have

$$\|x_{t+1} - x^*\|^2 = \|\mathcal{P}_{\mathcal{K}}(x_t - \eta g_{x_t}) - x^*\|^2 \leq \|x_t - \eta g_{x_t} - x^*\|^2 \tag{6.6}$$

$$= \|x_t - x^*\|^2 - 2\eta \langle g_{x_t}, x_t - x^* \rangle + \eta^2 \|g_{x_t}\|^2, \tag{6.7}$$

where the inequality uses the fact that projecting a vector outside $\mathcal{K}$ to $\mathcal{K}$ can only reduce its distance to a point in $\mathcal{K}$. We then use the Lipschitz condition ($\|g_{x_t}\|^2 \leq 1$) and the definition of the subgradient in eq. (6.1) to get

$$\|x_{t+1} - x^*\|^2 \leq \|x_t - x^*\|^2 - 2\eta\,(f(x_t) - f(x^*)) + \eta^2, \text{ or} \tag{6.8}$$

$$f(x_t) - f(x^*) \leq \frac{\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2}{2\eta} + \frac{\eta}{2}. \tag{6.9}$$

Summing up the above inequality for each t from 0 to $T-1$, we note that the RHS telescopes to give us

$$\left(\frac{1}{T}\sum_{t=0}^{T-1} f(x_t)\right) - f(x^*) \leq \frac{\|x_0 - x^*\|^2 - \|x_T - x^*\|^2}{2\eta T} + \frac{\eta}{2} \leq \frac{1}{2\eta T} + \frac{\eta}{2} \leq \epsilon, \tag{6.10}$$

where the second inequality used the fact that $\|x_0 - x^*\| = \|x^*\| \leq R = 1$. By convexity of f, $f(\hat{x}_T) \leq \frac{1}{T}\sum_{n=0}^{T-1} f(x_t)$, which proves the result. □

Note that although we stated and proved this for $\mathcal{K} = B(\vec{0}, R)$, the upper bound on the number of queries made to the oracles holds for any $\mathcal{K}$ that is contained in $B(\vec{0}, R)$. However, if we wanted to implement this algorithm, then the time complexity would depend on how hard it is to implement the operator $\mathcal{P}_{\mathcal{K}}$, which projects onto the set $\mathcal{K}$.

6.3 Randomized Lower Bound

In this section, we prove a lower bound for randomized first-order methods for non-smooth convex optimization, restated here for convenience:

Theorem 6.1.3 (Randomized lower bound)**.** *For any G, R, and ϵ, there exists a family of convex functions $f : \mathbb{R}^n \to \mathbb{R}$ with $n = O((GR/\epsilon)^2)$, with Lipschitz constant at most G on $B(\vec{0}, R)$, such that any classical (deterministic or bounded-error randomized) algorithm that*

solves Problem 6.1.1 on this function family must make $\Omega((GR/\epsilon)^2)$ queries to f or $\mathcal{FO}(f)$ in the worst case.

This lower bound is known and multiple proofs can be found in the literature [NY83, WS17]. Our proof is elementary and we did not find it written anywhere, although it is conceptually similar to the one in [NY83], and so we include it here for completeness. Our proof also has the dimension $n = \Theta(1/\epsilon^2)$, without any log factors, which is the best possible. As far as we are aware, the previous proofs required larger dimension. As we will see later, the family of instances used is also interesting because we can get a quantum speedup for it, because of which we have to look at other instances to prove the quantum lower bound.

We can now define the family of convex functions used in the lower bound. For any $\epsilon > 0$, we set $n = \lfloor .9/\epsilon^2 \rfloor$ and look at the following class of functions.

Definition 6.3.1. *Let $z \in \{-1, 1\}^n$. Let $f_z : \mathbb{R}^n \to \mathbb{R}$ be defined as*

$$f_z(x_1, \ldots, x_n) = \max_{i \in [n]} z_i x_i. \tag{6.11}$$

Each such function is convex since it is a maximum of convex functions [Nes04, Theorem 3.1.5]. Note that if $f_z(x) = z_i x_i$ for some $i \in [n]$, then $z_i e_i$ is a subgradient of f_z at x (since $f_z(x) + \langle z_i e_i, y - x \rangle = z_i y_i \leq f_z(y)$). Hence the function is 1-Lipschitz. We can also see that within the unit ball the function is minimized at the point

$$x^* = \frac{-1}{\sqrt{n}} \sum_{i \in [n]} z_i e_i, \tag{6.12}$$

and $f_z(x^*) = -1/\sqrt{n}$. Clearly given x^* we can recover z from it. We now show z can even be recovered from an ϵ-approximate minimum of f_z.

Lemma 6.3.2. *Let x be such that $f_z(x) - f_z(x^*) \leq \epsilon$. Then we can recover $z \in \{0, 1\}^n$ from $x \in \mathbb{R}^n$.*

Proof. Let $s_x \in \{-1, +1\}^n$ be the vector with $(s_x)_i = \text{sign}(x_i)$, where $\text{sign}(a) = +1$ if $a \geq 0$ and $\text{sign}(a) = -1$ otherwise. We claim that $z = -s_x$. Toward a contradiction, if $(s_x)_i \neq -z_i$ for some i, then $(s_x)_i = z_i$, since these only take values in $\{-1, +1\}$. In this case, x_i and z_i agree in sign, and hence $f_z(x) \geq z_i x_i \geq 0$. Since $\epsilon < 1/\sqrt{n}$ (because of our choice of n above) the point x cannot satisfy $f_z(x) - f_z(x^*) \leq \epsilon$. □

Since this function is not differentiable everywhere, for our lower bound we need to specify the behavior of the subgradient oracle on all inputs. The function is not differentiable only at $x \in \mathbb{R}^n$ where the maximum is achieved at multiple indices. In this case, the subgradient oracle responds as if the maximum was achieved on the smallest such index i, i.e., it responds with $z_i e_i$. Note that for this function, querying the subgradient oracle allows us to simulate a call to the function oracle as well, since the response is $z_i e_i$ for the index i that achieves

the maximum, so the function evaluates to $z_i x_i$ at that point, which we can compute since we know x. So we can assume without loss of generality that an algorithm only queries the subgradient oracle.

Now that the problem is fully specified, we will show that any randomized optimization algorithm using the function oracle and this subgradient oracle will require $\Omega(n)$ queries in order to solve Problem 6.1.1 with a constant probability of success.

The following will be the crux of the lower bound. Let $I \subseteq [n]$. We say a distribution $\mathcal{D}$ over $\{-1, 1\}^n$ is I-fixed if for $z \sim \mathcal{D}$ the random variable z_I is fixed and $z_{\overline{I}}$ is uniform over $\{-1, 1\}^{\overline{I}}$.

Lemma 6.3.3. *Let z be distributed according to an I-fixed distribution. Let x be an arbitrary query made to the f_z oracle. After one query to the subgradient oracles, the conditional distribution on z given the answer is I'-fixed with $I \subseteq I'$ and $\mathbb{E}[|I'|] \leq |I| + 2$.*

Proof. Let x be the algorithm's query. The index i that achieves the maximum in the definition of $f_z(x)$ can be computed as follows. Let $i_1, \ldots, i_n$ be the ordering of the indices 1 to n in decreasing order of $|x_i|$, with ties broken with the natural ordering on integers. The oracle outputs $f_z(x) = z_{i_j} x_{i_j}$ and chooses the subgradient $z_{i_j} e_{i_j}$ where j is the smallest index for which x_{i_j} agrees in sign with z_{i_j}, and if no such index exists, then $j = n$.

Since $f_z(x)$ can be computed given the subgradient $z_{i_j} e_{i_j}$, the only information obtained from a query is the prefix $\{z_{i_k}\}_{k \leq j}$. In other words, if the subgradient oracle responds with $z_{i_j} e_{i_j}$, then we have learned that for all indices $k \leq j$, we must have $\text{sign}(x_i) = -z_i$, but we have not learned any more since the oracle's output does not depend on the bits of z with index i_k with $k > j$. After this query, we know the bits z_{i_k} with $k \leq j$, but conditioned on these, the distribution on the remaining bits of z continues to be uniform. This is an I'-fixed distribution with $I' = I \cup \{i_k\}_{k \leq j}$. Intuitively, I' cannot be much larger than I since an index i_k is part of this set only if the algorithm correctly guessed the sign of z_{i_k} for this index and all indices with a smaller value of k. Since the initial distribution z was uniformly at random outside of I and x is fixed, the probability of correctly guessing the first index (according to the i_j ordering) that was not fixed is $1/2$, the probability of guessing the first two is $1/4$ and so on. Thus the expected number of new entries fixed by one query is $\sum_{k=1}^{n \setminus |I|} k \cdot \frac{1}{2^k} \leq 2$. □

We can use this to establish the final claim.

Lemma 6.3.4. *Let z be sampled uniformly at random from $\{-1, 1\}^n$. If a randomized algorithm $\mathcal{A}$ outputs an x with $f_z(x) - f_z(x^*) \leq \epsilon$ with probability at least $2/3$, then its query complexity is at least $n/3 - 1$.*

Proof. When $\mathcal{A}$ outputs a point x, we will require it to also query the oracle at x to see if it is indeed ϵ-optimal. This can increase its query complexity by at most one. Let the query complexity of this modified $\mathcal{A}$ be t. Whenever $\mathcal{A}$ does output an ϵ-optimal point, Lemma 6.3.2 implies that the conditional distribution on z is $[n]$-fixed. For each $i \in [0, \ldots, t]$, let I_i be the

random variable such that the distribution on z after i queries of $\mathcal{A}$ is I_i-fixed (Lemma 6.3.3 implies that after any sequence of queries it will be an I-fixed distribution for some I). Since z is sampled uniformly at random from $\{-1,1\}^n$, $I_0 = \emptyset$. And since we want the algorithm to succeed with probability at least $2/3$, $\mathbb{E}[|I_t|] \geq 2n/3$.

However, $|I_t| = \sum_{i=1}^{t} |I_i| - |I_{i-1}|$, and it is a simple consequence of Lemma 6.3.3 that $\mathbb{E}[|I_i| - |I_{i-1}|] \leq 2$ for all i. So by the linearity of expectation, $\mathbb{E}[|I_t|] \leq 2t$ and hence $t \geq n/3$. □

This proves a lower bound of $\Omega(1/\epsilon^2)$ on the randomized query complexity of first-order convex minimization for a function with $G = R = 1$. As noted earlier, this is without loss of generality and implies the more general bound in Theorem 6.1.3.

6.3.1 Quantum speedup

In this section we prove Theorem 6.1.4, restated for convenience:

Theorem 6.1.4 (Quantum algorithm for classically hard function family)**.** *There is a quantum algorithm that solves Problem 6.1.1 on the class of functions that appear in the classical lower bound of Theorem 6.1.3 using $O(GR/\epsilon)$ queries to the oracle for f.*

The quantum speedup for the above class of functions relies on Belovs' quantum algorithm for Combinatorial Group Testing [Bel14]. Belovs showed that given access to an oracle making OR queries to an n-bit string, the n-bit string can be learned in $O(\sqrt{n})$ quantum queries. More formally, Belovs showed the following [Bel14].

Theorem 6.3.5. *Let $x \in \{0,1\}^n$ and O_x be the unitary that for every $S \subseteq [n]$ and $b \in \{0,1\}$, satisfies $O_x|S\rangle|b\rangle = |S\rangle|b \oplus \mathsf{OR}_x(S)\rangle$, where $\mathsf{OR}_x(S) = 1$ if there is an $i \in S$ such that $x_i = 1$, and $\mathsf{OR}_x(S) = 0$ otherwise. Then we can learn x with high probability with $O(\sqrt{n})$ quantum queries to the oracle O_x.*

We can now prove Theorem 6.1.4.

Proof of Theorem 6.1.4. In our optimization problem, making the query $x = \frac{1}{\sqrt{n}} \sum_{i \in S} e_i$ to the function oracle returns $f_z(x) = \frac{1}{\sqrt{n}}$ if there is an $i \in S$ such that $z_i = 1$. If there is no such $i \in S$, then it will output $f_z(x) = 0$, unless $S = n$, in which case it will output $-\frac{1}{\sqrt{n}}$.

Hence a function value oracle for f_z can be used to make OR queries to the string z, since it outputs 1 if there is an $i \in S$ such that $z_i = 1$ and outputs 0 (or $-1/\sqrt{n}$) otherwise. Using Belovs' algorithm, with $O(\sqrt{n})$ such queries, we can learn the locations of all the 1s in z, which allows us to learn z completely. □

This quantum algorithm is also essentially optimal for this problem and it is not hard to show an $\Omega(\sqrt{n}/\log n)$ lower bound for quantum algorithms. A similar lower bound is shown in [CCLW20, Theorem 3.3], and we sketch a simpler proof of the claim here.

As discussed in the classical lower bound, what the subgradient oracle allows us to do is have a non-standard query to the unknown string $z \in \{-1, 1\}^n$. In this non-standard query, we get to order the bits of z however we like, and then submit a string in $\{-1, 1\}^n$ and ask for the first index (according to our ordering) where our string agrees with z. As we showed in the classical lower bound, if we solve the optimization problem, then we also learn z.

So we are left with answering the question of how hard it is to learn z given these non-standard queries to z. Given standard queries to z, where we can only query one bit of our choice, it is well known that we need $\Omega(n)$ queries to learn z. But our non-standard query is easy to implement using Grover's algorithm with only $O(\sqrt{n})$ standard queries, since all we have to do is find the first bit of z according to a known ordering where the queried string and z agree. If the problem of learning z with these non-standard queries used T non-standard queries, then we could implement the non-standard queries ourselves with cost $O(\sqrt{n})$ and compose the two algorithms to obtain an algorithm for learning z using standard queries with complexity $O(T\sqrt{n})$. (Thanks to the properties of the composition of bounded-error quantum query complexity [Rei11].) Since this problem has a lower bound of $\Omega(n)$, we get $T = \Omega(\sqrt{n}/\log n)$.

6.4 Quantum lower bound

In this section, we show that for any ϵ, there exists a 1-Lipschitz family of functions such that any quantum algorithm that solves Problem 6.1.1 on the unit ball must make $\frac{1}{100\epsilon^2}$ queries. In other words, there is no quantum first-order convex optimization algorithm that always outperforms the classical gradient descent algorithm described in Theorem 6.1.2. The function we will use was introduced by Nemirovsky and Yudin [NY83]. To show the quantum lower bound, we adapt to the quantum setting the lower bound strategy of Bubeck et al. [BJL$^+$19] in the model of parallel algorithms.

We restate the main result proved in this section for convenience:

Theorem 6.1.5 (Quantum lower bound)**.** *For any G, R, and ϵ, there exists a family of convex functions $f : \mathbb{R}^n \to \mathbb{R}$ with $n = \widetilde{O}((GR/\epsilon)^4)$, with Lipschitz constant at most G on $B(\vec{0}, R)$, such that any quantum algorithm that solves Problem 6.1.1 with high probability on this function family must make $\Omega((GR/\epsilon)^2)$ queries to f or $\mathcal{FO}(f)$ in the worst case.*

We start by first proving a qualitatively similar, but simpler result with a larger value of $n = \widetilde{O}((GR/\epsilon)^6)$ in Section 6.4.4. If we only care about the optimality of gradient descent in the dimension-independent setting, this lower bound is sufficient. But if we also want to understand the trade-off between dimension-independent and dimension-dependent algorithms, then we would like to show this lower bound with as small a value of n as we can. In Section 6.4.5, we improve the lower bound to achieve the value of n stated in this theorem.

