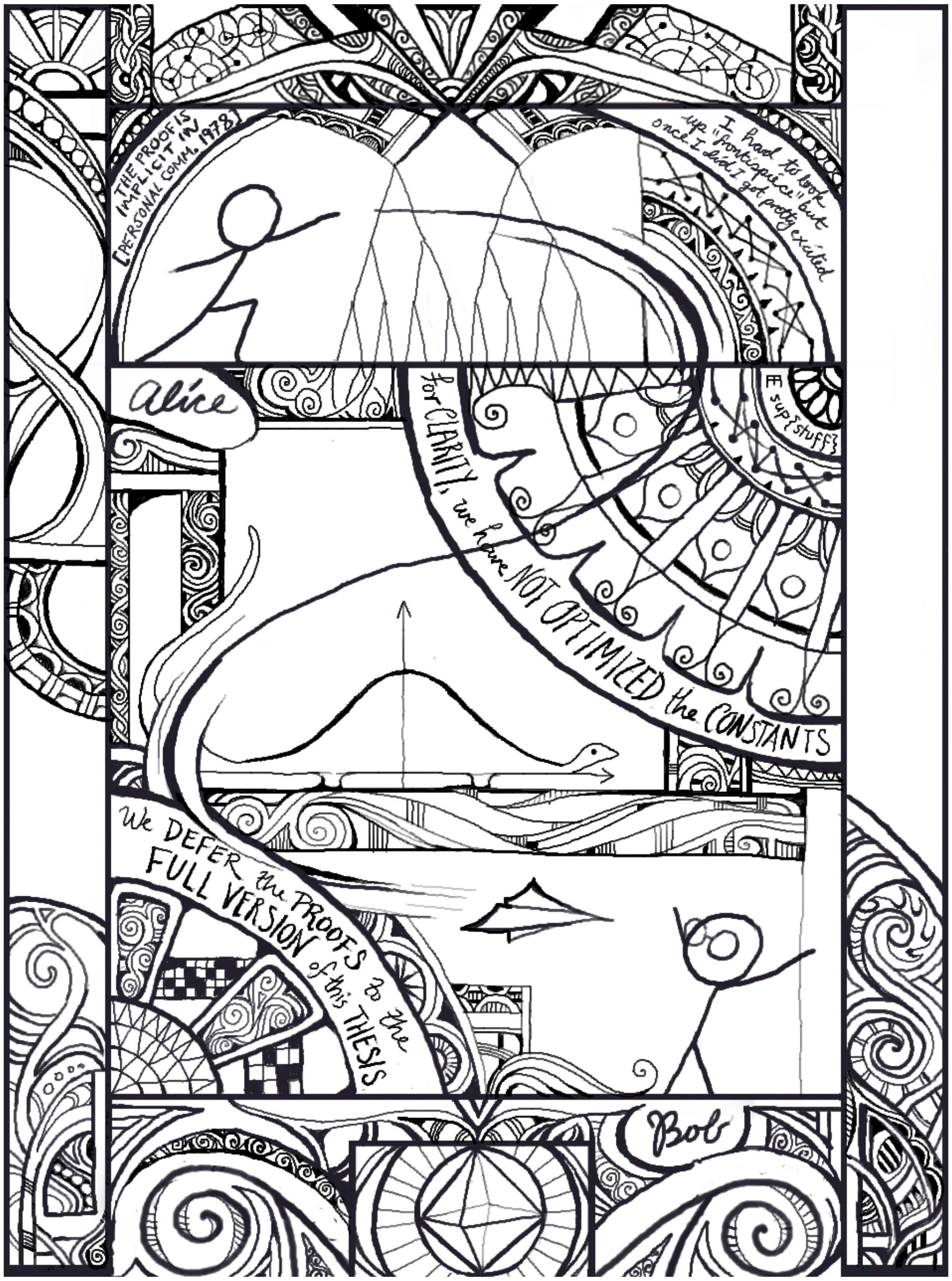
Any errors in this dissertation are probably fixable: topics in probability and error correcting codes

by

Mary Katherine Wootters

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mathematics)
in The University of Michigan
2014


THE PROOF IS IMPLICIT IN [PERSONAL COMM. 1978]
I had to look up "frontispiece" but once I did, I got pretty excited
Alice
for CLARITY, we have NOT OPTIMIZED the CONSTANTS
sup{stuff}
We DEFER the PROOFS to the FULL VERSION of this THESIS
Bob

For Isaac

ACKNOWLEDGEMENTS

I have so many people to thank for this thesis. First, I cannot express enough gratitude to Martin Strauss and Anna Gilbert. Martin and Anna have been the best (official and unofficial) advisors I could have hoped for in graduate school. They have gone far out of their way to make sure that I had every opportunity: to learn, travel, speak at conferences, do internships, and so on. Their support went far beyond academic. I don't think many other people can say that their advisor is an awesome running coach, or did a 5k swim race with them, or advises them on the logistics of mixing conference travel with cycling tours. Because of Martin and Anna, grad school was a lot of fun, academically and otherwise.

In addition to my advisor, I was fortunate to receive wisdom from several advisorly figures. In particular, I thank Brett Hemenway, Yaniv Plan, and Atri Rudra, who have taught me so much about how to do research. I also would like to thank all of my other coauthors and collaborators throughout grad school: Rich Baraniuk, Mark Davenport, Moritz Hardt, Simon Foucart, Carl Miller, Deanna Needell, Jelani Nelson, Hung Ngo, Rafi Ostrovsky, Eric Price, Yaoyun Shi, Ewout van den Berg, and David Woodruff. Finally, I thank the other members of my committee, Mark Rudelson, Roman Vershynin, and Grant Schoenebeck, for their helpful feedback on this dissertation and throughout graduate school. I especially thank Roman for the two excellent courses I took from him, and both Mark and Roman for organizing such great seminars while I've been here.

I also thank all of the institutions which have supported and housed me for the past five years, in particular the Math and EECS departments at UMich—especially Tara, Stephanie, Carrie, and Lauri, without whom I would likely be living on the street, locked out of my office, and without health insurance. I gratefully acknowledge a Rackham predoctoral fellowship for funding in my last year of graduate school. I thank the Simons Institute for Theoretical Computer Science for their hospitality and support during the Fall 2013 semester, and I thank the theory group at IBM Almaden for a wonderful internship during Summer 2011. Finally I thank Mighty Good Coffee and all three Espresso Royales in downtown Ann Arbor for their continued hospitality and caffeination.

I thank all of my friends and family. Thanks Brittan, for putting up with me as an office-mate. Thanks to Mom, Dad, and Nate for your support while I delay adulthood. And, thank you to Isaac, my best friend and the love of my life, for everything.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

The protagonists of this dissertation are Alice and Bob. Alice wants to send a message to Bob, in the face of seemingly insurmountable obstacles. While the full backstory of Alice and Bob and their personal lives is beyond the scope of this dissertation, there are a few common reasons to study such a scenario. When Alice and Bob are replaced with the more pragmatic but less evocotive "sender" and "receiver," there are immediate applications to communication; beyond that, this situation is relevant to storage, cryptography, complexity theory, and pseudorandomness, among others.

In the theory of *error correcting codes,* one studies variations of the above scenario, and hopes to provide Alice and Bob with the tools to succeed. In a standard set-up, Alice begins with a message x of length k, which she maps to a *codeword* $c = \mathcal{C}(x)$ of length n; she then sends this codeword to Bob. Unfortunately, the codeword may be corrupted *en route.* Bob's job is to take this corrupted codeword, and to determine Alice's original message x.

We will use tools from probability theory—mostly tools from high-dimensional probability—to study coding theory. The motivating question is: what should Alice and Bob do? On the combinatorial side of things, how should they pick the set $\mathcal{C}$ of all possible codewords? We call this set an *error correcting code.* On the algorithmic side of things, how can Alice efficiently encode x into a codeword $c = \mathcal{C}(x) \in \mathcal{C}$? How can Bob efficiently recover the message that Alice sent? In general, there is a trade-off between the effectiveness of their communication (as measured by robustness to noise, efficiency of encoding/decoding, and so on), and the amount of redundancy Alice and Bob must use.

In the settings we consider, the most successful approaches to these questions have been algebraic in nature: for the most part, they rely on properties of polynomials over finite fields. By approaching these problems from an analytic and probabilistic point of view, we improve the trade-off for Alice and Bob.

We consider two variants of coding theory, *list decoding* and *local decoding.* In each of these variants, Bob is not required to recover everything about x; but in return, he faces a more difficult task. In list decoding, Bob need not recover Alice's message x exactly, and instead may recover a short list which contains x; but the number of corruptions may be very large. In local decoding, Bob is only trying to recover a small portion of Alice's message, say a single bit of x; but he must manage this extremely quickly, without even looking at the entire codeword c. This dissertation answers two long-standing open questions in list decoding, and provide a new answer to an open-until-very-recently question in local decoding.

1.1 Overview of contributions

Before diving into the details, we outline our main contributions.

1.1.1 List decoding

We focus first on *list decoding.* List decoding was first introduced by Elias [27] and Wozencraft [112] in the 1950's, and has received a great deal of attention from both coding and complexity theorists over the past few decades. In list decoding, Bob need not recover Alice's message x uniquely, but instead may recover a short list of possible messages which includes x. One important reason to study list decoding is that Alice and Bob can handle much more error in this setting than in the standard setting. We focus on the challenge of designing codes $\mathcal{C}$ which allow for communication even in the presence of extreme noise. Given an amount of noise, we aim to minimize the amount of redundancy that Alice and Bob must use. We measure the redundancy of the code $\mathcal{C}$ by the *rate* of $\mathcal{C}$: if Alice wishes to send a message of length k, and actually ends up sending a codeword of length n, the rate is defined to be the ratio k/n. Thus, for a given amount of noise, we seek to maximize the rate of the code.