6.4.1 Function family and basic properties

We start by defining the family of functions $\mathcal{F} = \{f : \mathbb{R}^n \to \mathbb{R}\}$ that we use. The function family $\mathcal{F}$ depends on the dimension n and two other parameters k and γ. Since the function family we choose depends on ϵ, the parameters n, k, and γ will be functions of ϵ. Our choice of n, k, and γ will become clear later, but for now we simply choose them as follows. Let

$$k := \frac{1}{100\epsilon^2} \implies \epsilon = \frac{1}{10\sqrt{k}} \quad \text{and} \quad \gamma := \frac{1}{10k^{3/2}} = 100\epsilon^3. \tag{6.13}$$

We choose n such that it satisfies

$$\gamma \geq 8\sqrt{\frac{\log n}{n}} \implies n := O\left(\frac{\log(1/\epsilon)}{\epsilon^6}\right) = \widetilde{O}\left(\frac{1}{\epsilon^6}\right). \tag{6.14}$$

The discussion before Lemma 6.4.2 explains the choice of k and the discussion after Lemma 6.4.3 explains the choice of γ. For the dimension n, see the discussion at the beginning of Section 6.4.2.

We now define the function family for these specific choices of n, k, and γ.

Definition 6.4.1 (Hard function family)**.** *Let $\mathcal{V} = \{(v_1, \ldots, v_k) \mid \forall i, j, \in [k], \langle v_i, v_j \rangle = \delta_{ij}\}$ be the set of all k-tuples of orthonormal vectors in $\mathbb{R}^n$. Let the family of functions $\mathcal{F} = \{f_V\}_{V \in \mathcal{V}}$ be defined as*

$$f_{(v_1, v_2, \ldots, v_k)}(x) := \max_{i \in [k]} \{g_V^{(i)}(x)\}, \text{ where } g_V^{(i)}(x) := \langle v_i, x \rangle + (k - i)\gamma \|x\|. \tag{6.15}$$

We will show that any quantum algorithm that solves Problem 6.1.1 on the functions in this family must make k queries. As we will prove, informally what happens is each query of the quantum algorithm to the gradient oracle only reveals a single direction v_i to the algorithm. In fact, with very high probability the vectors are revealed in order, so that the algorithm first learns v_1, then v_2, and so on. As we will show in Lemma 6.4.3, any ϵ-optimal solution must overlap significantly with all v_i, and thus any quantum algorithm must make k queries. Since we want to show an $\Omega(1/\epsilon^2)$ bound, we choose k to be a small multiple of $1/\epsilon^2$, which explains our choice for k in eq. (6.13).

We now establish some basic properties of these functions.

Lemma 6.4.2 (Properties of f_V)**.** *For any $V \in \mathcal{V}$, let f_V and $g_V^{(i)}$ be as in Definition 6.4.1. Then f_V is convex with Lipschitz constant at most $1 + k\gamma \leq 2$ on $B(\vec{0}, 1)$, and*

$$\text{for } x \neq \vec{0}, \quad \nabla g_V^{(i)}(x) = v_i + (k - i)\gamma x / \|x\|, \text{ and} \tag{6.16}$$

$$\text{for } x = \vec{0}, \quad \partial g_V^{(i)}(\vec{0}) = \{v_i + (k - i)\gamma u \mid u \in B(\vec{0}, 1)\}, \text{ and} \tag{6.17}$$

$$\text{for any } x, \quad \partial f_V(x) = \text{ConvexHull}\big(\{u \in \partial g_V^{(i)}(x) \mid g_V^{(i)}(x) = f_V(x)\}\big), \tag{6.18}$$

where the convex hull of a set of vectors is the set of all convex combinations of vectors in the

set. Lastly, for any $\alpha > 0$, $f_V(\alpha x) = \alpha f(x)$ and $\partial f_V(\alpha x) = \partial f_V(x)$.

Proof. For all $V \in \mathcal{V}$, $f_V : \mathbb{R}^n \to \mathbb{R}$ is convex. This follows because linear functions and norms are convex functions [Nes04, Example 3.1.1], and the sum or maximum of convex functions is convex [Nes04, Theorem 3.1.5].

Let us now compute the subgradients of $g_V^{(i)}(x) = \langle v_i, x\rangle + (k-i)\gamma\|x\|$. The linear function $\langle v_i, x\rangle$ is differentiable and its gradient is simply v_i. The Euclidian norm $\|x\|$ is differentiable everywhere except at $x = \vec{0}$. At $x \neq \vec{0}$, the gradient of $\|x\|$ is $x/\|x\|$ and at $x = 0$, the set of subgradients is $B(\vec{0}, 1)$ [Nes04, Example 3.1.5]. We also know that $\partial(\alpha_1 f_1(x) + \alpha_2 f_2(x)) = \alpha_1 \partial f_1(x) + \alpha_2 \partial f_2(x)$ [Nes04, Lemma 3.1.9], which gives us the expressions for the subgradients of $g_V^{(i)}$.

For a function that is the maximum of functions $g_V^{(i)}$, we know that the set of subgradients is simply the convex hull of subgradients of those $g_V^{(i)}$ which achieve the maximum at the given point x [Nes04, Lemma 3.1.10].

The Lipschitz constant of a function is the maximum norm of any subgradient of the function. Since any vector in $\partial g_V^{(i)}$ has norm $1 + k\gamma$, and any vector in ∂f_V is the convex combination of vectors with norm at most $1 + k\gamma$, the Lipschitz constant of f_V is at most $1 + k\gamma \leq 2$.

Finally, it is easy to see from the definition of f_V that for $\alpha > 0$, $f_V(\alpha x) = \alpha f(x)$ since each term in the max gets multiplied by α. For $\partial f_V(\alpha x)$, note that this is a convex combination of $\partial g_V^{(i)}(\alpha x)$, and these do not depend on α. □

For convenience we work with this family of functions with Lipschitz constant at most 2 instead of 1, which doesn't change the asymptotic bounds since we could just divide every function f_V by 2.

The last property essentially says that querying the function or its subgradient on a scalar multiple of a vector x gives us only as much information as querying it on x. Thus we can assume that an algorithm only queries the oracles within the unit ball without loss of generality.

Now let us discuss the vector $x^* \in B(\vec{0}, 1)$ that minimizes $f_V(x)$ and vectors that ϵ-approximately solve the minimization problem. First note that if γ were equal to 0, then the function would simply be $\max_{i\in[k]}\langle v_i, x\rangle$, which requires us to minimize the component of x in k different directions subject to it being a unit vector. The solution to this is simply $\frac{-1}{\sqrt{k}}\sum_i v_i$. Now $-1/\sqrt{k} = -10\epsilon$, so the overlap of x with each direction v_i is a large multiple of ϵ. So even an ϵ-approximate solution must have reasonable overlap with each of the vectors v_i. Specifically, each overlap must be at least -9ϵ. Now in our function f_V the term γ is not 0, but that term at most perturbs the function by $k\gamma = \epsilon$, which again is much smaller than 10ϵ, and thus even approximate solutions must have significant overlaps with all v_i. We formalize these properties below.

Lemma 6.4.3 (Properties of the minimum)**.** *For any $V \in \mathcal{V}$, let $f_V : \mathbb{R}^n \to \mathbb{R}$ be the function*

in Definition 6.4.1 and let $x^* := \arg\min_{x \in B(\vec{0},1)} f_V(x)$. *Then* $f_V(x^*) \leq -9\epsilon$. *Furthermore, any* $x \in \mathbb{R}^n$ *that satisfies* $|f_V(x) - f_V(x^*)| \leq \epsilon$ *must satisfy for all* $i \in [k]$, $\langle v_i, x \rangle \leq -8\epsilon$.

Proof. Consider the vector $y = \frac{-1}{\sqrt{k}} \sum_{i \in [k]} v_i$. This is a vector in $B(\vec{0}, 1)$, satisfying $f_V(y) \leq \frac{-1}{\sqrt{k}} + (k-1)\gamma \leq \frac{-1}{\sqrt{k}} + k\gamma = -10\epsilon + \epsilon = -9\epsilon$, because we have $10\epsilon = \frac{1}{\sqrt{k}}$ and $k\gamma = \frac{1}{10\sqrt{k}} = \epsilon$. Thus $f_V(x^*) \leq f_V(y) \leq -9\epsilon$.

Now consider any vector x with $|f_V(x) - f_V(x^*)| \leq \epsilon$, which implies $f_V(x) \leq -8\epsilon$. If $\langle v_i, x \rangle > -8\epsilon$ for any $i \in [k]$, then $f_V(x) \geq \langle v_i, x \rangle + (k-i)\gamma\|x\| > -8\epsilon$, which is a contradiction. □

This result crucially uses the relation between γ and k and because we want $k\gamma$ to be a constant factor (say 10) smaller than $\sqrt{1/k}$, this informs our choice of γ in eq. (6.13). Our choice of n in eq. (6.14) will be discussed in the next section.

6.4.2 Probabilistic facts about the function family

So far all the properties we have discussed of our function family hold for any $V \in \mathcal{V}$, but now we want to talk about a hard distribution over such functions. Specifically we want to talk about choosing a uniformly random (according to the Haar measure) V from the infinite set $\mathcal{V}$. It is easy to see how to sample a random V once we can sample unit vectors from a subspace. We start by choosing v_1 to be a Haar random unit vector from $\mathbb{R}^n$, let v_2 be a Haar random unit vector from $\mathsf{span}(v_1)^\perp$, and so on, until v_k is a Haar random unit vector in $\mathsf{span}(v_1, v_2, \ldots, v_{k-1})^\perp$. In the following, to improve readability, we will use boldface to denote random variables.

We can now discuss what determines our choice of n. By construction, the family of functions $\mathcal{F}$ has the property that if the input vector x has equal inner product with all vectors v_i, then the maximum will be achieved uniquely on the first term $i = 1$ because the additive term $(k-i)\gamma\|x\|$ is largest for $i = 1$. Now what we want to ensure is that this property holds even when x does not have equal inner product with all v_i, but x is chosen uniformly at random from $B(\vec{0}, 1)$. Or equivalently, we want this property to hold when x is fixed, but the set V is chosen uniformly at random.

In either case, the inner product of x with a random unit vector v will be a random variable with mean 0 due to symmetry. But the expected value of $|\langle v, x \rangle|^2$ for a random unit vector v is $1/n$, and in fact it will be tightly concentrated around $1/n$. The following proposition follows from [Bal97, Lemma 2.2].

Proposition 6.4.4. *Let* $x \in B(\vec{0}, 1)$. *Then for a random unit vector* v, *and all* $c > 0$,

$$\Pr_v(|\langle x, v \rangle| \geq c) \leq 2e^{-nc^2/2}. \tag{6.19}$$

We choose γ so that it is very unlikely that the maximum is not achieved at $i = 1$ (probabililty polynomially small in n). From Proposition 6.4.4, we see that the probability of

any $|\langle v_i, x\rangle|^2$ being larger than a constant multiple of $\log n/n$ is inverse polynomially small. So it is sufficient to take γ^2 to be a large constant multiple of $\log n/n$ as in eq. (6.14).

In our lower bound we will need a slightly stronger result. We can show that if the vectors $v_1, \ldots, v_{t-1}$ are fixed (and hence known to the algorithm), and the remaining vectors $v_t, \ldots, v_k$ are chosen uniformly at random such that the set of vectors $\{v_1, \ldots, v_k\}$ is orthonormal, then the maximum will be achieved in the set $[t]$ with high probability. This generalizes the previous claim, which is the case of $t = 1$, where none of the vectors were fixed.

Lemma 6.4.5 (Most probable argmax)**.** *Let $1 \leq t \leq k$ be integers and $\{v_1, \ldots, v_{t-1}\}$ be a set of orthonormal vectors. Let $\{v_t, \ldots, v_k\}$ be chosen uniformly at random so that the set $\{v_1, \ldots, v_k\}$ is orthonormal. Then*

$$\forall x \in B(\vec{0}, 1) : \Pr_{v_t, \ldots, v_k} \left(\max_{i \in [k]} \langle v_i, x\rangle + (k-i)\gamma\|x\| \neq \max_{i \in [t]} \langle v_i, x\rangle + (k-i)\gamma\|x\| \right) \leq \frac{1}{n^7}. \quad (6.20)$$

Proof. Let E_x denote the event whose probability we want to upper bound. Since E_x and $E_{\alpha x}$, for any $\alpha \in [0, 1]$, are the same event, we can assume without loss of generality that $\|x\| = 1$. If event E_x occurs, then it must hold that

$$\max_{i \in \{t+1, \ldots, k\}} \langle v_i, x\rangle + (k-i)\gamma > \max_{i \in [t]} \langle v_i, x\rangle + (k-i)\gamma \geq \langle v_t, x\rangle + (k-t)\gamma. \quad (6.21)$$

We want to show that this event is very unlikely. To do so, let F_x be the event that for all $i \in \{t, \ldots, k\}$, $\langle v_i, x\rangle \in [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$. Note that if F_x occurs, then the terms in the max are in decreasing order, and we have

$$\langle v_t, x\rangle + (k-t)\gamma \geq \langle v_{t+1}, x\rangle + (k-t-1)\gamma \geq \cdots \geq \langle v_{k-1}, x\rangle + \gamma \geq \langle v_k, x\rangle, \quad (6.22)$$

which contradicts eq. (6.21). Thus if E_x holds then the complement of F_x, $\bar{F}_x$ must hold, which means $\Pr(E_x) \leq \Pr(\bar{F}_x)$. So let us show that F_x is very likely.

The event $\bar{F}_x$ holds only if there exists an $i \in \{t, \ldots, k\}$ such that $\langle v_i, x\rangle \notin [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$. We can upper bound this probability for any particular $i \in \{t, \ldots, k\}$ using Proposition 6.4.4 and the fact that v_i is chosen uniformly at random from an $n - t + 1$-dimension ball. This probability is at most $2e^{-(n-t+1)\gamma^2/8} = 2e^{-(n-t+1)\cdot 8\frac{\log n}{n}} \leq 2 \cdot 2^{-8\log n} = 2/n^8$, with the inequality holding because $n > 4t$. The probability that this happens for any i is at most $(k - t + 1) \leq k$ times this probability, by the union bound. Using the fact that $2k < n$, we get that $\Pr(E_x) \leq \Pr(\bar{F}_x) < 1/n^7$. □

Finally, we show that even if we knew the vectors $v_1, \ldots, v_{k-1}$, we cannot guess a vector x that is an ϵ-approximate solution to our problem, because it won't have enough overlap with v_k, which is unknown. In other words, for an algorithm to output an ϵ-optimal solution, it essentially must know the entire set V.

Lemma 6.4.6 (Cannot guess x^*)**.** *Let $k > 0$ be an integer and $\{v_1, \ldots, v_{k-1}\}$ be a set of orthonormal vectors. Let v_k be chosen uniformly at random from $\mathsf{span}(v_1, \ldots, v_{k-1})^\perp$ and let $V = (v_1, \ldots, v_k)$. Then*

$$\forall x \in B(\vec{0}, 1) : \Pr_{v_k}\left(f_V(x) - f_V(x^*) \leq \epsilon\right) \leq 2e^{-\Omega(k^2)}. \tag{6.23}$$

Proof. From Lemma 6.4.3, we know that an ϵ-optimal solution x must satisfy $\langle v_k, x\rangle \leq -8\epsilon$. But v_k is chosen uniformly at random from the space $\mathsf{span}(v_1, \ldots, v_{k-1})^\perp$ and any vector $x \in B(\vec{0}, 1)$ projected to that space also has length at most 1. So from Proposition 6.4.4 we know that for any $x \in B(\vec{0}, 1)$,

$$\Pr_{v_k}(\langle v_k, x\rangle \leq -8\epsilon) \leq \Pr_{v_k}(|\langle v_k, x\rangle| \geq 8\epsilon) \leq 2e^{-32(n-k+1)\epsilon^2} \leq 2e^{-\Omega(k^2)}. \tag{6.24}$$

□

6.4.3 Quantum query model

We now formally define the quantum query model in our setting. In the usual quantum query model the set of allowed queries is finite, whereas in our setting it is natural to allow the quantum algorithm to query the oracles at any point $x \in \mathbb{R}^n$. Due to Lemma 6.4.2, it is sufficient to allow the algorithm to query any $x \in B(\vec{0}, 1)$, but this is still a continuous space of queries, and hence a query vector could be a superposition over infinitely many states. Instead of formalizing this notion of quantum algorithms, we allow the algorithm to make discrete queries only, but to arbitrarily high precision. The reader is encouraged to not get bogged down by details and to think of the registers as storing the real values that they ideally should, but in the rest of this section we define these algorithms more carefully so that all the spaces involved are finite and well defined. This formalization is not specific to the quantum setting and is done classically as well if we do not want to manipulate real numbers as atomic objects.