The state of the list decoding literature is very interesting. Generally speaking, we have three ways of obtaining (guarantees about) list-decodable codes.

1. The first tool is a classical result called the *Johnson bound,* which is a combinatorial statement. The Johnson bound gives guarantees about the list-decodability of a code given its *distance* (a combinatorial property of the code to which we will return later). However, while the Johnson bound is the strongest statement possible using distance alone, there are codes which beat the guarantees of the Johnson bound.

2. A second approach comes from random codes. As we will see, a completely random subset $\mathcal{C} \subset \mathbb{F}_q^n$ is, with high probability, optimal for the list decoding problem. In particular, the guarantees for such a code go beyond the Johnson bound, and meet the information-theoretic limit for this problem. This is well known and in some sense not very interesting; in coding theory, it is often the case that a completely random code attains near-optimal combinatorial bounds. A more interesting direction is *structured* ensembles of random codes. For ex-

ample, we may consider a code selected uniformly at random from a collection of "nice" codes. Although a few pathological cases may make it impossible to say *"all "nice" codes have good list-decodability properties,"* perhaps it is still possible to say *"most "nice" codes have good list-decodability properties."* This has turned out to be surprisingly difficult. A natural starting point is *random linear codes*; that is, codes $\mathcal{C}$ which are a random linear subspace of $\mathbb{F}_q^n$. This simple case—which is much less random than a general random code—is already interesting (and nontrivial). It was asked by Elias [28] in 1991 whether random linear codes were as list-decodable as general random codes, and to date there has been a great deal of work on this [20, 44, 45, 49, 95]. Other possible "nice" families include certain ensembles of *Reed-Solomon codes,* which are a well-studied family of codes based on polynomials over finite fields.

3. In the past two decades, there has been a great deal of interest in explicit constructions of list-decodable codes, especially those which admit efficient encoding and decoding algorithms. This literature began in the late 1990's, with the work of Guruswami and Sudan [57,101], who showed how to efficiently list-decode Reed-Solomon codes up to the Johnson bound. Their work sparked a search for efficiently encodable and decodable codes which are list-decodable *beyond* the Johnson bound [23,48,51,62,63,78,87], and also a line of work trying to establish whether Reed-Solomon codes themselves might do the trick [12,50,94].

One interesting feature of the landscape sketched above is that, other than general random codes, the only optimally list-decodable codes we know about are highly structured—that is, they fall under Category 3. We start our investigation in this dissertation in Categories 1 and 2; surprisingly, our approach will also make some progress on Category 3.

Contributions in list decoding.

This dissertation makes the following contributions in list decoding.

- **List decodability of random linear codes.** We show that random linear codes, over constant-sized alphabets, are optimally list-decodable. This answers a question, asked by Elias [28], which had been open for over 20 years at the time of this writing. As an added benefit, our proof is quite simple. For large, non-constant alphabets, we can show (using a more complex argument) that random linear codes are *nearly* optimally list-decodable (up to logarithmic factors in the rate).

- **List decodability of Reed-Solomon codes.** We show that there do exist Reed-Solomon codes which are list-decodable beyond the Johnson bound. This answers a question first asked by Guruswami and Sudan over 15 years ago [56]—see [43, 94, 108] for explicit formulations of this problem. To the best of our knowledge, it was not known which way this question would go, and in fact there has been significant effort devoted to showing that such codes do not exist [12, 19, 50].

- **General statements about random families of codes.** In fact, the earlier two bullet points are corollaries of two very general theorems (one for small alphabets and one for large alphabets), which provides a way to obtain (nearly) optimally list-decodable codes from any code with good structural properties. This yields general statements which fall somewhere between Categories 1 and 2 above. For example, it is not true that any code with good distance is optimally list-decodable—the Johnson bound is tight in this respect—but we can show that "most" (suitable transformations of) codes with good distance *are* optimally list decodable.

- **A few more applications.** While random linear codes and Reed-Solomon codes are the headline applications of the machinery mentioned above, we show how it can be used to obtain other useful constructions. Examples include linear-time encodable, optimally list-decodable, binary codes; optimally list-decodable variants on Reed-Muller codes; and results about the list-decodability of randomly folded codes. Along the way, we also prove several "average-radius" variants of the Johnson bound, which appear to be folklore but are probably worth having written down.

1.1.2 Local decoding

In the second part of this thesis, we will consider *locally decodable codes.* In the local decoding setup, again Bob's job is again easier: he need only recover a single bit of Alice's message. The catch is that Alice doesn't know before she encodes her message which bit Bob will be interested in. Further, we insist that Bob work in *sublinear time.* In particular, he doesn't have time to look at the entire codeword c; his decoding is "local" in the sense that he needs to look at only a few bits of the codeword. Locally decodable codes have been lurking implicitly in coding theory [89] since the 1950's and in theoretical computer science [5, 16, 33, 34, 36, 82, 88] since the late 1980's, but the first explicit definition did not appear until later [75]. The reader is referred to [114] for an excellent survey.

The important trade-off in this setting is between the *locality* of the code—how many bits Bob must look at to recover a single bit of Alice's message—and the rate of the code. Generally speaking, there are two parameter regimes, in which very different approaches have been considered.

1. **Small locality, small rate.** In the first parameter regime, the locality is very small: Bob may look at only two or three bits. However, the rate is quite bad: Alice must send nearly 2^k bits to convey a message of length k. Approaches in this regime tend to be combinatorial [25, 26, 113]. At a high level, the arguments follow the same outline: show that Bob can succeed if there are *no* errors, and then argue that with enough randomization his queries will, with high probability, avoid any errors that do occur.

2. **Large locality, high rate.** In the second parameter regime, the rate of the code approaches 1 (so n is only very slightly larger than k), but the locality grows with n, perhaps like $n^{0.001}$. Codes in this regime have only been found very recently. Other than the work presented in this thesis, there are currently two families of codes known in this parameter regime, *multiplicity codes* [79] and *lifted codes* [41]. The arguments here are quite different that those in the small-locality regime. First, they are algebraic, rather than combinatorial. Second, they follow a different outline: when Bob considers $\omega(1)$ bits, it is no longer enough for him to succeed in the error-free setting. Indeed, he looks at so much of the codeword that there is no way he can avoid errors entirely, no matter how cleverly he randomizes.

Contributions in local decoding

In this dissertation, we make the following contributions in local decoding.

- **Locally decodable codes in the high-rate regime.** We will give the third known family in the large-locality, high-rate regime. Our codes will actually be locally *correctible,* which is a slightly stronger notion. As mentioned above, there are only two other constructions of such codes known. In fact, it was a conjecture of Dvir that such codes did not exist [21]. One of the main contributions of this work is that our construction is quite different from multiplicity codes and lifted codes—in fact, our style of argument follows the probabilistic and combinatorial arguments from the small-error, small-rate regime. Our work can be viewed as a way to port the small-locality line of reasoning to the high-rate setting.

- **Sublinear-time decoding of expander codes.** The construction mentioned above is in fact not new: we use a family of codes (called *Tanner codes* or *expander codes*) which have been around in some form or another since the 1980's, with roots going back to the 1960's. Thus, our results also give sublinear-time decoding algorithms for this well-studied family of codes. As far as we can tell, it was not suspected that these codes might provide the sought-after locality.

1.2 Dissertation outline

In Chapter 2, we will set up the formal notation and prove some simple lemmas that we will need. We will also prove a couple of (standard) theorems and work out a few computations, to set the stage for our results later. In Chapter 3 we present some results about the list-decodability of certain ensembles of binary codes; as a corollary, we will answer a question of [28], and show that random linear codes are (with high probability) as list-decodable as random codes. The results in Chapter 3 are based on the paper [111]. In Chapter 4, we will extend the arguments of Chapter 3 to deal with larger alphabet sizes. This will involve a fair amount of work and is the most technical part of the dissertation. As a corollary, we will show that there exist *Reed-Solomon* codes which are nearly optimally list-decodable; this answers a question posed by Guruswami and Sudan in [56]. The results in Chapter 4 are based on the paper [96], which is joint work with Atri Rudra. Given that the punchlines of Chapters 3 and 4 are corollaries of more general phenomena, it is natural to ask how far you can push this technique; Chapter 5 explores this question. In Chapter 5 we state a very general theorem about random operations of codes, and give recipes for obtaining optimally list-decodable codes. The results in Chapter 5 are joint work with Atri Rudra. Finally, in Chapter 6 we mix it up a bit and turn our attention to locally decodable codes, and we show that expander codes are locally decodabable in the high-rate regime. The results in Chapter 6 are based on the paper [68], which is joint work with Brett Hemenway and Rafail Ostrovsky.

CHAPTER 2

Set up and Preliminaries

2.1 Basic coding theory: background and definitions

We return to Alice and Bob. Formally, Alice and Bob employ an *error correcting code,* which is a subset $\mathcal{C} \subset \mathbb{F}_q^n$, for a finite field[1] $\mathbb{F}_q$. The size q of the field is called the *alphabet size.* The elements $c \in \mathcal{C}$ are called *codewords.* The codewords then have length n, which is called the *block length* of $\mathcal{C}$. For every *message* $x \in \mathbb{F}_q^k$ of length k, there is an *encoding function* which maps x to $c = \mathcal{C}(x) \in \mathcal{C}$. As we just did, we will occasionally overload notation and write $\mathcal{C} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ for a function whose image is the set $\mathcal{C} \subset \mathbb{F}_q^n$. The *size* of the code is then $|\mathcal{C}| = q^k$. In the Alice-and-Bob scenario above, Alice will choose a codeword to send to Bob; if Bob can correctly identify the codeword, he will have identified the message Alice wishes to send. The general setup is shown in Figure 2.1.