All the real numbers that appear will be represented using some b bits of precision, where b can be chosen by the algorithm. The reader should imagine b being arbitrarily large, say exponentially larger than all the parameters involved in the problem, so that the inaccuracy involved by using this representation is negligible. Then the algorithm represents the input $x \in B(\vec{0}, 1)$ using b bits of precision per coordinate. The oracle's response will also use b bits of precision per real number. For a given choice of b, the quantum algorithm will have some probability of success of solving the problem at hand. We then define the success probability of quantum algorithms that make q queries by taking a supremum over all b of q-query algorithms that solve the problem.

We can now define the oracles more precisely. Classically, the function oracle for a function $f : \mathbb{R}^n \to \mathbb{R}$ would simply implement the map $x \mapsto f(x)$, where we represent each entry of x and the output $f(x)$ using b bits, so $x \in \{0,1\}^{bn}$ and $f(x) \in \{0,1\}^b$. Let's say we have a classical circuit that implements this map using G gates, say over the gate set of AND, OR,

and NOT gates. Then it is easy to construct, in a completely black-box way, a quantum circuit using $O(G)$ gates (say over the gate set of Hadamard, CNOT, and T) that performs the unitary $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$, for every $x \in \{0,1\}^{bn}$, and $y \in \{0,1\}^b$. This is why it is standard to assume that the quantum oracle corresponding to the classical map $x \mapsto f(x)$ is a unitary that performs $U|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle$. We apply the same construction for the $\mathcal{FO}(f)$ oracle to get the quantum analogue of the classical map $x \mapsto g_x$, where g_x is some subgradient of f at x. Lastly, for convenience we will combine both the function and subgradient oracle into one oracle that when queried with x returns $f(x)$ and a subgradient at x. Since our function family is parameterized by $V \in \mathcal{V}$, we call this oracle O_V.

Let $\mathcal{A}$ be a quantum query algorithm that makes q queries. $\mathcal{A}$ is described by a sequence of unitaries $U_q O_V U_{q-1} O_V U_{q-2} O_V \cdots U_1 O_V U_0$ applied to an initial state, say $|0\rangle$. We assume that the output of $\mathcal{A}$, which is a vector x, is determined by measuring the first n registers storing real numbers using b bits.

6.4.4 Lower bound

We can now prove the quantum lower bound. Let $\mathcal{A}$ be a $k-1$ query quantum algorithm that solves Problem 6.1.1 on all the functions f_V for $V \in \mathcal{V}$. Due to Lemma 6.4.2, we can assume that the algorithm only queries the oracles with vectors $x \in B(\vec{0}, 1)$. We also need to describe the behavior of the subgradient oracle on inputs where the subgradient is not unique. On such inputs x, the subgradient is not unique because several indices $i \in [k]$ simultaneously achieve the maximum in $f_V(x)$. In this case, the subgradient will answer as if the smallest index i in this set achieved the maximum. Now let $\mathcal{A}$ be described by the sequence of unitaries $U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V U_0$ acting on the starting state $|0\rangle$. Let this sequence of unitaries be called A. Then the final state of the algorithm is $A|0\rangle$.

Recall that we defined $f_V(x) = \max_{i \in [k]} \{g_V^{(i)}(x)\}$. Let us also define functions $f_V^{(j)}$ where the maximization is only over the first j indices instead of all k indices. Specifically, let $f_V^{(j)} := \max_{i \in [j]} \{g_V^{(i)}(x)\}$. We previously defined the oracle O_V as corresponding to the function f_V. Let $O_V^{(j)}$ be the oracle corresponding to the functions $f_V^{(j)}$.

Now we define a sequence of unitaries starting with $A_0 = A$ as follows:

$$\begin{aligned}
A_0 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V U_0 \\
A_1 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V U_1 O_V^{(1)} U_0 \\
A_2 &:= U_{k-1} O_V U_{k-2} O_V \cdots O_V^{(2)} U_1 O_V^{(1)} U_0 \\
&\vdots \\
A_{k-1} &:= U_{k-1} O_V^{(k-1)} U_{k-2} O_V^{(k-2)} \cdots O_V^{(2)} U_1 O_V^{(1)} U_0
\end{aligned} \tag{6.25}$$

We want to show that the algorithm A_0 does not solve our problem. To do so, we will employ the hybrid argument, in which we show that the output of the algorithm A_i and A_{i+1}

is close, and thus the output of A_0 and A_{k-1} is close. Finally, we argue that the algorithm A_{k-1} does not solve our problem because the oracles in the algorithm do not know v_k. Let us first establish these two claims.

Lemma 6.4.7 (A_{k-1} does not solve the problem)**.** *Let $\mathcal{A}$ be a $k-1$ query algorithm and let A_{k-1} be defined as above. Let p_V be the probability distribution over $x \in B(\vec{0}, 1)$ obtained by measuring the output state $A_{k-1}|0\rangle$. Then* $\Pr_{V,x\sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$.

Proof. We want to show that the probability (over the random choice of V and the internal randomness of the algorithm) that A_{k-1} outputs an x that satisfies $f(x) - f(x^*) \leq \epsilon$ is very small.

Let us establish the claim for any fixed choice of $v_1, \ldots v_{k-1}$, since if the claim holds for any fixed choice of these vectors, then it also holds for any probability distribution over them. For a fixed choice of vectors, this claim is just $\Pr_{v_k,x\sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$. Now since the algorithm A_{k-1} only has oracles $O_V^{(i)}$ for $i < k$, the probability distribution p_V only depends on $v_1, \ldots, v_{k-1}$. Since these are fixed, this is just a fixed distribution p. So we can instead establish our claim for all $x \in B(\vec{0}, 1)$, which will also establish it for any distribution.

So what we need to establish is that for any $x \in B(\vec{0}, 1)$, $\Pr_{v_k}(f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k^2)}$, which is exactly what we showed in Lemma 6.4.6. □

Lemma 6.4.8 (A_t and A_{t-1} have similar outputs)**.** *Let $\mathcal{A}$ be a $k-1$ query algorithm and let A_t for $t \in [k-1]$ be the unitaries defined in eq.* (6.25)*. Then*

$$\mathop{\mathbb{E}}_{V}(\|A_t|0\rangle - A_{t-1}|0\rangle\|^2) \leq \frac{4}{n^7}. \tag{6.26}$$

Proof. From the definition of the unitaries in eq. (6.25) and the unitary invariance of the spectral norm, we see that $\|A_t|0\rangle - A_{t-1}|0\rangle\| = \|(O_V^{(t)} - O_V)U_{t-1}O_V^{(t-1)} \cdots O_V^{(1)}U_0|0\rangle\|$. Let us again prove the claim for any fixed choice of vectors $v_1, \ldots, v_{t-1}$, which will imply the claim for any distribution over those vectors. Once we have fixed these vectors, the state $U_{t-1}O_V^{(t-1)} \cdots O_V^{(1)}U_0|0\rangle$ is a fixed state, which we can call $|\psi\rangle$. Thus our problem reduces to showing for all quantum states $|\psi\rangle$,

$$\mathop{\mathbb{E}}_{v_t,\ldots,v_k}(\|(O_V^{(t)} - O_V)|\psi\rangle\|^2) \leq \frac{4}{n^7}. \tag{6.27}$$

Now we can write an arbitrary quantum state as $|\psi\rangle = \sum_x \alpha_x|x\rangle|\phi_x\rangle$, where x is the query made to the oracle, and $\sum_x |\alpha_x|^2 = 1$. Thus the LHS of eq. (6.27) is equal to

$$\mathop{\mathbb{E}}_{v_t,\ldots,v_k}\left(\left\|\sum_x \alpha_x(O_V^{(t)} - O_V)|x\rangle|\phi_x\rangle\right\|^2\right) \leq \sum_x |\alpha_x|^2 \mathop{\mathbb{E}}_{v_t,\ldots,v_k}\left(\|(O_V^{(t)} - O_V)|x\rangle\|^2\right). \tag{6.28}$$

Since $|\alpha_x|^2$ defines a probability distribution over x, we can again upper bound the right hand side for any x instead. Since $O_V^{(t)}$ and O_V behave identically for some inputs x, the

only nonzero terms are those where the oracles respond differently, which can only happen if $f_V^{(t)}(x) \neq f_V(x)$. When the response is different, we can upper bound $\|(O_V^{(t)} - O_V)|x\rangle\|^2$ by 4 using the triangle inequality. Thus for any $x \in B(\vec{0}, 1)$, we have

$$\mathop{\mathbb{E}}_{v_t,\ldots,v_k}\left(\|(O_V^{(t)} - O_V)|x\rangle\|^2\right) \leq 4 \mathop{\Pr}_{v_t,\ldots,v_k}(f_V^{(t)}(x) \neq f_V(x)) \leq 4/n^7, \tag{6.29}$$

where the last inequality follows from Lemma 6.4.5. □

Finally we can put these two lemmas together to prove our lower bound.

Lemma 6.4.9 ($\mathcal{A}$ does not solve the problem)**.** *Let $\mathcal{A}$ be a $k-1$ query algorithm. Let p_V be the probability distribution over $x \in B(\vec{0}, 1)$ obtained by measuring the output state $A|0\rangle$. Then* $\Pr_{V,x\sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) \leq \frac{1}{\text{poly}(n)}$.

Proof. Let P_V be the projection operator that projects a quantum state $|\psi\rangle$ onto the space spanned by vectors $|x\rangle$ for x such that $f_V(x) - f_V(x^*) \leq \epsilon$. Then $\|P_V A|0\rangle\|^2 = \Pr_{x\sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon)$. We know from Lemma 6.4.7 that $\mathbb{E}_V(\|P_V A_{k-1}|0\rangle\|^2) \leq 2e^{-\Omega(k^2)}$. We prove our upper bound on the probability by showing that it is approximately the same as $\mathbb{E}_V(\|P_V A_{k-1}|0\rangle\|^2)$.

Lemma 6.4.8 states that for all $1 \leq t < k$, $\mathbb{E}_V(\|A_t|0\rangle - A_{t-1}|0\rangle\|^2) \leq \frac{4}{n^7}$. Using telescoping sums and the Cauchy-Schwarz inequality, we see that

$$\mathop{\mathbb{E}}_V(\|A_{k-1}|0\rangle - A|0\rangle\|^2) \leq \mathop{\mathbb{E}}_V\left(\left(\sum_{t\in[k-1]} \|A_t|0\rangle - A_{t-1}|0\rangle\|\right)^2\right) \tag{6.30}$$

$$\leq \mathop{\mathbb{E}}_V\left(\sum_{t\in[k-1]} \|A_t|0\rangle - A_{t-1}|0\rangle\|^2\right)\left(\sum_{t\in[k-1]} 1^2\right) \leq \frac{4k}{n^7} \cdot k. \tag{6.31}$$

For all V, $|\|P_V A_{k-1}|0\rangle\| - \|P_V A|0\rangle\|| \leq \|P_V A_{k-1}|0\rangle - P_V A|0\rangle\| = \|P_V(A_{k-1}|0\rangle - A|0\rangle)\| \leq \|A_{k-1}|0\rangle - A|0\rangle\|$.

So $\mathbb{E}_V\left(\left(\|P_V A_{k-1}|0\rangle\| - \|P_V A|0\rangle\|\right)^2\right) \leq \frac{4k^2}{n^7}$. By Markov's inequality, $\Pr_V\left(\left(\|P_V A_{k-1}|0\rangle\| - \|P_V A|0\rangle\|\right)^2 \geq \frac{1}{n^4}\right) \leq \frac{4k^2}{n^3}$. So it is overwhelmingly likely that $\|P_V A|0\rangle\| - \|P_V A_{k-1}|0\rangle\| \leq \frac{1}{n^2}$, which implies $\|P_V A|0\rangle\|^2 - \|P_V A_{k-1}|0\rangle\|^2 \leq \frac{2}{n^2}$ since both norms are at most 1. Even assuming that in the unlikely cases the difference is the maximum possible, we still get $\mathbb{E}_V(\|P_V A|0\rangle\|^2 - \|P_V A_{k-1}|0\rangle\|^2) \leq \frac{4k^2}{n^3} + \frac{2}{n^2}$.

We can now use linearity of expectation and upper bound our required probability as

$$\mathop{\Pr}_{V,x\sim p_V}(f_V(x) - f_V(x^*) \leq \epsilon) = \mathop{\mathbb{E}}_V(\|P_V A|0\rangle\|^2) \leq 2e^{-\Omega(k^2)} + \frac{4k^2}{n^3} + \frac{2}{n^2}. \tag{6.32}$$

□

Note that this establishes a statement similar to Theorem 6.1.5, except with a polynomially larger value of n. This result is sufficient to establish the optimality of gradient descent in the

dimension-independent setting. In the next section we quantitatively improve the lower bound by reducing the value of n.

6.4.5 Improved lower bound using the wall function

In this section we improve the dimension dependence of the previous lower bound using the strategy used by [BJL+19], where they introduce a function called the *wall function*. We now provide a high-level overview of this strategy before getting into the details.

The previous construction required a larger dimension n because we needed to use a large value of γ, which in turn was large because we wanted the following key property (i.e., Lemma 6.4.5) to hold: If you query the function $f_V(x)$ with a random vector $x \in \mathbb{R}^n$, the function is almost certainly maximized on the first term in the max, and the answer of the gradient oracle is v_1. To reduce the parameter n, we will use a different function in this section. This function will be built out of the functions $p_V : \mathbb{R}^n \to \mathbb{R}$, where $V = (v_1, v_2, \ldots, v_k)$ is again a set of k orthonormal vectors:

$$p_V(x) := \max_{i \in [k]} \{\langle v_i, x\rangle - i\gamma\}, \tag{6.33}$$

where γ is unspecified for now. If we only allow the algorithm to query the oracle with a vector x with $\|x\| = 1$, this function is essentially the same as the function f_V we used in the previous section, up to an additive $k\gamma$ term. Allowing the algorithm to query p_V at vectors x with $\|x\| \leq 1$ is fine too, since our key property will still hold: Querying the gradient oracle with a random x with norm less than 1 will still return v_1 almost certainly. But if we allow the algorithm to query with vectors x with extremely large norm, the additive term $i\gamma$ will be negligible, and the property we want (that the answer is almost certainly v_1) will not hold anymore.

The wall function construction is a way of fixing this problem. The wall function constrains the set of points that can be queried to gain useful information about the set V. At the beginning, when the algorithm does not know the set V, the wall function essentially forces the algorithm to query the oracle with vectors x with small norm. If the oracle is queried with a vector of large norm, the wall function "hides" information about the set V by outputting an answer that (with high probability) is independent of V. More generally, if the algorithm has learned a subset of V, and the algorithm queries the oracle with a vector x with a large projection outside of the span of the vectors it knows, then (with high probability) the oracle's answer hides information about V. In this setting, querying a unit vector at random would be inadvisable since the whole vector would be outside of the span of the vectors the algorithm knows, and the oracle's response will be non-informative. The useful queries will be shorter vectors which do not trigger the wall function's obfuscation, since any projection of a short vector is also short. This restriction on the query vector length now allows us to choose a smaller value of γ than in the previous construction, and hence have a smaller dimension n.

Formal construction. We now describe the construction formally. As in the previous section, $V = (v_1, \ldots, v_k)$ is a set of k orthonormal vectors in $\mathbb{R}^n$. Our family of functions will depend on several parameters (n, k, δ, and γ), which are all functions of ϵ, which is the single parameter on which the function family depends.

Let us start with k. As before, we will show a lower bound of $\Omega(k)$, and so we want k to be a small multiple of $1/\epsilon^2$. Thus we choose $k := \frac{1}{100\epsilon^2}$. For some large enough constant c, we set

$$n := ck^2 \log k = \widetilde{O}\left(\frac{1}{\epsilon^4}\right). \tag{6.34}$$

This is chosen to satisfy eq. (6.37). Let δ be chosen such that

$$\frac{\delta}{\log(1/\delta)} := 32\sqrt{\frac{k \log n}{n}} + \frac{1}{\sqrt{k}} = \Theta\left(\frac{1}{\sqrt{k}}\right). \tag{6.35}$$

This value is chosen to make the first property in Lemma 6.4.10 hold. Let p_V be the function defined in eq. (6.33) with γ defined as

$$\gamma := 8\delta\sqrt{\log n/n}. \tag{6.36}$$

This value is chosen for a similar reason to before, and more precisely it is required in Lemma 6.4.11. As in the previous lower bound (and for the same reason), we want

$$k\gamma \leq \frac{1}{10\sqrt{k}}. \tag{6.37}$$

Our choice of n in eq. (6.34) satisfies eq. (6.37).

Before constructing the wall function, we need to define the *correlation cones* $C_1, \ldots, C_k$, which depend on $v_1, \ldots, v_k$:

$$C_i := \left\{ x \in \mathbb{R}^n \,\middle|\, \frac{|\langle v_i, x\rangle|}{\|x\|} \geq 8\sqrt{\frac{\log n}{n}} \right\}. \tag{6.38}$$

Note that if you choose a random unit vector x, it will most likely not be in C_i since the normalized inner product will be roughly $1/\sqrt{n}$. Thus C_i is the set of directions that correlate strongly with v_i.

We define the set

$$\Omega = \{x \in \mathbb{R}^n \mid \|x\| \in [\delta, 1] \wedge \forall i \in [k],\ x \notin C_i\} \tag{6.39}$$

to be the set of vectors that have non-negligible norm and are not in any of the correlation cones C_i. We want our construction to give non-informative answers on Ω so that the algorithm is forced to query on the complement of Ω.

We now define our non-informative function $h(x) = 2\|x\|^{1+\alpha}$, with $\alpha > 0$ set so that $\delta^\alpha = 1/2$. Note that because of the value of δ chosen, $1/\alpha = \Theta(\log n)$, so α is small. We want our wall function to be equal to $h(x)$ on Ω, but we need to define it everywhere in $\mathbb{R}^n$. We do so by extending this function to all of $\mathbb{R}^n$ by convexity. Informally this means that the function takes the smallest value it can outside Ω while remaining convex. Formally, we define the wall function as

$$\mathcal{W}_V(x) := \max_{y\in\Omega}\{h(y) + \langle \nabla h(y), x - y\rangle\}. \tag{6.40}$$

In the following lemma, we state some properties of the wall function established by [BJL+19].

Lemma 6.4.10 (Properties of the wall function)**.** *The wall function satisfies the following properties.*

1. *[BJL+19, Lemma 2]: The point $\tilde{x} = -\sum_{i\in[k]} v_i/\sqrt{k}$ satisfies $\mathcal{W}(\tilde{x}) \le -1/\sqrt{k}$.*
2. *[BJL+19, Lemma 3]: Let $x \in \mathbb{R}^n$. For any $t \in [k]$, let $x = w + z$, where $w \in \mathsf{span}(v_1, \ldots, v_t)$ and $z \in \mathsf{span}(v_1, \ldots, v_t)^\perp$. If $\forall j > t$, $z \notin C_j$, then $\mathcal{W}_V(x)$ does not depend on $v_{t+1}, \ldots, v_k$. For such x, $\mathcal{W}_V(x)$ takes the same value as the following function.*

$$\mathcal{W}_V^{(t)}(x) = \max_{a,b\in\mathbb{R}_+ : a^2+b^2\in[\delta^2,1]} \left\{ -2\alpha c^{1+\alpha} + 2\frac{1+\alpha}{c^{1-\alpha}} \left(\max_{y\in\tilde{\Omega}_{a,b}, \|y\|=a} \langle y, w\rangle + b\|z\| \right) \right\} \tag{6.41}$$

   where $c = \sqrt{a^2+b^2}$ and $\tilde{\Omega}_{a,b} = \{x \in \mathsf{span}(v_1, \ldots, v_t) \mid \forall i \le t \ \frac{|\langle v_i, x\rangle|}{\|x\|}\frac{a}{\sqrt{a^2+b^2}} < 8\sqrt{\log n/n}\}$.[6]
3. *Discussion after [BJL+19, Lemma 3]: Furthermore, if $\forall j > t$, $z \notin C_j$ and $\|z\| \ge \delta$, then $\max_{i\in[k]}\langle v_i, x\rangle - i\gamma \ne \max_{i\in[t]}\langle v_i, x\rangle - i\gamma$ implies that $\mathcal{W}_V(x) \ge p_V(x)$.*

The second property in the lemma implies that the value of the wall function on x grows with z, the uncorrelated projection of x. The third item states that if z is somewhat large and the maximum in the definition of p_V is achieved at an index with $i > t$, then $\mathcal{W}_V(x)$ is actually larger than $p_V(x)$.

Equipped with these properties, we define the actual class of functions. For a set V of k orthonormal vectors, we define

$$f_V(x) := \max\{p_V(x), \mathcal{W}_V(x)\}. \tag{6.42}$$

Note that $\mathcal{W}_V(x)$ is also convex, being a maximum of linear functions. The maximum norm of the gradient of any of the linear functions is $2(1+\alpha)$ and hence the function is 3-Lipschitz. Each of the p_V functions is 1-Lipschitz. Hence f_V is also 3-Lipschitz. This completely specifies the function f_V that we will use for a given value of ϵ.

[6] Lemma 3 in [BJL+19] gives a different definition of $\tilde{\Omega}$, but we believe this is the set they meant to define.

We also define the following functions.

$$p_V^{(t)}(x) := \max_{i\in[t]}\{\langle v_i, x\rangle - i\gamma\} \text{ and } f_V^{(t)}(x) = \max\{p_V^{(t)}(x), \mathcal{W}_V^{(t)}(x)\}. \tag{6.43}$$

Note that if the preconditions in item 2 of Lemma 6.4.10 are satisfied for some value t, then they are also satisfied for $t+1$. So for such x, $\mathcal{W}_V(x) = \mathcal{W}_V^{(t)}(x) = \mathcal{W}_V^{(t+1)}(x)$.

Lemma 6.4.10 implies some very convenient statements. Let $v_1, \ldots, v_k$ be fixed orthonormal vectors. Then for any point $x = w+z$, where $w \in \mathsf{span}(v_1, \ldots, v_{t-1})$ and $z \in \mathsf{span}(v_1, \ldots, v_{t-1})^\perp$ we can make the following statements.

- If $\|z\| \geq \delta$ and $\forall j \geq t,\ z \notin C_j$,
  then $\mathcal{W}_V(x) = \mathcal{W}_V^{(t-1)}(x)$ and also $p_V(x) \neq p_V^{(t-1)}(x) \implies \mathcal{W}_V(x) \geq p_V(x)$. Hence $f_V(x) = f_V^{(t-1)}(x)$.
- If $\|z\| < \delta$ and $\forall j \geq t,\ z \notin C_j$ and $p_V(x) = p_V^{(t)}(x)$,
  then $\mathcal{W}_V(x) = \mathcal{W}_V^{(t-1)}(x)$ and $p_V(x) = p_V^{(t)}(x)$. So $f_V(x) = f_V^{(t)}(x)$.

We now show the following lemma, akin to Lemma 6.4.5.

Lemma 6.4.11. *Let $1 \leq t \leq k$ be integers and $\{v_1, \ldots, v_{t-1}\}$ be a set of orthonormal vectors. Let $\{v_t, \ldots, v_k\}$ be chosen uniformly at random so that the set $\{v_1, \ldots, v_k\}$ is orthonormal. Then*

$$\forall x \in \mathbb{R}^n : \Pr_{v_t,\ldots,v_k}\left(f_V(x) \neq f_V^{(t)}(x)\right) \leq \frac{1}{n^7}. \tag{6.44}$$

Proof. Let $x = w + z$, where $w \in \mathsf{span}(v_1, \ldots, v_{t-1})$ and $z \in \mathsf{span}(v_1, \ldots, v_{t-1})^\perp$. Let E_x denote the event whose probability we want to upper bound.

If $\|z\| \geq \delta$, then E_x can only occur if $z \in C_j$ for some $j \geq t$. Using Proposition 6.4.4, the fact that each v_j in this range is chosen uniformly at random from an $n-t+1$-dimensional ball and a union bound, this probability is upper bounded by $k \cdot 2e^{-(n-t+1)\cdot 32\frac{\log n}{n}} \leq 2k \cdot 2^{-32\log n} = 2k/n^{32} \leq 1/n^{31}$, with the inequalities holding because $n > 4k$.

If $\|z\| \leq \delta$, then E_x can only occur if $z \in C_j$ for some $j \geq t$ *or* $p_V(x) \neq p_V^{(t)}(x)$. The former probability we have already upper bounded by $1/n^{31}$. The latter probability can be upper bounded as follows. Let E'_x refer to the event $p_V(x) \neq p_V^{(t)}(x)$. If E'_x occurs then it must hold that

$$\max_{i\in\{t+1,\ldots,k\}}\langle v_i, x\rangle - i\gamma = \max_{i\in\{t+1,\ldots,k\}}\langle v_i, z\rangle - i\gamma > \max_{i\in[t]}\langle v_i, x\rangle - i\gamma \geq \langle v_t, z\rangle - t\gamma. \tag{6.45}$$

We will show that this event is very unlikely. To do so, let F_x be the event that for all $i \in \{t, \ldots, k\}$, $\langle v_i, z\rangle \in [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$. Note that if F_x occurs, then the terms in the max are in decreasing order, and we have

$$\langle v_t, z\rangle - t\gamma \geq \langle v_{t+1}, z\rangle - (t+1)\gamma \geq \cdots \geq \langle v_{k-1}, z\rangle - (k-1)\gamma \geq \langle v_k, z\rangle - k\gamma, \tag{6.46}$$

which contradicts the previous equation. Thus if E'_x holds then the complement of F_x, $\bar{F}_x$ must hold, which means $\Pr(E'_x) \leq \Pr(\bar{F}_x)$. So let us show that F_x is very likely.

The event $\bar{F}_x$ holds if for any $i \in \{t, \ldots, k\}$, $\langle v_i, z \rangle \notin [-\frac{\gamma}{2}, +\frac{\gamma}{2}]$. We can upper bound this probability for any particular $i \in \{t, \ldots, k\}$ using Proposition 6.4.4 and the fact that v_i is chosen uniformly at random from an $n - t + 1$-dimension ball. This is the same as the probability that $\langle v_i, z/\delta \rangle \notin [-4\sqrt{\log n/n}, 4\sqrt{\log n/n}]$. Since $z/\delta \in B(\vec{0}, 1)$, this is at most $2e^{-(n-t+1)\cdot 8\frac{\log n}{n}} \leq 2 \cdot 2^{-8 \log n} = 2/n^8$, with the inequality holding because $n > 4t$. The probability that this happens for any i is at most k times this probability, by the union bound. Using the fact that $4k < n$, we get that $\Pr(E'_x) \leq \Pr(\bar{F}_x) < 1/2n^7$.

Putting it all together, we can upper bound the probability in the lemma statement by the maximum of $1/n^{31}$ and $1/n^{31} + 1/2n^7$, and so the lemma follows. □

Letting $\tilde{x} = -\sum_{i \in [k]} v_i/\sqrt{k}$, it is clear that $p_V(\tilde{x}) \leq -1/\sqrt{k}$. We have also seen that $\mathcal{W}_V(\tilde{x}) \leq -1/\sqrt{k}$. So $f_V(\tilde{x}) \leq -1/\sqrt{k} = -10\epsilon$. Any point x minimizing f_V to within ϵ of the optimum must satisfy $p_V(x) \leq -9\epsilon$. From eq. (6.37), we see that $k\gamma \leq \epsilon$. So the point x must also satisfy $\langle v_k, x \rangle \leq -8\epsilon$. Using this, we get the following analog of Lemma 6.4.6, whose proof is identical.

Lemma 6.4.12. *Let $k > 0$ be an integer and $\{v_1, \ldots, v_{k-1}\}$ be a set of orthonormal vectors. Let v_k be chosen uniformly at random from $\mathsf{span}(v_1, \ldots, v_{k-1})^\perp$ and let $V = (v_1, \ldots, v_k)$. Then*

$$\forall x \in B(\vec{0}, 1) : \Pr_{v_k} (f_V(x) - f_V(x^*) \leq \epsilon) \leq 2e^{-\Omega(k)}. \tag{6.47}$$

Finally, since the lemmas in the proof of the previous section's quantum lower bound (Section 6.4.4) used these two lemmas as a black box, the same proof allows us to argue that no algorithm can perform well if it makes at most $k - 1$ queries to the oracle. This completes the proof of Theorem 6.1.5.

6.5 Lower Bounds in Small Dimensions

Here we show some lower bounds on the query complexity of first-order convex optimization (Problem 6.1.1) when $n \leq O(1/\epsilon^4)$ (assume $G, R \leq O(1)$). We start with a lower bound that follows from our lower bound in the previous section.

Theorem 6.5.1. *Fix any n, ϵ such that $n \leq O(1/\epsilon^4)$. Solving Problem 6.1.1 on n-dimensional functions with accuracy ϵ requires query complexity at least $\widetilde{\Omega}(\sqrt{n})$.*

Proof. Let ϵ' be the smallest value such that when plugged into Theorem 6.1.5 with $G = R = 1$, we get a class of functions of dimension n for which solving Problem 6.1.1 to accuracy ϵ' requires $\Omega(1/\epsilon'^2)$ queries. In that proof, n turns out to be $\Theta(\log(1/\epsilon')/\epsilon'^4)$. Hence $\epsilon' > \epsilon$, and the complexity of ϵ-optimization of this function class is at least the complexity of its ϵ'-optimization, which we know is $\Omega(1/\epsilon'^2) = \widetilde{\Omega}(\sqrt{n})$. □

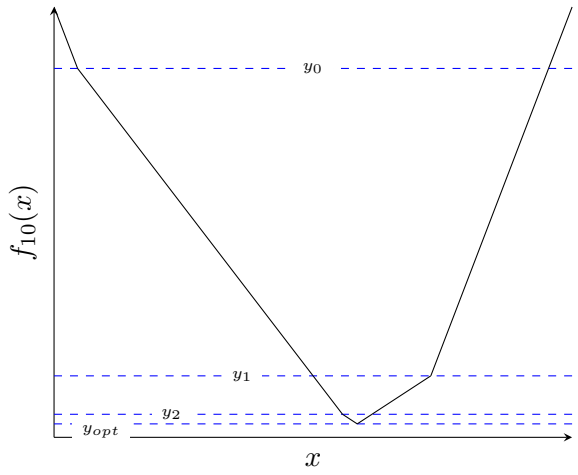


Figure 6.1: An inaccurate but representative depiction of the function $f_{10}(x)$.

We now analyze the dependence on ϵ of quantum algorithms solving Problem 6.1.1 in low dimensions. We show that quantum algorithms require a $\log(1/\epsilon)$ dependency on ϵ by exhibiting a function in 1-dimension that requires $\log(1/\epsilon)$-queries in order to ϵ-optimize. This rules out any quantum query algorithm with complexity $c(n)o(\log(1/\epsilon))$ for any function c.

In what follows, $\{0,1\}^{\leq t}$ is the set $\cup_{i\in[0,\ldots,t]}\{0,1\}^i$ and for a string w, $w_{\leq i}$ is the prefix of w of length i. ϵ denotes the empty string.

6.5.1 The 1-dimensional function

Consider the family of functions $\mathcal{F} = \{f_z\}_{z\in\{0,1\}^t}$ with f_z defined below. (See Figure 6.1 for a pictorial representation.)