Alphabet size and error model. The alphabet size q and the way in which errors may be introduced are important and linked parameters. In our error model, a *symbol* $c_i \in \mathbb{F}_q$ of a codeword $c \in \mathbb{F}_q^n$, is the smallest unit of communication that can be corrupted. Here, *corrupted* means that the symbol may be changed to another element of $\mathbb{F}_q$. There are many reasonable models of corruption. For

[1]While it is not in general necessary to assume that the alphabet has any sort of algebraic structure, for this thesis it will be convenient to consider codes over finite fields.

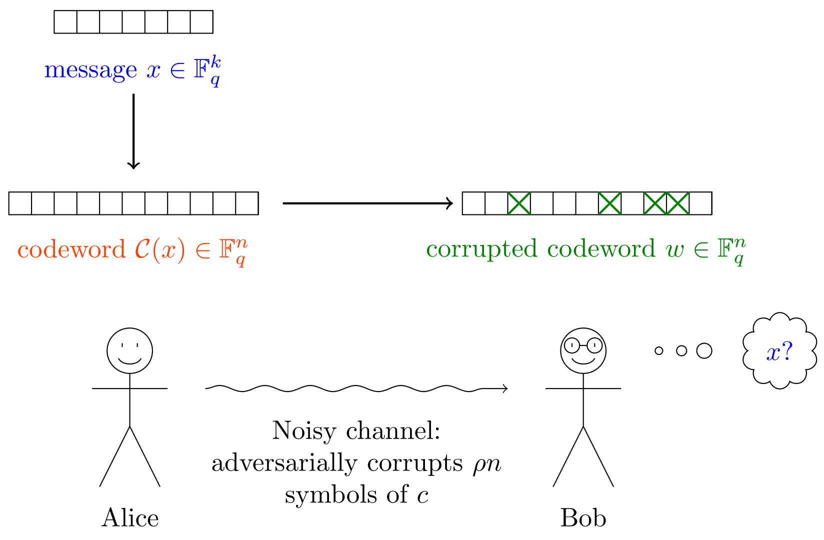

message $x \in \mathbb{F}_q^k$
codeword $\mathcal{C}(x) \in \mathbb{F}_q^n$
corrupted codeword $w \in \mathbb{F}_q^n$
x?
Noisy channel: adversarially corrupts ρn symbols of c
Alice
Bob

Figure 2.1: The set-up for error correcting codes: Alice-and-Bob version.

example, symbols could be changed (or not) independently at random; or only certain error patterns could be possible. In this work, we exclusively consider the most conservative, worst-case model. That is, up to ρn symbols may be corrupted, for some parameter $\rho \in [0, 1]$, and these corruptions may occur in any locations. When a symbol is corrupted, it can be changed to any other symbol in $\mathbb{F}_q$. We imagine that these corruptions are adversarial: someone who knows Alice and Bob's strategy is deliberately trying to mess them up.[2] The fraction ρ of errors that this adversary is allowed to introduce is called the *error rate.*

The study of error correcting codes was initiated in the seminal paper of Shannon [97]. Shannon considered a probabilistic error model; the adversarial model that we study here was indroduced by Hamming [66], and is often referred to as the "Hamming model."

Distance. In the Hamming model, Alice and Bob's success (combinatorially

[2]Again, who this bad guy is and why he's so out to get Alice and Bob is beyond the scope of this dissertation; suffice it to say that this worst-case model is not only nice and conservative in the communication setting, but it also turns out to be essential for applications in complexity theory and other areas.

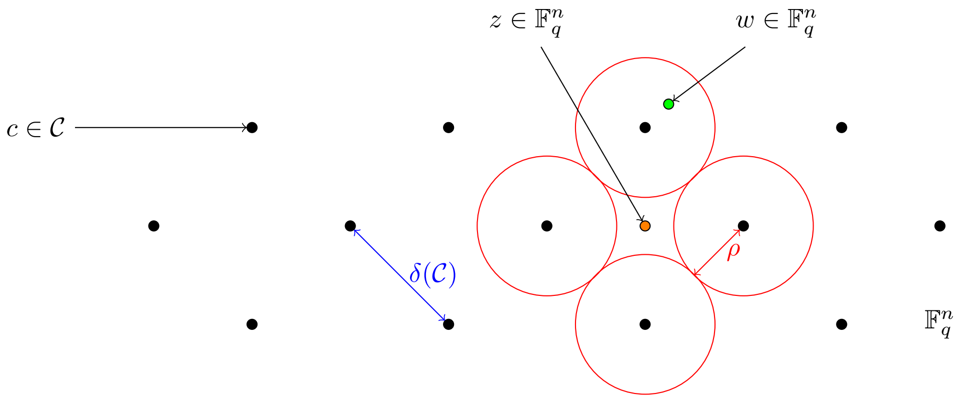


Figure 2.2: The set-up for error correcting codes: Combinatorial version. The black dots represent the elements of the code $\mathcal{C}$, with distance $\delta(\mathcal{C})$. If $w \in \mathbb{F}_q^n$ differs from a codeword $c \in \mathcal{C}$ in at most ρn places, and $\rho \leq \delta/2$, it is possible to uniquely determine c from w. On the other hand, if $z \in \mathbb{F}_q^n$ differs from $c \in \mathcal{C}$ in more than $\delta/2$ places, it may not be possible to determine c from z.

speaking) is determined by the *distance* of the code:

$$\delta(\mathcal{C}) := \min_{c,c' \in \mathcal{C}} \delta(c, c'),$$

where $\delta(c, c')$ is the *relative Hamming distance* between c and c':

$$\delta(c, c') := \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{c_i \neq c'_i}.$$

If the distance $\delta(\mathcal{C})$ is larger than 2ρ, then for any $w \in \mathbb{F}_q^n$ Bob may receive, there is at most one $c \in \mathcal{C}$ so that $\delta(c, w) < \rho$. Thus, no matter what errors are introduced, Bob can uniquely determine which codeword c was sent. On the other hand, if the distance is smaller than 2ρ, there is always an error pattern which will trip up Alice and Bob. Thus, in the Hamming model, the distance of the code characterizes the acceptable (to Alice and Bob) error rates ρ. This combinatorial view of error correcting codes is shown in Figure 2.2.

Rate. Another important quantity which measures the effectiveness of Alice and Bob's communication is the ratio of the length k of Alice's message (the number of symbols she wants to send) to the length n of the codeword (the number of symbols she actually sends). This quantity $R = k/n$ is called the *rate* of the code $\mathcal{C}$. Since there are q^k possible messages of length k, the code $\mathcal{C}$ has size q^k; thus, the rate is given by

$$R = \frac{\log_q(|\mathcal{C}|)}{n}.$$

Note that the rate is always between 0 and 1.

Families and ensembles of codes. We consider Alice and Bob's situation as n becomes very large. To that end, we will consider *families of codes.* A family $\mathcal{C} = \mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \ldots$ is a sequence of codes, so that the length of $\mathcal{C}_i$ is n_i, and $n_i \nearrow \infty$. We can define distance and rate for families as we did with codes: if the rate of $\mathcal{C}_i$ is R_i and the distance is δ_i, the rate R and distance δ of $\mathcal{C}$ are given by

$$R = \liminf_i R_i \qquad \text{and} \qquad \delta = \liminf_i \delta_i.$$

Above, we have used the same notation ($\mathcal{C}$) for a family of codes as we used for a particular code; this will be the first in a long line of notational abuses on this topic. In particular, we will henceforth refer to a family of codes as simply a "code," and we will refer to its rate and distance in terms of the length n of the code. We will also invoke standard asymptotic notation (e.g., $R = 1 - o(1)$ to indicate that the rate approaches 1 as n tends to infinity) to describe the behavior of codes as n becomes large. For reference, we define this notation in Section 2.5.

We also consider (families of) *random* codes. That is, we fix a distribution $\mathcal{D}$ on subsets $\mathcal{C}$ of $\mathbb{F}_q^n$, and we imagine that $\mathcal{C}$ is a code drawn from $\mathcal{D}$. In this case, we may be interested in, say, bounding the rate and distance with high probability. As before, we will be interested in the case that n gets large, and we will have in mind a sequence of distributions $\mathcal{D}_1, \mathcal{D}_2, \ldots$, so that $\mathcal{D}_i$ is a distribution on subsets of $\mathbb{F}_q^{n_i}$, for an infinite family of increasing n_i. We will sometimes call such a family of random codes an *ensemble* of codes.

2.1.1 The rate-distance trade-off: some basic bounds

The fundamental problem in coding theory is understanding the trade-off between distance and rate. The larger the distance, the larger the error rate can be. The larger the rate, the more information Alice can send to Bob. This trade-off has been studied since the beginning of time—that is, since around 1950—beginning with the work of Hamming [66]. Since then, the literature has grown far too much to be completely surveyed here. See [54, 84] for an introduction to coding theory. In order to get a feel for the types of rate-distance trade-offs we can hope for, we mention a few classical results below.

We start with the *Singleton bound,* which states that any code $\mathcal{C} \in \mathbb{F}_q^n$ with distance $\delta = \delta(\mathcal{C})$ must have rate at most

$$R \leq 1 - \delta + 1/n. \tag{2.1}$$

To see this, consider the projection of $\mathcal{C}$ onto the first $(1-\delta)n+1$ coordinates of $\mathbb{F}_q^n$. This projection is injective, because by definition, no two codewords agree in more than $(1-\delta)n$ symbols. Thus, we have $|\mathcal{C}| \leq q^{(1-\delta)n+1}$, which implies the bound.