Define the intervals $\{I_w\}_{w\in\{0,1\}^{\leq t}}$ recursively as

- $I_\epsilon = [-1, 1]$.
- If $I_w = [p_\ell, p_r]$, then $I_{w0} = \left[\frac{4p_\ell}{5} + \frac{p_r}{5}, \frac{3p_\ell}{5} + \frac{2p_r}{5}\right]$ and $I_{w1} = \left[\frac{2p_\ell}{5} + \frac{3p_r}{5}, \frac{p_\ell}{5} + \frac{4p_r}{5}\right]$. (In words, divide I_w into five equal intervals. I_{w0} is the second interval and I_{w1} is the fourth interval.)

It is easy to see that

- $|I_{w0}| = |I_{w1}| = |I_w|/5$, and so $I_w = 2/5^{|w|}$.
- If w is a prefix of w', then $I_{w'} \subset I_w$.
- $I_{w0} \cap I_{w1} = \emptyset$.

We also define the following values.

- $y_0 = 0$,

- $y_1 = -2/5$,
- $y_2 = y_1 - 2/(5 \cdot 15)$,
- $y_3 = y_2 - 2/(5 \cdot 15^2)$ and so on until
- $y_t = y_{t-1} - 2/(5 \cdot 15^{t-1})$.
- Finally $y_{opt} = y_t - 1/15^t$.

Definition 6.5.2. *For a $z \in \{0,1\}^t$, let $f_z : \mathbb{R} \to \mathbb{R}$ be the piecewise linear function defined by connecting the following values with lines.*

- *For $|x| > 1$, $f_z(x) = |x| - 1$.*
- *If $I_{z_{\leq i}} = [p_\ell, p_r]$, then $f_z(p_\ell) = f_z(p_r) = y_i$. For instance, $f_z(-1) = f_z(1) = 0$.*
- *If $I_z = [p_\ell, p_r]$, then $f_z(\frac{p_\ell + p_r}{2}) = y_{opt}$.*

We make the following observations about f_z.

Observation 6.5.3. *1. f_z is convex and 1-Lipschitz.*

2. *f_z takes its minimum value at the midpoint of the interval I_z, where it takes the value y_{opt}.*
3. *To solve Problem 6.1.1 on f_z with $\epsilon \leq 1/15^t$, it is necessary that the algorithm outputs a point in I_z.*

Proof. We prove part 1 by computing the gradients (also referred to as slopes here) of f_z. Let $i \leq t-1$, $I_{z_{\leq i}} = [p_\ell, p_r]$ and $I_{z_{\leq i+1}} = [q_\ell, q_r]$. (Note that $p_\ell < q_\ell < q_r < p_r$.) The slope of f_z between p_ℓ and q_ℓ is either $-\frac{2/(5 \cdot 15^i)}{2/5^{i+1}} = -1/3^i$ if $z_{i+1} = 0$ or $-\frac{2/(5 \cdot 15^i)}{3 \cdot 2/5^{i+1}} = -1/3^{i+1}$ if $z_{i+1} = 1$. Similarly the slope between q_r and p_r is either $1/3^{i+1}$ or $1/3^i$.

If $I_z = [p_\ell, p_r]$, the slope between p_ℓ and $\frac{p_\ell + p_r}{2}$ is $-\frac{1/15^t}{1/5^t} = -1/3^t$ and the slope between $\frac{p_\ell + p_r}{2}$ and p_r is $1/3^t$. Hence the slope is non-decreasing and f_z is convex.

Parts 2 and 3 follow from the definition of the function. □

Note that the gradient may not be defined at some points, which must be the left or right endpoints of some interval $I_w = [p_\ell, p_r]$. Our subgradient oracle will answer subgradient queries at p_ℓ by giving the gradient that f_z has at the right of p_ℓ and subgradient queries at p_r by giving the gradient that f_z has at the left of p_ℓ.

We now see some important facts about the outputs of the oracles.

Theorem 6.5.4. *For an $x \in [-1,1]$, the value and subgradient of f_z at input x is decided by i_x and $z_{\leq i_x+1}$ where $i_x \in [0, \ldots, t-1]$ is the unique number such that $x \in I_{z_{\leq i}}$ but $x \notin I_{z_{\leq i+1}}$ (and i_x is equal to t if $x \in I_z$).*

Proof. Suppose we had the values of i_x and $z_{\leq i_x+1}$. Knowing $z_{\leq i_x+1}$, we also know $I_{z_{\leq i_x}}$ and $I_{z_{\leq i_x+1}}$. By the definition of the function class, the only points at which we do not know the value and subgradients of f_z are the points in $I_{z_{\leq i_x+1}}$.

□

The Lower Bound

We show our lower bound by reducing to the following problem.

Problem 6.5.5 (Find The First Difference)**.** *Given a string $z \in \{0,1\}^t$, define the 'first difference' function $d_z : \{0,1\}^t \to [n] \cup \perp$ such that $d_z(z') = \min\{i \in [t] \mid z_i \neq z'_i\}$, or $\perp$ if $z = z'$. With query access to d_z, the task is to find the string z.*

It was shown by van Apeldoorn, Gilyén, Gribling and de Wolf [vAGGdW20, Theorem 26] that the above problem requires $\Omega(t)$ queries even for a quantum query algorithm. We show an $\Omega(t)$ lower bound for our optimization task by a reduction to this problem.

Theorem 6.5.6. *Given a q-query quantum algorithm for optimizing a function from class f_z to within $1/15^t$ of the optimum, we get a $2q$-query quantum algorithm that solves Problem 6.5.5.*

Proof. We show that we can deterministically simulate the f_z oracles using two calls to d_z, where the simulation does not know the value of z. To do this, we give two useful deterministic functions Enc and Dec that translate our optimization queries to queries for Problem 6.5.5 and translate answers for Problem 6.5.5 to answers for our optimization queries.

- Enc takes as input an input $x \in \mathbb{R}$ and outputs a $z' \in \{0,1\}^t$.
- Dec takes as input a $x \in \mathbb{R}, z' \in \{0,1\}^t$ and an $i \in [t] \cup \{\perp\}$ and outputs $y, g \in \mathbb{R}$.
- The two real numbers $\mathsf{Dec}(x, \mathsf{Enc}(x), d_z(\mathsf{Enc}(x)))$ are $f_z(x)$ and the subgradient of f_z at x that the subgradient oracle would have output.
- For any x that optimizes f_z to within $1/15^t$ of the optimum, $\mathsf{Enc}(x)$ is a string z' such that $d_z(z') = \perp$.

Given such functions, any query algorithm solving our optimization problem can be transformed into a query algorithm solving Problem 6.5.5 with at most twice the number of queries, thus completing the proof. We take a few moments to detail this transformation before showing the existence of the functions.

> Take a quantum query algorithm that optimizes f_z to within $1/15^t$ of the optimum. We modify this circuit by replacing its oracle calls O_{f_z} with the unitaries U_{Enc}, O_{d_z} and U_{Dec} (to be defined shortly). We also add the unitary U_{Enc} at the end of the query algorithm to the output register. Formally, let O_{f_z} operate on registers $in, val, grad$ and O_{d_z} operate on registers in_d, out_d. We define the unitaries U_{Enc} and U_{Dec} as arbitrary unitaries that satisfy the following.
>
> - $U_{\mathsf{Enc}} : |x\rangle_{in}|0\rangle_{in_d} \mapsto |x\rangle_{in}|\mathsf{Enc}(x)\rangle_{in_d}$ for any $x \in \mathbb{R}$ that is representable in the basis states of the

register in. Since Enc is a deterministic computation, this is easily computable and reversible and can hence be a unitary.

- $U_{\mathsf{Dec}} : |x\rangle_{in}|\mathsf{Enc}(x)\rangle_{in_d}|d_z(\mathsf{Enc}(x))\rangle_{out_d}|a\rangle_{val}|b\rangle_{grad} \mapsto |x\rangle_{in}|0\rangle_{in_d}|0\rangle_{out_d}|a \oplus f_z(x)\rangle_{val}|b \oplus \delta f_z(x)\rangle_{grad}$ for any $x \in \mathbb{R}$ representable in the basis states of the register in, any basis state a in val and b in $grad$, where $\delta f_z(x)$ is the subgradient that would have been returned by the oracle. To see that this can be implemented via a unitary requires more work. Since Dec is a deterministic computation, we can implement the operation $|x\rangle_{in}|\mathsf{Enc}(x)\rangle_{in_d}|d_z(\mathsf{Enc}(x))\rangle_{out_d}|a\rangle_{val}|b\rangle_{grad} \mapsto |x\rangle_{in}|\mathsf{Enc}(x)\rangle_{in_d}|d_z(\mathsf{Enc}(x))\rangle_{out_d}|a \oplus f_z(x)\rangle_{val}|b \oplus \delta f_z(x)\rangle_{grad}$. Following this, we use another query to O_{d_z} (which is its own inverse, as query oracles happen to be) to map $|\mathsf{Enc}(x)\rangle_{in_d}|d_z(\mathsf{Enc}(x))\rangle_{out_d} \mapsto |\mathsf{Enc}(x)\rangle_{in_d}|0\rangle_{out_d}$ and then an inverse of U_{Enc} to map $|x\rangle_{in}|\mathsf{Enc}(x)\rangle_{in_d} \mapsto |x\rangle_{in}|0\rangle_{in_d}$.

Let $|\psi\rangle$ be an arbitrary quantum state of the original quantum algorithm. Note that $U_{\mathsf{Dec}}O_{d_z}U_{\mathsf{Enc}}|\psi\rangle|0\rangle_{in_d}|0\rangle_{out_d} = (O_{f_z}|\psi\rangle)\,|0\rangle_{in_d}|0\rangle_{out_d}$. Equipped with these unitaries, any quantum query algorithm on registers in, val, $grad$ and an ancillary register anc making queries to O_{f_z} is replaced with a quantum query algorithm on registers in, val, $grad$, anc, in_d and out_d making queries to O_{d_z} which simulates the original algorithm, i.e. if the original algorithm reaches state $|\psi\rangle$, the simulated algorithm reaches state $|\psi\rangle|0\rangle_{in_d}|0\rangle_{out_d}$. Assuming for simplicity that the original algorithm stores its final output in the register in, the simulated algorithm stores its final output in the register in_d, and the probability of the latter giving a correct output is at least the probability of the former giving a correct output.

This leaves us with proving that the functions Enc and Dec do exist. We take Enc to be the function mapping x to $w0^{t-|w|}$ where $w = \arg\max_{w' \in \{0,1\}^{\leq t}, x \in I_{w'}}\{|w'|\}$ (note that the maximizer is unique).

To define Dec, we look at the output of $d_z(\mathsf{Enc}(x))$. If it is $\perp$, then the value of z is revealed. If it is $i \in [t]$, then we know $z_{\leq i}$. From Theorem 6.5.4, x and $z_{\leq i}$ decides $f_z(x)$ and the subgradient of f_z at x as long as $x \notin I_{z_{\leq i}}$. We know $x \notin I_{z_{\leq i}}$ since if it were, the value of $\mathsf{Enc}(x)$ would have had $z_{\leq i}$ as a prefix and so $d_z(\mathsf{Enc}(x))$ would have had to be larger than i. Hence we know $f_z(x)$ and the subgradient of f_z at x and the function Dec exists. □

Corollary 6.5.7. *Given an $\epsilon > 0$, let $t = \lfloor \log_{15}(1/\epsilon) \rfloor$. Solving Problem 6.1.1 on our function class from Definition 6.5.2 requires $\Omega(t) = \Omega(\log(1/\epsilon))$ queries.*

In the classical case, we know that $\approx n \log(1/\epsilon)$ is the complexity of first-order convex optimization when $n \leq O(1/\epsilon^2)$. Can we make a similar statement for the quantum complexity? In our thesis we could not show as strong a lower bound, but we do manage to show a lower bound of $\widetilde{\Omega}(\sqrt{n}\log(1/\epsilon))$ in the next section.

6.5.2 The n-dimensional function

The function here is quite similar to the 1-dimensional function, except that we make independent copies of it in each of the n-dimensions.

Definition 6.5.8. *Given $\overline{z} = (z^{(1)}, z^{(2)}, \ldots, z^{(n)}) \in (\{0,1\}^t)^n$, define $f_{\overline{z}} : \mathbb{R}^n \to \mathbb{R}$ as*

$$f_{\overline{z}}(x) = \max_{i \in [n]} f_{z^{(i)}}(x_i).$$

We consider the subgradient oracle for this function to be the one that outputs the subgradient corresponding to the minimum i that maximizes the expression above.

Note that the complexity of optimizing over this class of functions is at least $\Omega(t)$ since we can arbitrarily set the values of $z^{(2)}$ to $z^{(n)}$ and we'll still have the 1-dimensional function class to optimize over, requiring $\Omega(t)$ queries. We want to show a lower bound of $\widetilde{\Omega}(\sqrt{n}t)$. In the case where $t \geq 2^{\sqrt{n}}$ the lower bound of $\Omega(t)$ already shows such a lower bound. So from here on we will assume that $t < 2^{\sqrt{n}}$.

Let us first look at some subproblems that we want to accomplish using the oracles of $f_{\bar{z}}$. Consider, for any $i \in [t]$, the 'subproblem' of finding out $z_i^{(1)}, z_i^{(2)}, \ldots, z_i^{(n)}$. Since any $1/15^t$-optimal point x must lie in $I_{z^{(1)}} \times I_{z^{(2)}} \times \cdots \times I_{z^{(n)}}$ and these z values can be read off the point x, any successful algorithm has also solved each subproblem. Furthermore if x does not lie in $I_{z^{(1)}_{\leq i}} \times I_{z^{(2)}_{\leq i}} \times \cdots \times I_{z^{(n)}_{\leq i}}$ then the values of $z^{(1)}_{>i}$ do not come into play when computing the function value. We can use this property in order to come up with a hybrid argument.

We show that each subproblem needs around $\sqrt{n}$ queries to solve, and that without having solved one, queries barely get any information about subsequent subproblems. If instead no information was revealed about subsequent subproblems, then the algorithm has no choice but to solve each subproblem in succession, hence requiring around $\sqrt{n}t$ queries. The hybrid argument formalizes the statement that this is effectively the case even when only a little information is revealed about later subproblems.

We now start with analyzing the complexity of solving a subproblem.

Lemma 6.5.9. *Let $\bar{b} = (b_1, b_2, \ldots, b_n)$, with each $b_i \in \{0,1\}$ and define $f_{\bar{b}}$ as in Definition 6.5.8. Solving Problem 6.1.1 with $\epsilon = 1/15$ on the class of functions $f_{\bar{b}}$ requires $\Omega(\sqrt{n}/\log n)$ queries. Furthermore, for $\delta \geq 2^{-\sqrt{n}}$, $\Omega(\sqrt{n}/\log(n/\delta))$ queries are required to get even a δ probability of success when $\bar{b}$ is distributed uniformly from $\{0,1\}^n$.*

Proof. We prove this by showing that queries to $f_{\bar{b}}$ can be simulated by 'find the first difference with $\bar{b}$' queries wherein you may also specify a permutation of $[n]$ and 'first' is interpreted as first in the permuted list of bits. We will see that under this simulation, $1/15$-approximating $f_{\bar{b}}$ implies that $\bar{b}$ is learned. As stated in the discussion at the end of Section 6.3.1 the constant-error quantum query complexity of the task of learning $\bar{b}$ with such queries is $\Omega(\sqrt{n}/\log n)$. The argument is quite straightforward so we present it before showing that they simulate queries to $f_{\bar{b}}$.

Any of these 'find the first difference under this permutation' queries can be computed with a small error probability with $O(\sqrt{n})$ queries to the bits of $\bar{b}$. With $O(\sqrt{n}\log(\sqrt{n}\log(n/\delta)))$ queries to the bits of $\bar{b}$, the error can be made $\delta/\sqrt{n}$.