Sphere-packing arguments can also be used to get a handle on how large (or small) a code $\mathcal{C}$ with distance δ can (or must) be. Recall that we are working over an alphabet of size q, and for $z \in \mathbb{F}_q^n$, define the (q-ary) *Hamming ball* of radius ρ about z, denoted $B_q(z,\rho) \subset \mathbb{F}_q^n$, by

$$B_q(z,\rho) = \left\{ x \in \mathbb{F}_q^n \,:\, \delta(z,x) \leq \rho \right\}.$$

Suppose that $\mathcal{C}$ is a maximal code with distance δ. How big can $\mathcal{C}$ be? The balls

$B_q(c, \delta/2)$ for $c \in \mathcal{C}$ are all disjoint, and all contained in $\mathbb{F}_q^n$, so we must have [3]

$$|\mathbb{F}_q^n| \geq \sum_{c \in \mathcal{C}} |B_q(c, \delta/2)| = |\mathcal{C}||B_q(0, \delta/2)|.$$

On the other hand, since $\mathcal{C}$ is maximal, $\mathbb{F}_q^n$ is covered by the union of $B_q(c, \delta)$ for $c \in \mathcal{C}$. Thus,

$$|\mathbb{F}_q^n| \leq \left| \bigcup_{c \in \mathcal{C}} B_q(c, \delta) \right| \leq \sum_{c \in \mathcal{C}} |B_q(c, \delta)| = |\mathcal{C}||B_q(0, \delta)|.$$

Putting these together, we conclude that

$$\frac{q^n}{|B_q(0, \delta)|} \leq |\mathcal{C}| \leq \frac{q^n}{|B_q(0, \delta/2)|}. \tag{2.2}$$

The lower bound on $\mathcal{C}$ is known as the *Gilbert-Varshamov (GV) bound.* The upper bound is called the *Hamming bound.* To understand these bounds, it is helpful to get an idea of the size of the *Hamming ball* $B_q(0, \delta)$. We can write

$$|B_q(0, \delta)| = \sum_{j=1}^{\lfloor n\delta \rfloor} \binom{n}{j} (q-1)^j,$$

although perhaps that's not very illuminating. One way to get good intuition for $|B_q(0, \delta)|$ is through the *q-ary entropy function* H_q. We define

$$H_q(x) = x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x). \tag{2.3}$$

The q-ary entropy function is a generalization of the standard (binary) entropy. It is plotted in Figure 2.3, and we will return to its behavior in much more detail below. It turns out that $H_q(\delta)$ nicely characterizes the size of Hamming balls over $\mathbb{F}_q$. Indeed, (see [84] for the computation), for any $\delta \in (0, 1 - 1/q)$ and for sufficiently large n,

$$q^{n(H_q(\delta) - o(1))} \leq |B_q(0, \delta)| \leq q^{nH_q(\delta)}. \tag{2.4}$$

[3] Notice that for any c, $|B_q(c, \delta/2)| = |B_q(0, \delta/2)|$, which follows from the fact that the map $x \mapsto x - c$ is an automorphism of $\mathbb{F}_q^n$ which preserves Hamming distance.

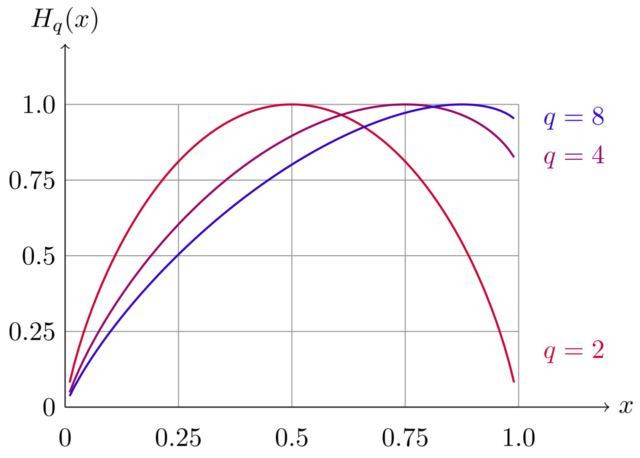


Figure 2.3: The q-ary entropy function $H_q(x)$.

Combining (2.2) and the above, we see that the the rate $R = \log_q(|\mathcal{C}|)/n$ of a maximal code $\mathcal{C}$ of distance δ is bounded by

$$1 - H_q(\delta) - o(1) \leq R \leq 1 - H_q(\delta/2). \tag{2.5}$$

Some useful facts about $H_q(x)$. To understand Equation (2.5), we expand a bit on the function $H_q(\delta)$. Shown in Figure 2.3, the function $H_q(x)$ attains its maximum (which is 1) at $1 - 1/q$. It will be useful to investigate its behavior near this point. Very near to $1 - 1/q$, say at $1 - 1/q - \varepsilon$ for $\varepsilon \ll 1/q$, the behavior of $H_q(x)$ is roughly quadratic in ε; for slightly larger ε, it is roughly linear. To be more precise, consider the series expansion of $H_q(1 - 1/q - \varepsilon)$ near $\varepsilon = 0$. We have

$$H_q(1 - 1/q - \varepsilon) = 1 - \left(\frac{q^2}{2(q-1)\log(q)}\right)\varepsilon^2 + O_q(\varepsilon^3), \tag{2.6}$$

which describes the behavior of H_q for constant q and $\varepsilon \to 0$. On the other hand, expanding $H_q(1 - 1/q - \varepsilon)$ near $q = \infty$, we have

$$H_q(1 - 1/q - \varepsilon) = 1 - 1/q - \varepsilon - \frac{H_2(\varepsilon)}{\log_2(q)} - o_\varepsilon(1/q) = 1 - \varepsilon + O_\varepsilon(1/\log(q)), \tag{2.7}$$

which describes the behavior of H_q for constant ε and growing q. In chapters 4 and 5, we will be interested in situations where both $\varepsilon \to 0$ and $q \to \infty$ at the same time. In this case, the asymptotics of $H_q(1 - 1/q - \varepsilon)$ can get a little hairy, but the following is true:

(2.8)
$$H_q(1-1/q-\varepsilon) = \begin{cases} 1 - \Theta\left(\frac{q\varepsilon^2}{\log(q)}\right) & q = O(1/\varepsilon) \text{ (``small'' } q\text{)} \\ 1 - \Theta(\varepsilon) & q = \Omega(\varepsilon^{-c}) \text{ for some constant } c > 1 \text{ (``large'' } q\text{)} \\ 1 - \varepsilon(1 - o(1)) & q = \varepsilon^{-\omega(1)} \text{ (``very large'' } q\text{)} \end{cases}$$

We will informally use the labels “small q” and “large q” to refer to the parameter regimes above. These regimes do not capture everything—we may have $q = \log(1/\varepsilon)/\varepsilon$, for example—but they are enough intuition for our purposes.

Let us consider the take-away from these calculations, combined with (2.5), which said that the rate R of the largest code of distance δ obeys

$$1 - H_q(\delta) - o(1) \le R \le 1 - H_q(\delta/2).$$

The first inequality (the Gilbert-Varshamov bound) then implies that there are codes with distance $\delta = 1 - 1/q - \varepsilon$ and rate approaching ε^2 (small q) or ε (large q). On the other hand, the second inequality (the Hamming bound) indicates that perhaps we can do better. For large q, it is known that in fact we can. For binary codes, where $q = 2$, it is not known whether one can do better than the Gilbert-Varshamov bound.

Pinning down the best possible rate-distance trade-off is still an open question, and there are tighter bounds than those presented here. However, the bounds given above will be enough intuition for the work in this thesis. In our work, we will study several classical ensembles of codes, whose rates and distances are well-understood (and generally speaking match the Gilbert-Varshamov bound), and prove new results about their capabilities. We introduce some of these families in the next section.

2.1.2 Examples of codes

Most of the codes in this dissertation are *linear codes,* which are codes $\mathcal{C}$ so that $\mathcal{C}$ forms a linear subspace over $\mathbb{F}_q$. In this case, the message length k is the dimension of the subspace, and this is called the *dimension* of the code; as before the rate is $R = k/n$. We may write a linear code $\mathcal{C}$ as

$$\mathcal{C} = \left\{ x^T G \,:\, x \in \mathbb{F}_q^k \right\},$$

where $G \in \mathbb{F}_q^{k \times n}$ is a matrix[4] of rank k over $\mathbb{F}_q$. We refer to G as a *generator matrix* for $\mathcal{C}$. We may also write a linear code $\mathcal{C}$ as

$$\mathcal{C} = \left\{ y \in \mathbb{F}_q^n \,:\, Hy = 0 \right\},$$

for some matrix[5] $H \in \mathbb{F}_q^{(n-k)\times n}$ of rank $n - k$. We refer to H as a *parity check matrix* for $\mathcal{C}$. Notice that if G is a generator matrix for $\mathcal{C}$ and H is a parity-check matrix for $\mathcal{C}$, then $HG^T = 0$: H spans the kernel of G.

In Chapters 3 and 4, we study (uniformly) *random linear codes.* A uniformly random linear code $\mathcal{C}$ of rate R is just a uniformly random linear subspace of $\mathbb{F}_q^n$ of dimension $k = Rn$. It will be convenient to also consider the distribution on linear codes that arises from choosing a generator matrix G uniformly at random from $\mathbb{F}_q^{k \times n}$. These are slightly different distributions, because a matrix G drawn at random may not have full rank. However, for the parameter values we are interested in, these distributions are very similar. We will abuse language and use "uniformly

[4]In coding theory, one generally writes G as a short fat matrix and multiplies messages as row vectors on the left. When in Rome...