> We can use Grover's search to find out if there is a difference in the queried and actual bitstring in $O(\sqrt{n}\log(\sqrt{n}\log(n/\delta))) = O(\sqrt{n}\log(\sqrt{n}/\delta))$ queries with error $\leq \delta/\sqrt{n}\log n$. We then use binary search to find the first difference, the whole process taking $O\left(\sqrt{n} + \sqrt{n/2} + \ldots\right)\log(\sqrt{n}/\delta) = O(\sqrt{n}\log(\sqrt{n}/\delta))$ queries. The overall error is still $\leq \delta/\sqrt{n}$. That we can reduce the error, use subroutines multiple times

and have errors add up only linearly is guaranteed by the BQP Subroutine Theorem. (See the discussion before Section 2.3.1.)

We know from Lemma 2.3.3 that $\Omega(n)$ queries to the bits of $\overline{b}$ are required in order to learn $\overline{b}$. In fact, a quantum query algorithm $\mathcal{A}$ must make $\Omega(n)$ queries even to satisfy

$$\mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\mathcal{A} \text{ outputs } \overline{b}\right) \geq 2^{-\Omega(n)}.$$

So if there existed an algorithm $\mathcal{A}$ that made k queries to $f_{\overline{b}}$ and outputs $\overline{b}$ with probability at least δ, then there is an algorithm that makes $O(k\sqrt{n}\log(\sqrt{n}/\delta))$ queries to the bits of $\overline{b}$ and behaves similarly to $\mathcal{A}$ in that it outputs $\overline{b}$ with probability at least $\delta - k\cdot\delta/\sqrt{n} \geq \delta/2$ when $k < \sqrt{n}/2$. Since we are only dealing with $\delta \geq 2^{-\sqrt{n}} \gg 2^{-\Omega(n)}$, we know that k must be at least $\Omega\left(\frac{n}{\sqrt{n}\log(\sqrt{n}/\delta)}\right) \geq \Omega\left(\frac{\sqrt{n}}{\log(\sqrt{n}/\delta)}\right)$.

We now show the relationship between the two types of queries. Let $x = (x_1, x_2, \ldots, x_n)$ be a point in $\mathbb{R}^n$. If any x_i has absolute value larger than 1, we know the function value is $\max_i\{|x_i| - 1\}$. In order to compute $f_{\overline{b}}(x)$ on other inputs, we compute, for each $i \in [n]$, $y_{i,0} = f_0(x_i)$ and $y_{i,1} = f_1(x_i)$. Let $y_{i,max}$ and $y_{i,min}$ be the maximum and minimum of the two respectively. Note that

$$f_{\overline{b}}(x) = \max_{i\in[n]} f_{b_i}(x_i) = \max_{i\in[n]} y_{i,b_i}.$$

Our 'find the first difference under this permutation' query would then be specified by (a) the permutation specified by the decreasing order of $y_{i,max}$, and (b) the bitstring $\overline{b'}$ where $b'_i = 1$ if $y_{i,min} = y_{i,1}$ and 0 otherwise. (That is, we guess that $\overline{b}$ is set to minimize y_{i,b_i} for each index i.)

Let the answer of the first difference query be $j \in [n] \cup \{\perp\}$ and let S be the set of indices before j under the ordering. Then we have that

$$f_{\overline{b}}(x) = \max\{y_{j,max}, \max_{i\in S} y_{i,min}\}.$$

We can compute this since we know from the first difference query all the bits of $\overline{b}$ in indices $S \cup \{j\}$. The subgradient can also be computed since we know which the first index is that maximized the expression in the definition of $f_{\overline{b}}$.

It is also clear from the definition of $f_{\overline{b}}$ that $f_{\overline{b}}(x)$ is within $1/15$ of its optimal value if and only if $x \in \prod_{i\in[n]} I_{b_i}$ and on such an input x, the simulated query would involve $\overline{b'} = \overline{b}$, resulting in the output being $\perp$ and hence $\overline{b}$ would be learned.

(The details of how a simulation can be carried out in the setting of a quantum query algorithm is given in the proof of *Theorem* 6.5.6.) □

We now state a consequence of the above that will be very useful for our main hybrid

argument. Let $q \geq \Omega\left(\frac{\sqrt{n}}{\log(nt)}\right)$ be such that any quantum query algorithm making q queries has success at most $\frac{1}{100nt^5}$ for the problem in Lemma 6.5.9.

Corollary 6.5.10. *Choose an arbitrary function $g : \mathbb{R}^n \to \mathbb{R}$ such that for $x \notin \prod_{i\in[n]} I_{b_i}$, $f_{\overline{b}}(x) = g(x)$. Let $\mathcal{A}$ be a quantum algorithm making at most q queries to $f_{\overline{b}}$ and $\mathcal{A}'$ be the same as $\mathcal{A}$ but with queries to $f_{\overline{b}}$ replaced with queries to g. Then for any initial state $|\psi\rangle$,*

$$\mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\||\mathcal{A}|\psi\rangle - \mathcal{A}'|\psi\rangle\|^2\right) \leq 1/25t^5.$$

Proof. Fix a value of $\overline{b}$. Then for any state $|\phi\rangle = \sum_x \alpha_x|x\rangle|\phi_x\rangle$ in the appropriate registers,

$$O_{f_{\overline{b}}}|\phi\rangle - O_g|\phi\rangle = \sum_{x:f_{\overline{b}}(x)\neq g(x)} \alpha_x(O_{f_{\overline{b}}} - O_g)|x\rangle|\phi_x\rangle,$$

which has norm squared at most $\sum_{x:f_{\overline{b}}(x)\neq g(x)} |\alpha_x|^2 \cdot 4$. Since $f_{\overline{b}}(x) \neq g(x)$ implies that $x \in \prod_{i\in[n]} I_{b_i}$, we can upper bound $\sum_{x:f_{\overline{b}}(x)\neq g(x)} |\alpha_x|^2$ by the probability that measuring $|\phi\rangle$ gives us a point that 1/15-optimizes $f_{\overline{b}}(x)$.

Now let $\mathcal{A}$ be an arbitrary quantum algorithm making at most q queries. Without loss of generality, let it be $U_qO_{f_{\overline{b}}}U_{q-1}O_{f_{\overline{b}}}\cdots O_{f_{\overline{b}}}U_1O_{f_{\overline{b}}}U_0$. We define hybrid algorithms that swap some oracles with oracles for g.

- $\mathcal{A}_0$ is the same as $\mathcal{A}$.
- $\mathcal{A}_i$ replaces the last i oracle queries of $\mathcal{A}$ with oracle queries for g.

$$\mathcal{A}_i = U_qO_gU_{q-1}\cdots U_{q-i+1}O_gU_{q-i}O_{f_{\overline{b}}}U_{q-i-1}\cdots U_1O_{f_{\overline{b}}}U_0.$$

Note that $\mathcal{A}_q$ is the same as $\mathcal{A}'$. We now analyze, for any $i \in [q]$, $\mathbb{E}_{\overline{b}\in\{0,1\}^n}\left(\||\mathcal{A}_{i-1}|\psi\rangle - \mathcal{A}_i|\psi\rangle\|^2\right)$. For any $\overline{b}$, we look at the quantity $\||\mathcal{A}_{i-1}|\psi\rangle - \mathcal{A}_i|\psi\rangle\|^2$.

$$\begin{aligned}\||\mathcal{A}_{i-1}|\psi\rangle - \mathcal{A}_i|\psi\rangle\|^2 &= \|O_{f_{\overline{b}}}U_{q-i}O_{f_{\overline{b}}}\cdots U_1O_{f_{\overline{b}}}U_0|\psi\rangle - O_gU_{q-i}O_{f_{\overline{b}}}\cdots U_1O_{f_{\overline{b}}}U_0|\psi\rangle\|^2 \quad (6.48)\\ &= \|O_g|\phi\rangle - O_{\overline{b}}|\phi\rangle\|^2 \quad (6.49)\end{aligned}$$

where $|\phi\rangle = U_{q-i}O_{f_{\overline{b}}}\cdots U_1O_{f_{\overline{b}}}U_0|\psi\rangle$. As we saw earlier, this quantity is upper bounded by 4 times the probability that measuring the state $|\phi\rangle$ gives us a point that 1/15-optimizes $f_{\overline{b}}$. Taking the expectation over all the values of $\overline{b}$, we get that $\mathbb{E}_{\overline{b}\in\{0,1\}^n}\left(\||\mathcal{A}_{i-1}|\psi\rangle - \mathcal{A}_i|\psi\rangle\|^2\right)$ is upper bounded by 4 times the probability that the algorithm $U_{q-i}O_{f_{\overline{b}}}\cdots U_1O_{f_{\overline{b}}}U_0|\psi\rangle$ actually solves the problem in Lemma 6.5.9 when $\overline{b}$ is sampled uniformly at random. We know by the definition of q that this probability is at most $\frac{1}{100nt^5}$. We can now complete the proof with an

application of Cauchy-Schwarz.

$$\mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\|\mathcal{A}|\psi\rangle-\mathcal{A}'|\psi\rangle\|^2\right)=\mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\left\|\sum_{i\in[q]}\mathcal{A}_{i-1}|\psi\rangle-\mathcal{A}_i|\psi\rangle\right\|^2\right) \tag{6.50}$$

$$\leq \mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\left(\sum_{i\in[q]}\|\mathcal{A}_{i-1}|\psi\rangle-\mathcal{A}_i|\psi\rangle\|\right)^2\right) \tag{6.51}$$

$$= \mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\left(\sum_{i\in[q]}\|\mathcal{A}_{i-1}|\psi\rangle-\mathcal{A}_i|\psi\rangle\|\cdot 1\right)^2\right) \tag{6.52}$$

$$\leq \mathop{\mathbb{E}}_{\overline{b}\in\{0,1\}^n}\left(\left(\sum_{i\in[q]}\|\mathcal{A}_{i-1}|\psi\rangle-\mathcal{A}_i|\psi\rangle\|^2\right)\left(\sum_{i\in[q]}1^2\right)\right) \tag{6.53}$$

$$\leq \frac{q}{25nt^5}\cdot q\leq\frac{1}{25t^5}. \qquad \square$$

We can now formalize the set-up of our main hybrid argument. Let us define some intermediate functions. For each $i\in[t]$, define $f^{(i)}_{\overline{z}}=f_{\overline{z_{\leq i}}}$ where $\overline{z_{\leq i}}=(z^{(1)}_{\leq i},\ldots,z^{(n)}_{\leq i})$. Note that $f(x)=f^{(i)}(x)$ for all $x\notin I_{z^{(1)}_{\leq i}}\times I_{z^{(2)}_{\leq i}}\times\cdots\times I_{z^{(n)}_{\leq i}}$. *Hence for x outside that region, even the subgradient queries output the same value.*

> One important observation is the recursive nature of these functions. Given the values of $z^{(1)}_{<i},\ldots,z^{(n)}_{<i}$, the function $f^{(i)}_{\overline{z}}$ is 'equivalent' to the function $f_{\overline{b}}$ where $\overline{b}=(z^{(1)}_i,\ldots,z^{(n)}_i)$ with the following equivalence: The relevant domain of the former, $I_{z^{(1)}_{<i}}\times I_{z^{(2)}_{<i}}\times\cdots\times I_{z^{(n)}_{<i}}$, is linearly mapped to $[-1,1]^n$. The output of the former is shifted up by y_{i-1} (see Definition 6.5.2) and then scaled by 15^{i-1} to get the corresponding output of the latter.

Let $\mathcal{A}$ be an algorithm that claims to optimize $f_{\overline{z}}$ within tq queries. Without loss of generality, let $\mathcal{A}$ be

$$\mathcal{A}=\mathcal{A}_0=U_{tq}O_{f_{\overline{z}}}U_{tq-1}O_{f_{\overline{z}}}\cdots O_{f_{\overline{z}}}U_1O_{f_{\overline{z}}}U_0.$$

We consider the following hybrid algorithms.

Definition 6.5.11 (Hybrid algorithms)**.** *Define $\mathcal{A}_1$ to $\mathcal{A}_t$ as follows.*

- *$\mathcal{A}_1$ replaces the first q oracle queries of $\mathcal{A}_0$ (between U_q and U_0) with oracles for $f^{(1)}_{\overline{z}}$.*
- *$\mathcal{A}_2$ is the same as $\mathcal{A}_1$ except that it replaces the q oracle queries between U_{2q} and U_q with oracles for $f^{(2)}_{\overline{z}}$.*
- *Inductively we define, for $i\in[t]$, $\mathcal{A}_i$ to be the same as $\mathcal{A}_{i-1}$ except that it replaces the q oracle queries between U_{iq} and $U_{(i-1)q}$ with oracles for $f^{(i)}_{\overline{z}}$.*

We now show that the behaviour of $\mathcal{A}_{i-1}$ is similar to that of $\mathcal{A}_i$ for each $i\in[t]$, and that $\mathcal{A}_t$ does not compute the function. These follow from Corollary 6.5.10 and Lemma 6.5.9 and when put together, we get that A does not compute the function.

Lemma 6.5.12 ($\mathcal{A}_{i-1}$ and $\mathcal{A}_i$ have similar outputs)**.** *Let $i \in [t]$ and $\mathcal{A}_{i-1}$ and $\mathcal{A}_i$ be the hybrid algorithms as defined in Definition 6.5.11. Then for any state $|\psi\rangle$,*

$$\mathop{\mathbb{E}}_{\overline{z}}(\|\mathcal{A}_{i-1}|\psi\rangle - \mathcal{A}_i|\psi\rangle\|^2) \leq \frac{1}{25t^5}. \tag{6.54}$$

Proof. Let us arbitrarily fix the values of $z_{<i}^{(1)}, z_{<i}^{(2)}, \ldots, z_{<i}^{(n)}$ and of $z_{>i}^{(1)}, z_{>i}^{(2)}, \ldots, z_{>i}^{(n)}$. Let $z_i^{(1)}, z_i^{(2)}, \ldots, z_i^{(n)}$ be chosen uniformly at random. Note that with the fixed values, our algorithms need not make any queries before $U_{(i-1)t}$ since the oracles there depend only on fixed values and can be simulated.

A_{i-1} and A_i are different only because of q $f_{\overline{z}}$ oracles that have been replaced with $f_{\overline{z}}^{(i)}$ oracles. These two functions differ only in $\prod_{j\in[n]} I_{z_{\leq i}^{(j)}}$. Since we have fixed and hence know the values of $I_{z_{<i}^{(j)}}$, we may as well restrict ourselves to $\prod_{j\in[n]} I_{z_{<i}^{(j)}}$. Here the function $f_{\overline{z}}^{(i)}$ is merely a 'scaling and shifting' of the function $f_{\overline{b}}$ where $\overline{b} = (z_i^{(1)}, z_i^{(2)}, \ldots, z_i^{(n)})$.

Let $|\phi\rangle = U_{(i-1)q} O_{f_{\overline{z}}^{(i-1)}} \cdots O_{f_{\overline{z}}^{(1)}} U_0 |\psi\rangle$ be the state of the algorithms $\mathcal{A}_i$ and $\mathcal{A}_{i-1}$ after they go through the initial unitaries that they share. We can use Corollary 6.5.10 to conclude that for any state $|\psi\rangle$,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\overline{z}_i}(\|A_{i-1}|\psi\rangle - A_i|\psi\rangle\|^2) && (6.55)\\
&= \mathop{\mathbb{E}}_{\overline{z}_i}\left(\|O_{f_{\overline{z}}} U_{iq-1} \cdots U_{(i-1)q+1} O_{f_{\overline{z}}} |\phi\rangle - O_{f_{\overline{z}}^{(i)}} U_{iq-1} \cdots U_{(i-1)q+1} O_{f_{\overline{z}}^{(i)}} |\phi\rangle\|^2\right) && (6.56)\\
&\leq \frac{1}{25t^5}. && (6.57)
\end{aligned}$$

Since the uniform distribution for $\overline{z}$ is a convex combination of the uniform distributions for $\overline{z}_i$ with the various fixings of the other variables, this holds for the uniform distribution on $\overline{z}$ as well. □

Lemma 6.5.13 ($\mathcal{A}_t$ does not solve the task)**.** *Let $p'_{\overline{z}}$ be the probability that $\mathcal{A}_t$ outputs a valid $1/15^t$-optimal point when run on the function $f_{\overline{z}}$. Then*

$$\mathop{\mathbb{E}}_{\overline{z}}(p'_{\overline{z}}) \leq \frac{1}{100nt^5}. \tag{6.58}$$

Proof. Again we arbitrarily fix the values of $\overline{z}_{<t}$. The first $(t-1)q$ queries are made to oracles that only depend on these fixed values. Hence $\mathcal{A}$ becomes a q query algorithm querying the bits $\overline{z}_t$. Setting $\overline{b} = \overline{z}_t$, we see that the algorithm is optimizing a 'scaled and shifted' version of $f_{\overline{b}}$. We know from the definition of q that no q-query algorithm can optimize $f_{\overline{b}}$ with success probability at least $\frac{1}{100nt^5}$.