[5]The parity-check matrix is multiplied by a column vector on the right—all is right with the world.

random linear code" to refer to both of these distributions. When it comes time to make precise statements, we will be more clear about which we mean.

Linear codes have a lot of structure. If Alice has her hands on G, she may encode a message $x \in \mathbb{F}_q^k$ as $x^T G \in \mathcal{C}$ reasonably efficiently. If Bob has his hands on G, he may test quickly whether or not $z \in \mathbb{F}_q^n$ is a codeword in $\mathcal{C}$, and if it is, he may quickly recover the corresponding message x. Linear codes obey a lot of useful symmetries: for example, the distance of a linear code is same as the distance of any codeword to the all-zero codeword. We define here a few linear codes which will be especially useful to us.

Reed-Solomon codes

The work in Chapter 4 is motivated by *Reed-Solomon codes.* Reed-Solomon codes [90], which are based on polynomials over finite fields, are one of the most-studied families in coding theory. They are prevalent in practice, showing up everywhere from storage in CD-ROMs[6] to QR codes for smartphones[7] to schemes for high-thoughput screening of DNA [106]. See [110] for more discussion of the many applications of Reed-Solomon codes.

Definition 2.1. *The* Reed-Solomon *code of degree $k-1$ and length n with evaluation points $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ is*

$$\mathrm{RS}_q(k,n) = \left\{ (f(\alpha_1), \ldots, f(\alpha_n)) \in \mathbb{F}_q^n \,:\, f \in \mathbb{F}_q[x], \deg(f) \leq k-1 \right\}.$$

Notice that the definition of Reed-Solomon codes implies that the alphabet size q must be large; indeed, it must be larger than n. In particular, when we consider the family of Reed-Solomon codes and let n go to infinity, q must also grow to infinity.

[6] too old-fashioned?
[7] that's better

One reason that Reed-Solomon codes are so prevalent is because they have the optimal rate-distance trade-off. To see this, we compute the rate and distance below:

Distance. The distance of $\mathrm{RS}_q(k,n)$ is exactly $(k-1)/n$. Indeed, for any two polynomials f, g of degree at most $k-1$, the number of $\alpha \in \mathbb{F}_q$ so that $f(\alpha) = g(\alpha)$ is at most $k-1$, the number of roots of $f-g$. Conversely, for any set of evaluation points, the distance between the codewords corresponding to $f(x) = 0$ and $g(x) = (x-\alpha_1)(x-\alpha_2)\cdots(x-\alpha_{k-1})$ is precisely $(k-1)/n$.

Rate. The rate of $\mathrm{RS}_q(k,n)$ is exactly k/n; this follows from the fact that the generator matrix for $\mathrm{RS}_q(k,n)$, given by

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \vdots & & \vdots \\ \alpha_1^k & \alpha_2^k & \alpha_3^k & \cdots & \alpha_n^k \end{bmatrix}$$

has full rank.

Thus, a Reed-Solomon code with distance δ has rate $R = k/n = 1 - \delta + 1/n$, exactly matching the Singleton bound. It also matches the "big-q" version of the GV bound: a Reed-Solomon code of distance $\delta = 1 - 1/q - \varepsilon = 1 - \varepsilon - o(1)$ has rate

$$R = 1 - \delta - o(1) = \varepsilon - o(1).$$

Above, we imagine that ε is a small constant, and we recall that $1/q = o(1)$, as $q > n$ for Reed-Solomon codes is not constant.

Expander codes

In Chapter 6, we will study a family of linear codes based on *expander graphs.* These codes, called *Tanner codes,* are formed from d-regular bipartite graphs $G =$

(U, V, E), and an *inner code* $\mathcal{C}_0 \subset \mathbb{F}_q^d$. The idea of using bipartite graphs to define linear codes goes back to Gallager [32] in the 1960's; the version we will use is due to Tanner [105] and to Sipser and Spielman [98]. In these variants, the graph G and the inner code $\mathcal{C}_0$ are used to define a Tanner code $\mathcal{C} = \mathcal{C}(G, \mathcal{C}_0) \subset \mathbb{F}_q^{Nd}$; here, $N = |V| = |U|$ is the number of vertices on each side of G. Thus, the length of $\mathcal{C}$ is $n = Nd$, which is the number of edges in G.

A codeword $c \in \mathcal{C}$ will be interpreted as a labeling of the edges of G. For a vertex u, we will use $\Gamma(u)$ to denote the edges adjacent to u (so, $\Gamma(u) \subset E$ has size d). We will fix an ordering on these edges, and write $\Gamma(u) = \{\Gamma_1(u), \Gamma_2(u), \ldots, \Gamma_d(u)\}$.

Definition 2.2. *Let G be a d-regular bipartite graph, and let $\mathcal{C}_0 \in \mathbb{F}_q^d$. With the notation above, the* Tanner code $\mathcal{C} = \mathcal{C}(G, \mathcal{C}_0)$ *is defined by*

$$\mathcal{C} = \left\{ c \in \mathbb{F}_q^{Nd} \,:\, \forall u \in U \cup V, (c_{\Gamma_1(u)}, c_{\Gamma_2(u)}, \ldots, c_{\Gamma_d(u)}) \in \mathcal{C}_0 \right\}.$$

Above, we index the coordinates of $c \in \mathcal{C}$ by the edges of G, and use c_e to denote the e-th coordinate.

A picture of this construction is shown in Figure 2.4. As suggested above, we will choose the graph G to be an *expander graph.* When the underlying graph G is an expander graph, then it turns out [98] that the code $\mathcal{C}$ can be encoded and decoded extremely quickly, in time linear in n. In Chapter 6, we will give algorithms for decoding these codes in *sublinear* time. We will defer the definition of expander graphs until they are needed in Chapter 6. For now, we will simply observe some facts about the rate of $\mathcal{C}$ when $\mathcal{C}_0$ is linear, for arbitrary bipartite graphs G.

It is not immediately obvious that the Tanner code $\mathcal{C}$ we just defined is non-empty; why should there be any labeling of the edges that are consistent with $\mathcal{C}_0$? If $\mathcal{C}_0$ is linear, we can answer this questions by counting linear constraints. Indeed, if $\mathcal{C}_0$ is

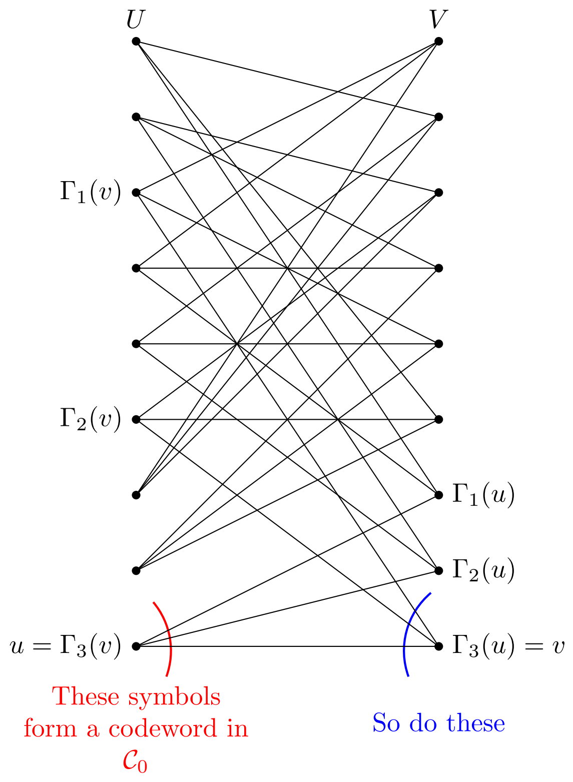


Figure 2.4: A codeword in an Tanner code $\mathcal{C}$ is a labeling of the edges of a bipartite graph G. A labeling c is in $\mathcal{C}$ if at every vertex $u \in U$ and $v \in V$, the labels on the edges coming out of u (and v) form a codeword in an inner code $\mathcal{C}_0$.

a linear code of rate R_0, then it is defined by $d(1-R_0)$ linear constraints, and by definition $\mathcal{C}$ is a linear code defined by $2N(d(1-R_0))$ (possibly redundant) linear constraints, $d(1-R_0)$ constraints for each of the $2N$ vertices. In particular, the rate of $\mathcal{C}$ is at least

$$R(\mathcal{C}) \geq \frac{Nd - 2N(d(1-R_0))}{Nd} = 2R_0 - 1.$$

Thus, as long as $R_0 > 1/2$, $\mathcal{C}$ has positive rate and we have done something nontrivial.

With a few examples of codes under our belts, we will return to the slog of definitions, and introduce two variants of the general coding theory set-up.

2.2 List-Decodable codes

Alice and Bob will fail when the error rate exceeds half of the minimum distance of the code. Indeed, consider the point z in Figure 2.2; if there are two codewords $c, c' \in \mathcal{C}$ with $\delta(c, c') < 2\rho$, then there is always some $z \in \mathbb{F}_q^n$ with $\delta(c, z), \delta(c', z) < \rho$. This is perhaps disappointing: what if the error rate is bigger than 1/2? In some applications, ρ may be nearly 1. Is there anything to be done in this case? The answer is yes: Alice and Bob can use *list decoding*.

In list decoding, Bob is allowed to return a short list of messages $x_1, \ldots, x_L \in \mathbb{F}_q^k$, as long as he can guarantee that Alice's message appears somewhere in the list. Formally, we have the following definition.