Since the uniform distribution for $\overline{z}$ is a convex combination of the uniform distributions for $\overline{z}_t$ with the various fixings of the other variables, this holds for the uniform distribution on $\overline{z}$ as well. □

Lemma 6.5.14 ($\mathcal{A}$ does not solve the task)**.** *Let $p_{\overline{z}}$ be the probability that $\mathcal{A}$ outputs a valid $1/15^t$-optimal point when run on the function $f_{\overline{z}}$. Then*

$$\mathop{\mathbb{E}}_{\overline{z}}(p_{\overline{z}}) \leq 2/t. \tag{6.59}$$

Proof. This proof is very similar to the proof of Lemma 6.4.9.

Let $P_{\overline{z}}$ be the projection operator that projects a quantum state $|\psi\rangle$ onto the space spanned by vectors $|x\rangle$ for x being $1/15^t$-optimal for $f_{\overline{z}}$. Then when the input is $f_{\overline{z}}$, $\|P_{\overline{z}}\mathcal{A}|0\rangle\|^2 = p_{\overline{z}}$. We know from Lemma 6.5.13 that $\mathbb{E}_{\overline{z}}(\|P\mathcal{A}_t|0\rangle\|^2) \leq \frac{1}{100nt^5}$. We prove our upper bound on the probability by showing that it is not far from $\mathbb{E}_{\overline{z}}(\|P_{\overline{z}}\mathcal{A}_t|0\rangle\|^2)$.

Lemma 6.5.12 states that for all $1 \leq t < k$, $\mathbb{E}_{\overline{z}}(\||A_{i-1}|0\rangle - A_i|0\rangle\|^2) \leq \frac{1}{25t^5}$. Using telescoping sums and the Cauchy-Schwarz inequality, we see that

$$\mathop{\mathbb{E}}_{\overline{z}}(\|\mathcal{A}_t|0\rangle - \mathcal{A}|0\rangle\|^2) \leq \mathop{\mathbb{E}}_{\overline{z}}\left(\left(\sum_{i\in[t]} \|\mathcal{A}_{i-1}|0\rangle - \mathcal{A}_i|0\rangle\|\right)^2\right) \tag{6.60}$$

$$\leq \mathop{\mathbb{E}}_{\overline{z}}\left(\sum_{i\in[t]} \|\mathcal{A}_{i-1}|0\rangle - \mathcal{A}_i|0\rangle\|^2\right)\left(\sum_{i\in[t]} 1^2\right) \leq \frac{t}{25t^5}\cdot t \leq \frac{1}{25t^3} \tag{6.61}$$

For all $\overline{z}$, $|\|P_{\overline{z}}\mathcal{A}_t|0\rangle\| - \|P_{\overline{z}}\mathcal{A}|0\rangle\|| \leq \|P_{\overline{z}}\mathcal{A}_t|0\rangle - P_{\overline{z}}\mathcal{A}|0\rangle\| = \|P_{\overline{z}}(\mathcal{A}_t|0\rangle - \mathcal{A}|0\rangle)\|$. Since $P_{\overline{z}}$ is a projection, this is at most $\|\mathcal{A}_t|0\rangle - \mathcal{A}|0\rangle\|$.

Hence $\mathbb{E}_{\overline{z}}((\|P_{\overline{z}}\mathcal{A}_t|0\rangle\| - \|P_{\overline{z}}\mathcal{A}|0\rangle\|)^2) \leq \frac{1}{25t^3}$. By Markov's inequality, $\Pr_{\overline{z}}((\|P_{\overline{z}}\mathcal{A}_t|0\rangle\| - \|P_{\overline{z}}\mathcal{A}|0\rangle\|)^2 \geq \frac{1}{25t^2}) \leq \frac{1}{t}$. So it is overwhelmingly likely that $\|P_{\overline{z}}\mathcal{A}|0\rangle\| - \|P_{\overline{z}}\mathcal{A}_t|0\rangle\| \leq \frac{1}{5t}$, which implies $\|P_{\overline{z}}\mathcal{A}|0\rangle\|^2 - \|P_{\overline{z}}\mathcal{A}_t|0\rangle\|^2 \leq \frac{2}{5t}$ since both norms are at most 1. Even assuming that in the unlikely cases the difference is the maximum possible, we still get $\mathbb{E}_{\overline{z}}(\|P_{\overline{z}}\mathcal{A}|0\rangle\|^2 - \|P_{\overline{z}}\mathcal{A}_t|0\rangle\|^2) \leq \frac{1}{t} + \frac{2}{5t}$.

We can now use linearity of expectation and upper bound our required probability as

$$\mathop{\mathbb{E}}_{\overline{z}}(p_{\overline{z}}) = \mathop{\mathbb{E}}_{\overline{z}}(\|P_{\overline{z}}\mathcal{A}|0\rangle\|^2) \leq \frac{1}{100nt^5} + \frac{1}{t} + \frac{2}{5t}. \tag{6.62}$$

□

We finish this section by viewing this from the lens of convex optimization.

Theorem 6.5.15. *For any $n \in \mathbb{N}, G, R, \epsilon > 0$, there exists a family of functions $f : \mathbb{R}^n \to \mathbb{R}$ with Lipschitz constant G in the ball of radius R such that any quantum algorithm that solves Problem 6.1.1 with high probability on this function family must make $\Omega(\sqrt{n}\log(GR/\epsilon\sqrt{n})/\log(n\log(GR/\epsilon\sqrt{n})))$ queries to f or $\mathcal{FO}(f)$ in the worst case.*

Proof. Let $t = \lfloor \log(1/\epsilon) \rfloor$. We showed in Lemma 6.5.14 that optimizing $f_{\overline{z}}$ in the domain $[-1,1]^n$ requires $\Omega(\sqrt{nt}/\log(nt))$ queries. Since this domain is inside the ball of radius $\sqrt{n}$ and the function is 1-Lipschitz, we have a family of functions with $GR/\epsilon = \sqrt{n}/\epsilon$. From the discussion after the definition of Problem 6.1.1, we see that its complexity is a function of n

and GR/ϵ. Hence our lower bound is actually for all G, R, ϵ and the lower bound is

$$\Omega\left(\frac{\sqrt{n}\log(\frac{GR}{\epsilon\sqrt{n}})}{\log(n\log(\frac{GR}{\epsilon\sqrt{n}}))}\right).$$

Note that if $G = R = 1$, then this lower bound only makes sense when $n < 1/\epsilon^2$. At $n = 1/2\epsilon^2$, it simplifies to $\widetilde{\Omega}(\sqrt{n})$ and when $n < 1/\epsilon^{2-\Omega(1)}$, it simplifies to $\widetilde{\Omega}(\sqrt{n}\log(1/\epsilon))$. □

We end this section by making an observation about the above task but in the classical setting. One can show a randomized lower bound of $\Omega(n\log(\frac{GR}{\epsilon\sqrt{n}}))$ for the same function family that we use above. The proof of this is almost exactly the proof of Lemma 6.3.4. One argues that if $\overline{z}$ were chosen uniformly at random from $(\{0,1\}^t)^n$, then every query would leave the posterior distribution as the uniform distribution over some subset of the tn bits. On expectation, one would learn at most 2 bits and so $\Omega(tn)$ queries are needed in order to query with constant probability a $1/15^t$-approximate solution.

6.6 Open problems

We showed that in the black-box setting, no quantum algorithm can beat gradient descent in general, in the dimension-independent regime. Here are some interesting questions left open by our work:

1. We showed in Theorem 6.1.4 that the class of functions used in the randomized lower bound can be solved faster with quantum queries. Is there a more interesting class of functions on which we can achieve a quantum speedup?

2. Can the quantum lower bound in Section 6.4 be made to work using the simpler class of functions $f_V(x) = \max_i \langle v_i, x\rangle$ (which is our function with $\gamma = 0$)? If so, this might also decrease the dimension n required.

3. We showed in Section 6.5 that even dimension-dependent algorithms can experience at best a quadratic speedup. Can we establish tighter quantum lower bounds in this regime showing that there is no speedup? When $1/\epsilon$ is a large polynomial in n, the complexity of gradient descent is also a large polynomial in n, but a dimension-dependent algorithm such as the center of gravity method [Bub15] yields an $O(n\log n)$ upper bound. Can we establish an $\widetilde{\Omega}(n)$ lower bound in this regime? This is essentially the same as the problem left open by [CCLW20, vAGGdW20], but phrased in the language of membership and separation oracles.

Chapter 7

Some Open Problems

In this chapter we collate some open problems that arise in various parts of this thesis.

As mentioned in Chapter 3, it was conjectured that every function with small randomized communication complexity has an *exact* representation that is *efficient* as measured by the γ_2 norm of its communication matrix. We also noted that there is a good reason for why we do not have a counterexample, and that there is only one known explicit function that can possibly refute this conjecture. We conjecture that this function does indeed refute the conjecture.

Define the function Integer Inner Product (IIP) as defined by [CLV19].

Definition 7.1. *The function* $\mathsf{IIP} : [-N, N]^6 \times [-N, N]^6 \to \{0, 1\}$ *is defined as*

$$\mathsf{IIP}(x_1, \ldots, x_6, y_1, \ldots, y_6) = 1 \text{ if and only if } \sum_{i \in [6]} x_i y_i = 0.$$

Conjecture 7.2: $\gamma_2(\mathsf{IIP})$

$\gamma_2(\mathsf{IIP}) \geq N^{\Omega(1)}$.

Some other conjectures we are interested in arise while studying randomized parity decision trees. The first of these is a fundamental question about the power of randomization in parity decision trees. Our conjecture is that all that randomized parity decision trees can do is balancing deterministic parity decision trees. We noted in Theorem 3.3.6 that the balancing of parity decision trees is in some sense captured by the notion of Subspace Decision Tree depth. Our conjecture is as follows.

Conjecture 7.3: Randomized PDTs are limited to Subspace Queries

For any total function $f : \{0,1\}^n \to \{0,1\}$, $\mathsf{R}^{\oplus}_{1/3}(f) \in \tilde{\Theta}(\mathsf{D}^{\wedge \oplus}(f))$.

We are also interested in a fascinating combinatorial phenomenon that occurs in decision trees, but that is currently not known in parity decision trees. In the proof of Lemma 5.1.2 it is shown that if we partition the n-dimensional Boolean hypercube into c subcubes, then there is a decision tree with at most $2^{\mathsf{polylog}(c,n)}$ leaves such that the leaves form a refinement of the partition. We conjecture the equivalent statement is true in the parity decision tree model as well. (We had mentioned this before as Conjecture 5.1.3.)

Conjecture 7.4: Partitioning the Hypercube with Affine Spaces

Let $\mathcal{T}$ be a partition of the n-dimensional Boolean hypercube into c affine subspaces. Then there is a parity decision tree with at most $2^{\mathsf{polylog}(c,n)}$ leaves such that the leaves form a refinement of $\mathcal{T}$.

In Section 3.5 we noted a few ways in which measures of f *exactly* lift to measures of $f \circ \mathsf{XOR}$. However, there is a multiplicative loss of $1/n$ when lifting from approximate sparsity to approximate rank for Boolean functions since we go through spectral norm (Theorem 3.5.6). We conjecture that this is not necessary. (We had mentioned this previously as Conjecture 3.5.7.)

Conjecture 7.5: Sparsity Lifting

Let $f : \{0,1\}^n \to \{0,1\}$. Then $\mathsf{rank}_{1/3}(f \circ \mathsf{XOR}) = \Theta(\mathsf{spar}_{1/3}(f))$.

In Chapter 5, we came up with a class of functions that are known to be hard for RPDTs, but whose compositions with XOR are not known to be hard for randomized communication protocols. We pose a conjecture regarding well-spread subspaces, that losing entropy 'with respect to each subspace' must entail a large loss of entropy. This is Conjecture 5.5.1 and we state a simplified version here.

Conjecture 7.6: A Shearer-Like Statement for Subspaces

Let $\mathcal{V} = \{V_1, \ldots, V_m\}$ be an n-dimensional (s,h)-dual subspace design[a]. Let B_i be the coset map[b] of V_i. Let X be a random variable over $\{0,1\}^n$ such that $\|B_i(X) - \mathcal{U}_{2^{\mathsf{codim}(V_i)}}\|_1 \geq \Omega(1)$ for $100h$ values of $i \in [m]$. Then $H(X) \leq n - \Omega(s)$.

[a] a way of defining well-spread, see Definition 5.3.1
[b] a natural notion of projection, see Section 5.2.1

In Chapter 6, we show various lower bounds on the quantum query complexity of first-order convex optimization. However, as shown in Figure 1.4b (reproduced here as Figure 7.1), there is a gap between the upper and lower bounds when n is small. The open problem of finding a stronger lower bound is equivalent to other open problems posed by [CCLW20, vAGGdW20]. In particular for the class of functions parametrized by $\Omega(n)$ orthonormal vectors $v_1, \ldots, v_k \in \mathbb{R}^n$ as

$$f_{v_1,\ldots,v_k}(x) = \max_{i \in [k]} \langle v_i, x \rangle,$$

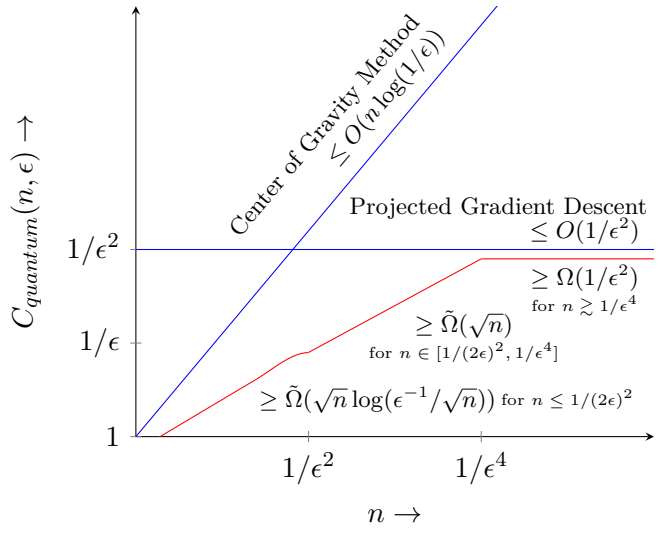


Figure 7.1: The plot shown above is with an arbitrary fixed value of ϵ. The axes are plotted in log-scale and the graph is for representational purposes. Note that $\log(\epsilon^{-1}/\sqrt{n}) = \Omega(1)$ for $n = 1/2\epsilon^2$ and $\Omega(\log(1/\epsilon))$ for $n = 1/\epsilon^{2-\Omega(1)}$.
In this thesis, we show the above-plotted lower bounds on the complexity of first-order convex optimization for quantum algorithms. In particular, we show that there is no dimension-independent quantum algorithm that can outperform the deterministic algorithm of Projected Gradient Descent.

we conjecture that a quantum query algorithm that solves the task of optimizing $f_{v_1,\ldots,v_k}$ to within an additive $0.1/\sqrt{k}$ of the optimum would require $\Omega(n)$ queries. If true, this would essentially close the gap.