Definition 2.3. *A code $\mathcal{C} \subset \mathbb{F}_q^n$ is (ρ, L)-list decodable, if for every $z \in \mathbb{F}_q^n$,*

$$|\{c \in \mathcal{C} \,:\, \delta(z, c) \leq \rho\}| \leq L.$$

We refer to the largest ρ so that Definition 2.3 holds as the *list-decoding radius* of $\mathcal{C}$ for list size L. For our purposes, we will usually hope that L is "reasonably small," which might mean "polynomial in a few parameters of interest." When L is clear, we will sometimes just refer to this ρ as the list-decoding radius of $\mathcal{C}$. The Alice-and-Bob setup is shown in Figure 2.5; the combinatorial interpretation is shown in Figure 2.6.

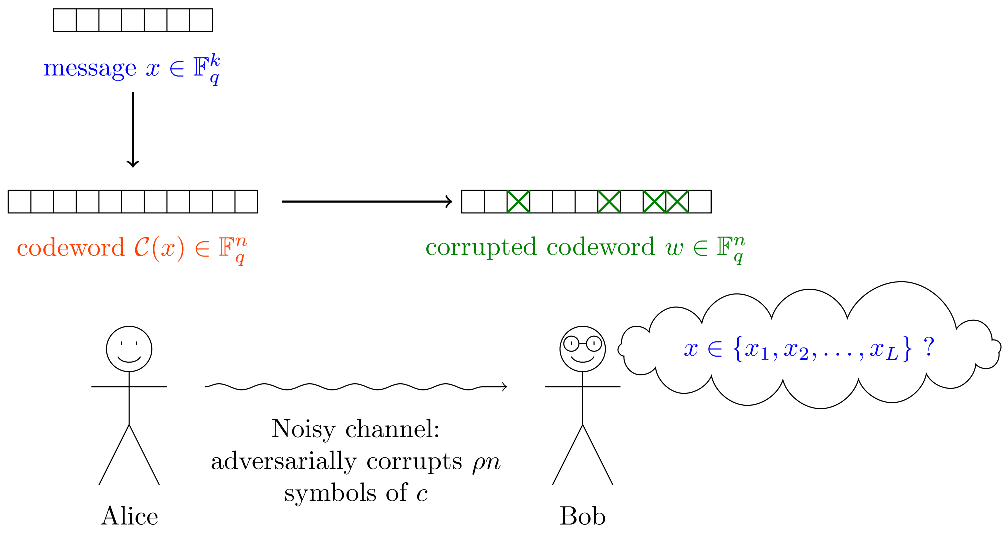

message $x \in \mathbb{F}_q^k$
codeword $\mathcal{C}(x) \in \mathbb{F}_q^n$
corrupted codeword $w \in \mathbb{F}_q^n$
$x \in \{x_1, x_2, \ldots, x_L\}$?
Noisy channel: adversarially corrupts ρn symbols of c
Alice
Bob

Figure 2.5: The set-up for list-decodable codes: Alice-and-Bob version.

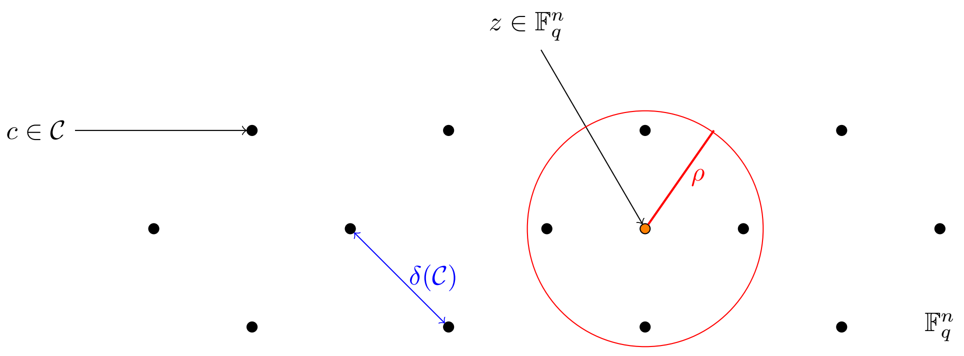


Figure 2.6: The set-up for list-decodable codes, from a combinatorial perspective. Even though the is not a unique codeword $c \in \mathcal{C}$ which are within ρ of z, there are not that many. The code pictured above is $(\rho, 4)$-list-decodable.

List decoding was introduced, independently, by Elias and Wozencraft [27, 112] in the late 1950's. Since then, it has found uses throughout theoretical computer science, not only in communication, but also in complexity theory and pseudorandomness. For example,[8] in complexity theory list decodable codes have been used

[8]for the reader already familiar with the lingo

(often implicitly) in hardness amplification [103], constructing hardcore predicates from one-way functions [36], and in average-case hardness of the permanent [17, 37]. In pseudorandomness, list-decodable codes are intimately connected to pseudorandom gadgets like extractors [60], expanders, condensers, and so on [107]. We refer the reader to the excellent surveys of Sudan [102] and Vadhan [108], as well as to Guruswami's thesis [43], for the many applications of list decodability.

To get a feel for what can and cannot be done in list decoding, we will survey some results and do a few computations below.

2.2.1 List-decoding radius vs. rate

The motivation for list decoding is to handle extremely large error rates; how large is large? First, it is not hard to see that $\rho > 1 - 1/q$ is just too large. Indeed, in expectation a random received word $y \in \mathbb{F}_q^n$ will agree with a given codeword c in a $1/q$-fraction of the places, and so we expect for y to be this close to a large number of $c \in \mathcal{C}$. The remarkable fact is that this is the only barrier: Alice and Bob can handle any $\rho < 1 - 1/q$. The following theorem, called the *list decoding capacity theorem,* pins down the rate we can hope for for any $\rho < 1 - 1/q$.

Theorem 2.4 (List decoding capacity theorem, [28, 116])**.** *Fix $q \geq 2$ and let $\rho \in (1, 1/q)$. Then the following are true.*

(1) For any code $\mathcal{C} \subset \mathbb{F}_q^n$ which is (ρ, L)-list decodable, with rate $R = 1 - H_q(\rho) + \gamma$ for any $\gamma > 0$,

$$L \geq q^{n(\gamma - o(1))}.$$

(2) On the other hand, for all $L \geq 1$, there is a (ρ, L)-list-decodable code with rate

$$R \leq 1 - H_q(\rho) - 1/L.$$

The proof of Theorem 2.4 is relevant for some of the results in this dissertation, so we include it here.

Proof. For Item (1), consider picking a random received word $z \in F^n$, and fix $c \in \mathcal{C}$. We have

$$\mathbb{P}\left\{c \in B_q(z,\rho)\right\} = \frac{|B_q(c,\rho)|}{q^n} \geq q^{-n(1-H_q(\rho)-o(1))},$$

using (2.4). Then, in expectation over z,

$$\mathbb{E}\left[|B_q(z,\rho) \cap \mathcal{C}|\right] \geq q^{Rn} q^{-n(1-H_1(\rho)-o(1))} = q^{n(\varepsilon - o(1))}.$$

In particular, there is some z so that the number of codewords within ρ of z is exponentially large in n, the block length of $\mathcal{C}$.

For Item (2), we again will use the probabilistic method. Fix $R \leq 1 - H_q(\rho) - 1/L$, and let $k \leq Rn$ be an integer. Choose $\mathcal{C} = \left\{\mathcal{C}(x) : x \in \mathbb{F}_q^k\right\} \subset \mathbb{F}_q^n$, so that $\mathcal{C}(x)$ is chosen uniformly at random, independently for the different $x \in \mathbb{F}_q^k$. Now, for a fixed $z \in \mathbb{F}_q^n$ and a fixed set of messages $\Lambda \subset \mathbb{F}_q^k$, with $|\Lambda| = L+1$, consider the event $E_{z,\Lambda}$ that

$$\mathcal{C}(x) \in B_q(z,\rho) \qquad \forall x \in \Lambda$$

The probability of this event is

$$\mathbb{P}\left\{E_{z,\Lambda}\right\} = \prod_{x \in \Lambda} \mathbb{P}\left\{\mathcal{C}(x) \in B_q(z,\rho)\right\} \leq \left(\frac{|B_q(z,\rho)|}{q^n}\right)^{L+1} \leq q^{-n(L+1)(1-H_q(\rho))},$$

where in the first equality we have used independence, and in the last inequality we have again used (2.4). Now, by the union bound, the probability that $E_{z,\Lambda}$ occurs for *any* z, Λ is

$$\begin{aligned}\mathbb{P}\left\{\exists z, \Lambda \text{ such that } E_{z,\Lambda}\right\} &\leq q^n \binom{q^k}{L+1} q^{-n(L+1)(1-H_q(\rho))} \\ &\leq q^n q^{k(L+1)} q^{-n(L+1)(1-H_q(\rho))} \\ &\leq q^{n(1+(1-H_q(\rho)-1/L)(L+1)-(L+1)(1-H_q(\rho)))} \\ &= q^{-n/L} \\ &< 1.\end{aligned}$$

In particular, there exists a code $\mathcal{C}$ so that none of the events $E_{z,\Lambda}$ occur. But if this is the case, then by definition $\mathcal{C}$ is (ρ, L)-list-decodable. □

Let us apply our intuition (2.8) about the behavior of $H_q(1 - 1/q - \varepsilon)$ to the conclusions of Theorem 2.4 in the parameter regime when $\rho = 1 - 1/q - \varepsilon$:

Corollary 2.5. *Suppose that $\mathcal{C}$ is $(1 - 1/q - \varepsilon, L)$-list decodable with list size L polynomial in $1/\varepsilon$. Then the rate R of $\mathcal{C}$ must obey*

$$R \leq 1 - H_q(1 - 1/q - \varepsilon) \leq \min\left\{\varepsilon, \frac{q\varepsilon^2}{\log(q)}\right\}.$$

Corollary 2.5 is not the tightest statement we could make (in terms of the constants), but for this thesis we care about the asymptotic dependence of ε and q, rather than the exact values of the constants.

The "large-ρ" parameter regime

The take-away from Theorem 2.4 and Corollary 2.5 is that *list decoding effectively doubles the correctable fraction of errors.* For any (nontrivial) code over an alphabet of size q, the distance cannot be more than $1 - 1/q$, and so no more than a $\frac{1}{2}\left(1 - \frac{1}{q}\right)$ fraction of errors can be recovered from uniquely. However, when the decoder may output a short list, there are codes which can tolerate a $1 - \frac{1}{q} - \varepsilon$ fraction of errors, for any $\varepsilon > 0$. This fact has been crucially exploited in numerous applications of list decoding in theoretical computer science and in particular to the complexity theoretic applications mentioned above. There are two important features of these applications:

1. For complexity applications, it is necessary for the fraction of correctible errors to be arbitrarily close to $1 - \frac{1}{q}$. This is less important in the communication setting (where we might hope ρ is close to 0), but for clarity of exposition we will stick with Alice and Bob: our motivation is captured in Figure 2.7.

2. As we saw above, the optimal rate to correct $1 - \frac{1}{q} - \varepsilon$ fraction of errors is known, given by

$$R^*(q, \varepsilon) := 1 - H_q(1 - 1/q - \varepsilon),$$

   and bounded by Corollary 2.5. However, for complexity applications it is often enough to design a code with rate $\Omega(R^*(q, \varepsilon))$ with the same error correction capability.[9]

We study list decoding in these parameter regimes. That is, we seek to correct a $1 - 1/q - \varepsilon$ fraction of errors, with rate $\widetilde{\Omega}(R^*(q, \varepsilon))$ which may be suboptimal by multiplicative factors. The ultimate goal is to get the correct dependence on ε and q.

The proof of (2) in Theorem 2.4 implies that a random code $\mathcal{C}$ is optimally list-decodable with high probability. We digress for a moment to remark on the importance that independence played in this proof: if the encodings $\mathcal{C}(x)$ were not independent, for different x, then a priori the probability of the event $E_{z,\Lambda}$ might be much larger, and the union bound would not go through. For example, suppose

[9]In fact in some applications even polynomial dependence on $R^*(q, \varepsilon)$ is sufficient.

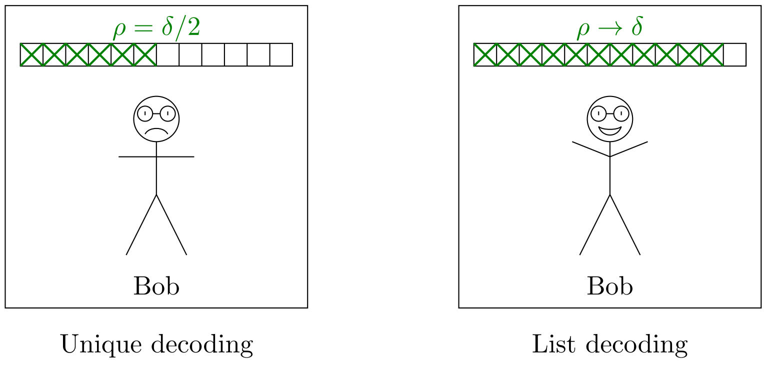


Figure 2.7: Moving from unique decoding to list decoding allows Alice and Bob to nearly double the tolerable error rate, from $\rho = \delta/2$ to $\rho = \delta$. To illustrate this phenomenon, we include a picture above of how happy this makes Bob.

that instead of taking $\mathcal{C}$ to be a random code, we considered a random linear code. That is, choose a random matrix $G \in \mathbb{F}_q^{n \times k}$, and set $\mathcal{C}(x) = Gx$. Now it is no longer the case that the encodings $\{\mathcal{C}(x) : x \in \Lambda\}$ are independent; in fact, since $Gx + Gy = G(x+y)$, they are not even 3-wise independent. We may modify the approach of the proof above (following [116]) to work.

Proposition 2.6 ([116]). *Let $q \geq 2$ and choose $\rho \in (0, 1-1/q)$. Let $\mathcal{C}$ be a random linear code of rate $R \leq 1 - H_q(\rho) - \frac{1}{\lceil \log_q(L+1) \rceil}$. Then with high probability $\mathcal{C}$ is (ρ, L)-list-decodable.*

Proof. We modify the proof of Theorem 2.4, part (2), above. As before, the plan will be to bound $\mathbb{P}\{E_{z,\Lambda}\}$ and take a union bound. Any set $\Lambda \subset \mathbb{F}_q^k$ of $L+1$ messages must contain at least $\log_q(L+1)$ linearly independent vectors: this follows because any subspace of $\mathbb{F}_q^k$ of dimension t contains at most q^t messages. Now, for any set of linearly independent messages $x \in \mathbb{F}_q^k$, the corresponding codewords $\mathcal{C}(x) = Gx \in \mathbb{F}_q^n$ are independent random variables. Thus, we may bound

$$\mathbb{P}\{E_{z,\Lambda}\} \leq \left(\mathbb{P}\{Gx \in B_q(z,\rho)\}\right)^{\lceil \log_q(L+1) \rceil} \leq q^{-n(1-H_q(\rho))\lceil \log_q(L+1) \rceil}.$$

The proof proceeds by taking a union bound, as before.

□

We note that Proposition 2.6 is exponentially worse than part (2) of Theorem 2.4, in terms of the list sizes. Indeed, when $\rho = 1 - 1/q - \varepsilon$, then in order to obtain the "correct" rate (as per Corollary 2.5), we must set $L = q^{1/\varepsilon}$ or q^{1/ε^2}, which is much larger than $1/\mathrm{poly}(\varepsilon)$. It is a natural question whether or not we can do better [28]. We will return to this question in Chapters 3 and 4, where we will answer it in the affirmative.

2.2.2 List-decoding radius vs. distance, and the Johnson bound

Above, we quantified the best trade-off we can hope for between the rate R of a code and its list-decoding radius ρ. A related question is the trade-off between the distance δ and the list-decoding radius ρ. We have already discussed the trade-off between δ and R, summarized by (2.5) and (2.8), and this gives us an idea about what we might hope for for the trade-off between δ and ρ.

Intuitively, it seems like good distance should be enough to imply a large list-decoding radius. Indeed, in Figure 2.6, it seems reasonable that if all of the points of $\mathcal{C}$ are very spread out, there should be no way to capture too many of them in a ball of radius ρ. What sort of trade-off could we hope for? Suppose that $\mathcal{C}$ lies on the Gilbert-Varshamov bound (the first inequality in (2.5)) and has distance δ; thus the rate is $R = 1 - H_q(\delta) - o(1)$. Then the list-decoding capacity theorem (Theorem 2.4) indicates that we may hope to obtain nontrivial list-decoding guarantees with the list-decoding radius ρ approaching the distance δ of the code. This quantitative intuition might make us hope that any code with distance $1 - 1/q - \Omega(\varepsilon)$ should have list-decoding radius $\rho = 1 - 1/q - \varepsilon$. Unfortuately, this is just too good be true.[10]

[10] Indeed, a random coding argument [38, Section 4.3] shows that there are (non-linear) q-ary codes of distance

But there is some statement we can make along these lines, known as the Johnson Bound.

Theorem 2.7 (Johnson bound, [72]). *Let $\mathcal{C} \subset \mathbb{F}_q^n$ have distance δ, and let*

$$\rho \leq (1 - 1/q)\left(1 - \sqrt{1 - \frac{q\delta}{q-1}}\right) =: J_q(\delta).$$

Then $\mathcal{C}$ is (ρ, L)-list decodable, for $L = qn^2\delta$.

When $\delta = 1 - 1/q - \varepsilon^2$, then $J_q(\delta)$ is at least $1 - 1/q - \varepsilon$. Thus, in our setting, the Johnson bound states that any code with distance $1 - 1/q - \varepsilon^2$ has list-decoding radius $\rho \geq 1 - 1/q - \varepsilon$.

Average-radius, average-distance Johnson bound

There are many proofs of the Johnson bound [1,20,28,29,38,58,59,72,73,81]. We will give a proof here, for $q = 2$ and $d = 1/q - \varepsilon$, which will be instructive in the future. This proof is similar to (and inspired by) the proof in [20].