Conjecture 7.7: Minimizing Max-Correlation

Solving Problem 6.1.1 on the function class stated above with $G = 1, R = 1, \epsilon = 0.1/\sqrt{k}$ requires $\Omega(n)$ queries even for quantum query algorithms.

Bibliography

[AA05] Scott Aaronson and Andris Ambainis. Quantum search of spatial regions. *Theory of Computing*, 1:47–79, 2005.

[Abe78] H. Abelson. Lower bounds on information transfer in distributed computations. In *19th Annual Symposium on Foundations of Computer Science (SFCS 1978)*, pages 151–158, 1978.

[ABT19] Anurag Anshu, Naresh Goud Boddu, and Dave Touchette. Quantum log-approximate-rank conjecture is also false. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 982–994, 2019.

[Ake78] S. B. Akers. Binary decision diagrams. *IEEE Trans. Comput.*, 27:509–516, 1978.

[Alo03] Noga Alon. Problems and results in extremal combinatorics—i. *Discrete Mathematics*, 273:31 – 53, 2003.

[AMS99] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58:137–147, 1999.

[vAG19] Joran van Apeldoorn and András Gilyén. Improvements in Quantum SDP-Solving with Applications. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, pages 99:1–99:15, 2019.

[vAGGdW17] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, oct 2017.

[vAGGdW20] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. *Quantum*, 4:220, 2020.

[AUY83] Alfred V. Aho, Jeffrey D. Ullman, and Mihalis Yannakakis. On notions of information transfer in vlsi circuits. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, page 133–139, 1983.

[Bal97] Keith Ball. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31:1–58, 1997.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26:1510–1523, 1997.

[BCWdW01] Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87, 2001.

[BdW01] Harry Buhrman and Ronald de Wolf. Communication complexity lower bounds by polynomials. In *Proceedings of the 16th Annual Conference on Computational Complexity*, page 120, 2001.

[Bel14] Aleksandrs Belovs. Quantum algorithms for learning symmetric juntas via adversary bound. In *Proceedings of the 2014 IEEE 29th Conference on Computational Complexity*, page 22–31, 2014.

[BFS86] Laszlo Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, SFCS '86, page 337–347, 1986.

[BGS75] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the p =? np question. *SIAM Journal on Computing*, 4:431–442, 1975.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74, 2002.

[BJL+19] Sébastien Bubeck, Qijia Jiang, Yin Tat Lee, Yuanzhi Li, and Aaron Sidford. Complexity of highly parallel non-smooth convex optimization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 13900–13909, 2019.

[BKL+19] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, pages 27:1–27:14, 2019.

[BMT19] Mark Bun, Nikhil S. Mande, and Justin Thaler. Sign-rank can increase under intersection. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132, pages 30:1–30:14, 2019.

[BPSW06] Paul Beame, Toniann Pitassi, Nathan Segerlind, and Avi Wigderson. A strong direct product theorem for corruption and the multiparty communication complexity of disjointness. *Comput. Complex.*, 15:391–432, 2006.

[BS83] Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22:317 – 330, 1983.

[BS90] Jehoshua Bruck and Roman Smolensky. Polynomial threshold functions, AC^0 functions and spectral norms (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 632–641, 1990.

[BS17] Fernando G.S.L. Brandão and Krysta M. Svore. Quantum speed-ups for solving semidefinite programs. In *58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, oct 2017.

[BS18] Eric Balkanski and Yaron Singer. Parallelization does not accelerate convex optimization: Adaptivity lower bounds for non-smooth convex minimization. *arXiv preprint arXiv:1808.03880*, 2018.

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8:231–357, 2015.

[CCLW20] Shouvanik Chakrabarti, Andrew M. Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. *Quantum*, 4:221, 2020.

[CG85] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, SFCS '85, page 429–442, 1985.

[CGS20] Arkadev Chattopadhyay, Ankit Garg, and Suhail Sherif. Towards stronger counterexamples to the log-approximate-rank conjecture. *CoRR*, abs/2009.02717, 2020.

[Chr13] Michael Christ. The optimal constants in Holder-Brascamp-Lieb inequalities for discrete Abelian groups. *arXiv preprint arXiv:1307.8442*, 2013.

[CKLM19] Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudo-random properties. *Computational Complexity*, 28:617–659, 2019.

[CLV19] Arkadev Chattopadhyay, Shachar Lovett, and Marc Vinyals. Equality Alone Does not Simulate Randomness. In *34th Computational Complexity Conference (CCC 2019)*, volume 137, pages 14:1–14:11, 2019.

[CM17] Arkadev Chattopadhyay and Nikhil S. Mande. Dual polynomials and communication complexity of XOR functions. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:62, 2017.

[CMS20] Arkadev Chattopadhyay, Nikhil S. Mande, and Suhail Sherif. The log-approximate-rank conjecture is false. *Journal of the ACM*, 67, 2020.

[DG19] Jelena Diakonikolas and Cristóbal Guzmán. Lower bounds for parallel and randomized convex optimization. In *Proceedings of the Thirty-Second Conference on Learning Theory*, volume 99, pages 1132–1157, 2019.

[DM18] Irit Dinur and Or Meir. Toward the KRW composition conjecture: Cubic formula lower bounds via communication complexity. *Computational Complexity*, 27:375–462, 2018.

[EH89] Andrzej Ehrenfeucht and David Haussler. Learning decision trees from random examples. *Information and Computation*, 82:231 – 246, 1989.

[EPR35] A. Einstein, B. Podolsky, and N. Rosen. Can quantum-mechanical description of physical reality be considered complete? *Physical Review*, 47:777–780, 1935.

[FFGT15] Yuri Faenza, Samuel Fiorini, Roland Grappe, and Hans Raj Tiwary. Extended formulations, nonnegative factorizations, and randomized communication protocols. *Mathematical Programming*, 153:75–94, 2015.

[FMP+15] Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald de Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM*, 62:17:1–17:23, 2015.

[For05] ForgeGod. Communication complexity - wikipedia. `https://en.wikipedia.org/w/index.php?title=Communication_complexity&oldid=30090425`, December 2005.

[Gav16] Dmitry Gavinsky. Entangled simultaneity versus classical interactivity in communication complexity. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 877–884, 2016.

[GAW19] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, page 1425–1444, 2019.

[GJPW17] Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. In *Automata, Languages, and Programming - 44th International Colloquium on Automata, languages and Programming, ICALP*, pages 52:1–52:15, 2017.

[GJW18] Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM Journal on Computing*, 47:241–269, 2018.

[GK16] Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. *Combinatorica*, 36:161–185, 2016.

[GKNS20] Ankit Garg, Robin Kothari, Praneeth Netrapalli, and Suhail Sherif. No quantum speedup over gradient descent for non-smooth convex optimization. *CoRR*, abs/2010.01801, 2020.

[GL14] Dmitry Gavinsky and Shachar Lovett. En route to the log-rank conjecture: New reductions and equivalent formulations. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 514–524, 2014.

[GLM+16] Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45:1835–1869, 2016.

[GPW15] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 1077–1088, 2015.

[GPW17] Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 132–143, 2017.

[Gro96] Vince Grolmusz. Harmonic analysis, real approximation, and the communication complexity of boolean functions. In *Computing and Combinatorics*, pages 142–151. Springer Berlin Heidelberg, 1996.

[Gro97] Vince Grolmusz. On the power of circuits with gates of low l_1 norms. *Theoretical Computer Science*, 188:117–128, 1997.

[GS08] Ben Green and Tom Sanders. Boolean functions with small spectral norm. *Geometric and Functional Analysis (GAFA)*, 18:144–162, 2008.

[GS19] Anna Gál and Ridwan Syed. Upper bounds on communication in terms of approximate rank. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:6, 2019.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.

[GW08] Andreas Griewank and Andrea Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Second Edition*. Society for Industrial and Applied Mathematics, 2008.

[GXY17] Venkatesan Guruswami, Chaoping Xing, and Chen Yuan. Subspace designs based on algebraic function fields. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80, pages 86:1–86:10, 2017.

[HHL18] Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. *SIAM Journal on Computing*, 47:208–217, 2018.

[HHL20] Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Sign rank vs discrepancy. In *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169, pages 18:1–18:14, 2020.

[Hoe63] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58:13–30, 1963.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, 1998.

[JK10] Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 247–258, 2010.

[JL01] William B. Johnson and Joram Lindenstrauss. Chapter 1 - basic concepts in the geometry of banach spaces. In W.B. Johnson and J. Lindenstrauss, editors, *Handbook of the Geometry of Banach Spaces*, volume 1, pages 1 – 84. Elsevier Science B.V., 2001.

[Joh14] Fritz John. *Extremum Problems with Inequalities as Subsidiary Conditions*, pages 197–215. Springer Basel, 2014.

[Jor05] Stephen P. Jordan. Fast quantum algorithm for numerical gradient estimation. *Physical Review Letters*, 95:050501, 2005.

[Kit97] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52:1191–1249, 1997.

[KL18] Sham M Kakade and Jason D Lee. Provably correct automatic sub-differentiation for qualified programs. In *Advances in Neural Information Processing Systems 31*, pages 7125–7135. Curran Associates, Inc., 2018.

[Kla07] Hartmut Klauck. Lower bounds for quantum communication complexity. *SIAM Journal on Computing*, 37:20–46, 2007.

[KMSY14] Gillat Kol, Shay Moran, Amir Shpilka, and Amir Yehudayoff. Approximate nonnegative rank is equivalent to the smooth rectangle bound. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 701–712, 2014.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[KP20] Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. *Physical Review A*, 101:022316, 2020.

[Kra96] Matthias Krause. Geometric arguments yield better bounds for threshold circuits and distributed computing. *Theoretical Computer Science*, 156:99–117, 1996.

[KS92] Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5:545–557, 1992.

[KW88] Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 539–550, 1988.

[LBvEB74] H.W. Lenstra, M.R. Best, and Peter van Emde Boas. A sharpened version of the aanderaa-rosenberg conjecture. *Report 30/74, Mathematisch Centrum Amsterdam (1974)*, 01 1974.

[Lee59] C. Y. Lee. Representation of switching circuits by binary-decision programs. *The Bell System Technical Journal*, 38:985–999, 1959.

[Lee12] Troy Lee. Some open problems about nonnegative rank. `http://research.cs.rutgers.edu/~troyjlee/open_problems.pdf`, 2012.

[Lin59] Carolus Linnaeus. *Systema naturae per regna tria naturae :secundum classes, ordines, genera, species, cum characteribus, differentiis, synonymis, locis.* Impensis Direct. Laurentii Salvii, 1759.

[LM00] B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, 28:1302–1338, 2000.

[LMSS07] Nati Linial, Shahar Mendelson, Gideon Schechtman, and Adi Shraibman. Complexity measures of sign matrices. *Combinatorica*, 27:439–463, 2007.

[Lov90] László Lovász. Communication complexity: A survey. In B. Korte, L. Lovász, H. Prömel, and A. Schrijver, editors, *Paths, flows, and VLSI-layout*, pages 235–265. Springer-Verlag, 1990.

[Lov14] Shachar Lovett. Recent advances on the log-rank conjecture in communication complexity. *Bulletin of the EATCS*, 112, 2014.

[Lov16] Shachar Lovett. Communication is bounded by root of rank. *Journal of the ACM*, 63:1:1–1:9, 2016.

[LR13] Troy Lee and Jérémie Roland. A strong direct product theorem for quantum query complexity. *Comput. Complex.*, 22:429–462, 2013.

[LS88] László Lovász and Michael E. Saks. Lattices, möbius functions and communication complexity. In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 81–90, 1988.

[LS09a] T. Lee and A. Shraibman. An approximation algorithm for approximation rank. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 351–357, 2009.

[LS09b] Troy Lee and Adi Shraibman. Lower bounds in communication complexity. *Foundations and Trends in Theoretical Computer Science*, 3:263–398, 2009.

[LS09c] Nati Linial and Adi Shraibman. Learning complexity vs communication complexity. *Combinatorics, Probability and Computing*, 18:227–245, 2009.

[LS09d] Nati Linial and Adi Shraibman. Lower bounds in communication complexity based on factorization norms. *Random Structures & Algorithms*, 34:368–394, 2009.

[LWY20] Kasper Green Larsen, Omri Weinstein, and Huacheng Yu. Crossing the logarithmic barrier for dynamic boolean data structure lower bounds. *SIAM Journal on Computing*, 49, 2020.

[MNSW98] Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. *Journal of Computer and System Sciences*, 57:37–49, 1998.

[MS82] Kurt Mehlhorn and Erik M. Schmidt. Las vegas is better than determinism in vlsi and distributed computing (extended abstract). In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, page 330–337, 1982.

[NC16] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information (10th Anniversary edition)*. Cambridge University Press, 2016.

[Nem94] A. Nemirovski. On parallel complexity of nonsmooth convex optimization. *Journal of Complexity*, 10:451 – 463, 1994.

[Nes04] Yurii Nesterov. *Introductory Lectures on Convex Optimization*. Springer US, 2004.

[Nes18] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[New91] Ilan Newman. Private vs. common random bits in communication complexity. *Inf. Process. Lett.*, 39:67–71, 1991.

[NS92] Noam Nisan and Mario Szegedy. On the degree of boolean functions as real polynomials. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, page 462–467, 1992.

[NY83] Arkadiĭ Nemirovsky and David Borisovich Yudin. Problem complexity and method efficiency in optimization. *Wiley Series in Discrete Mathematics and Optimization*, 1983.

[O'D14] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[OWZ+14] Ryan O'Donnell, John Wright, Yu Zhao, Xiaorui Sun, and Li-Yang Tan. A composition theorem for parity kill number. In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 144–154, 2014.

[PS86] Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33:106 – 123, 1986.

[Raz92] Alexander A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106:385–390, 1992.

[Raz03] Alexander A. Razborov. Quantum communication complexity of symmetric predicates. *Izvestia: Mathematics*, 67:145–159, 2003.

[Rei11] Ben Reichardt. Reflections for quantum query algorithms. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 560–569, 2011.

[RM97] R. Raz and P. McKenzie. Separation of the monotone nc hierarchy. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, FOCS '97, page 234, 1997.

[RSW+19] Patrick Rebentrost, Maria Schuld, Leonard Wossnig, Francesco Petruccione, and Seth Lloyd. Quantum gradient descent and Newton's method for constrained polynomial optimization. *New Journal of Physics*, 21:073023, 2019.

[RT19] Ran Raz and Avishay Tal. Oracle separation of BQP and PH. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 13–23. ACM, 2019.

[SdW19] Makrand Sinha and Ronald de Wolf. Exponential separation between quantum communication and logarithm of approximate rank. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 966–981, 2019.

[Spi71] Philip M. Spira. On time hardware complexity tradeoÆs for boolean functions. *Proceedings of the Fourth Hawaii International Symposium on System Sciences*, pages 525–527, 1971.

[STV17] Amir Shpilka, Avishay Tal, and Ben Lee Volk. On the structure of boolean functions with small spectral norm. *Computational Complexity*, 26:229–273, 2017.

[TWXZ13] Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier sparsity, spectral norm, and the log-rank conjecture. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 658–667, 2013.

[Ver99] Nikolai K. Vereshchagin. Relativizability in complexity theory. In Lev Beklemishev, Mati Pentus, Nikolai Vereshchagin, and M. Pentus, editors, *Provability, Complexity, Grammars*, 1999.

[Weg00] Ingo Wegener. *Branching Programs and Binary Decision Diagrams*. Society for Industrial and Applied Mathematics, 2000.

[Wol19] Ronald de Wolf. Quantum computing: Lecture notes. *CoRR*, abs/1907.09415, 2019.

[WS17] Blake Woodworth and Nathan Srebro. Lower bound for randomized first order convex optimization. *arXiv preprint arXiv:1709.03594*, 2017.

[Yan91] Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43:441–466, 1991.

[Yao79] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the 11h Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1979, Atlanta, Georgia, USA*, pages 209–213, 1979.

[Yao83] Andrew C. Yao. Lower bounds by probabilistic arguments. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, page 420–428, 1983.

[Zha14] Shengyu Zhang. Efficient quantum protocols for XOR functions. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1878–1885, 2014.