Theorem 2.8 (Average-distance, average-radius Johnson bound for $q = 2$). *Let $\mathcal{C} \subset \mathbb{F}_2^n$ be a binary code. Then for any $\Lambda \subset \mathbb{F}_2^n$ with $|\Lambda| = L$, and for any $z \in \mathbb{F}_2^n$,*

$$\frac{1}{L}\sum_{x \in \Lambda} \delta(\mathcal{C}(x), z) \geq \frac{1}{2}\left(1 - \sqrt{1 - \frac{2}{L^2}\sum_{x \neq y \in \Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y))}\right).$$

Proof. Let $\Phi \in (\pm 1)^{n \times 2^k}$ be the matrix whose columns are indexed by $x \in \mathbb{F}_2^k$, so

$1 - 1/q - \varepsilon$ so that one can find a Hamming ball of radius $1 - 1/q - \Omega(\sqrt{\varepsilon})$ which contains super-polynomially many codewords. There are similar results for linear codes [42,55].

that $\Phi_{j,x} = (-1)^{\mathcal{C}(x)_j}$. Let φ_j denote the j-th column of Φ. Then

$$\begin{aligned}
\max_z \sum_{x\in\Lambda} 1 - \delta(\mathcal{C}(x), z) &= \frac{1}{n}\sum_{j=1}^{n} \max_{b\in\{0,1\}} \sum_{x\in\Lambda} \mathbf{1}_{\mathcal{C}(x)_j=\alpha} \\
&= \frac{1}{n}\sum_{j=1}^{n} \max_{\alpha\in\{0,1\}} \sum_{x\in\Lambda} \frac{(-1)^{\alpha}(-1)^{\mathcal{C}(x)_j}+1}{2} \\
&= \frac{1}{n}\sum_{j=1}^{n} \left(L + \sum_{j=1}^{n} |\langle \varphi_j, \mathbf{1}_\Lambda\rangle| \right) \\
&= \frac{1}{2}\left(L + \frac{1}{n}\left\|\Phi \mathbf{1}_\Lambda\right\|_1 \right) \\
&\le \frac{1}{2}\left(L + \frac{1}{\sqrt{n}}\left\|\Phi \mathbf{1}_\Lambda\right\|_2 \right),
\end{aligned}$$

using Cauchy-Schwarz in the final line. The claim then follows from the definition of Φ and the fact that the (x, y)-entry of $\Phi^T\Phi$ is given by $n(1 - 2\delta(\mathcal{C}(x), \mathcal{C}(y)))$. Indeed, from this, we have

$$\left\|\Phi\mathbf{1}_\Lambda\right\|_2^2 = \mathbf{1}_\Lambda^T \Phi^T \Phi \mathbf{1}_\Lambda = n \sum_{x\in\Lambda}\sum_{y\in\Lambda} \left(1 - 2\delta(\mathcal{C}(x), \mathcal{C}(y))\right),$$

and plugging this in above and a little bit of rearrangement gives the statement. $\square$

Theorem 2.8 is stronger than Theorem 2.7 in two important ways; to understand them, we will derive Theorem 2.7 from Theorem 2.8. First, notice that by the definition of list-decodability, $\mathcal{C}$ is list-decodable if and only if, for all sets $\Lambda \subset \mathbb{F}_q^k$ of size $L+1$, and for all $z \in \mathbb{F}_q^n$, there is at least one $x \in \Lambda$ so that $\delta(\mathcal{C}(x), z) \ge \rho$; in other words,

$$\max_{x\in\Lambda} \delta(\mathcal{C}(x), z) \ge \rho.$$

Because the average is always smaller than the maximum, it suffices for

$$\frac{1}{L}\sum_{x\in\Lambda} \delta(\mathcal{C}(x), z) \ge \rho,$$

which is the form that the bound in Theorem 2.8 takes. Next, we observe that if the minimum distance $\delta(\mathcal{C})$ of $\mathcal{C}$ is large, then the averaged distance term that shows up in Theorem 2.8 is also large:

$$\sum_{x \neq y \in \Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y)) \geq L(L-1)\delta(\mathcal{C}).$$

Incorporating these observations into the statement of Theorem 2.8, we obtain that a binary code $\mathcal{C}$ is (ρ, L)-list-decodable for all

$$\rho \leq \frac{1}{2}\left(1 - \sqrt{1 - 2\left(1 - \frac{1}{L+1}\right)\delta}\right).$$

Comparing this to $J_2(\delta) = \frac{1}{2}\left(1 - \sqrt{1-2\delta}\right)$, we find that for large L these are basically the same in our parameter regime: if we are shooting for $\rho = 1/2 - \varepsilon$, we may take $\delta = 1/2 - O(\varepsilon^2)$ and $L = O(1/\varepsilon^2)$.

We call Theorem 2.8 a *average-radius, average-distance* Johnson bound. It is average-radius because it shows that the averaged list-decoding radius

$$\max_{z,\Lambda} \frac{1}{L} \sum_{x \in \Lambda} \delta(\mathcal{C}(x), z)$$

is large. It is average-distance because it depends on the averaged distances

$$\frac{1}{L(L-1)} \sum_{x \neq y} \delta(\mathcal{C}(x), \mathcal{C}(y)),$$

rather than the minimum distance. Many proofs of the Johnson bound can actually be tweaked to imply average-radius or average-distance Johnson bounds. This fact appears to be folklore, but the distinction is important to us. In Chapter 4 we will give a few more average-radius, average-distance Johnson bounds, which hold for all q.

2.2.3 List decoding of Reed-Solomon codes and beyond

Reed-Solomon codes play an important role in the history of list decoding. Like any code with good distance, the Johnson bound implies that Reed-Solomon codes have large list-decoding radius. More precisely, Theorem 2.7, combined with our earlier calculations about Reed-Solomon codes, imply that a Reed-Solomon code of rate ε^2 (over $\mathbb{F}_q$ for $q \gg 1/\varepsilon^2$) has distance about $1 - \varepsilon^2$ and hence list-decoding radius $\rho = 1 - \varepsilon$. The remarkable thing about Reed-Solomon codes is that they can be list-decoded to this radius *efficiently*.

The celebrated work of Guruswami and Sudan [57, 101], in the late 1990's, gave an efficient list-decoding algorithm for Reed-Solomon codes which works up to the Johnson bound. This was the first non-trivial progress in efficiently list decoding codes up to radius $1 - \varepsilon$, and it made a big impact. However, as we saw from the List Decoding Capacity Theorem (Theorem 2.4), this is not the best rate/list-decoding-radius trade-off we could hope for. Ideally, if $\rho = 1 - \varepsilon$, we ought to be able to find codes of rate ε.

Eventually, the "Johnson bound barrier" was broken for efficiently decodable codes [87], and now we know of several families of codes which are efficiently list decodable all the way to the list-decoding capacity theorem [51, 62–64, 78]. For the most part, these codes are generalizations of Reed-Solomon codes. However, no more progress was made on Reed-Solomon codes themselves, except for a few negative results. Indeed, it has been conjectured that Reed-Solomon codes are *not* list-decodable beyond the Johnson bound [19], and so significant effort has been put in to proving this. So far, we know that if the evaluation points contain certain algebraic structure, then indeed Reed-Solomon codes can't be list-decoded, even combinatorially, much beyond the Johnson bound [12]. Further, if we pass to a related problem, called *list-recovery,* then the analogue of the Johnson bound is the right answer for Reed-Solomon codes [50]. Finally, it seems likely that, no matter what the evaluation points, list-decoding Reed-Solomon codes much beyond the Johnson bound will be computationally difficult [19], even if it is combinatorially possible.

We will return to this question later. One of the main contributions of Chapter 4 is that there *are* Reed-Solomon codes which are list-decodable well beyond the Johnson bound, and nearly to list-decoding capacity. In fact, we will show that most Reed-Solomon codes achieve this, in the sense that choosing the evaluation points at random is a good bet.

2.2.4 Summary

To sum up, the state of (existential) knowledge about list-decodable codes, before the work in this thesis, is as follows.

- Random codes are list-decodable to capacity. Other random ensembles of codes, even the simple case of random linear codes, have proved difficult to analyze.
- The Johnson bound gives us a structural condition (distance) which implies good list-decodability, but it does not (and cannot[11]) go as far as the lower bound imposed by the list-decoding capacity theorem.
- We know of a few, very specific, families of codes, based on Reed-Solomon codes, which are list-decodable to capacity. These codes also are efficiently list-decodable.
- As for Reed-Solomon codes themselves, we know that some choices of evaluation points will obstruct list-decoding to capacity.

There are some gaps and open questions in this landscape. We will return to these later in Chapters 3, 4, and 5, where we will fill some gaps and answer some questions. For now, we will consider the basics of list-decoding covered, and move on to *locally decodable codes.*

[11]That is, there are codes whose distance and list-decoding radius match the Johnson bound, see [55]

2.3 Locally Decodable codes

Suppose that Bob only wants to recover a single symbol of Alice's message x. Of course, if Bob is equipped to recover all of x, as he has been so far, he can do this easily. However, in order to recover all of x, Bob must at least look at the entire codeword $c = \mathcal{C}(x)$, and in particular the time he will take to do this is $\Omega(n)$. In local decoding, one wonders if Bob might do better.

More precisely, the setup will be as follows. Alice will encode her message as $c = \mathcal{C}(x) \in \mathbb{F}_q^n$, as before. As before, an adversary will corrupt a ρ-fraction of the symbols of c, to create a corrupted codeword $w \in \mathbb{F}_q^n$. Bob will be given query access to w. Now, the adversary additionally gives Bob an index $i \in \{1, \ldots, k\}$. Bob's job will be to make $Q \ll n$ *queries* to w, and from these queries he must produce a guess of x_i. It is not hard to see that Bob's queries must be randomized: indeed, because the adversary will know Bob's stratregy, if Bob were to look at a deterministic set of $Q \ll n$ queries, then the adversary could simply corrupt every single query Bob observes. Thus, we will demand that, for all i, he succeed with high probability. The setup is shown in Figure 2.8, and a more formal definition is given below.

Remark 1. *In locally decodable codes, the number of* ***q****ueries is generally denote by q. Of course, in coding theory, it is often the case that $q = |\mathbb{F}_q|$ is the size of the alphabet. Resolving this notational collision is one of the greatest open problems in local decoding. Not wanting to bite of more than we can chew with this thesis, we will punt and denote the number of queries by Q.*

Definition 2.9 (Locally Decodable Codes (LDCs))**.** *Let $\mathcal{C} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ be a code. Then $\mathcal{C}$ is (Q, ρ)-locally decodable with error probability η if there is a randomized* *algorithm Δ, so that for any $w \in \mathbb{F}_q^n$ with $\delta(w, \mathcal{C}(x)) < \rho$, for each $i \in [k]$,*

$$\mathbb{P}\left\{\Delta(w,i) = x_i\right\} \geq 1 - \eta,$$

and further Δ accesses at most Q symbols of w. Here, the probability is taken over the internal randomness of the decoding algorithm Δ.

In this dissertation, we will also be concerned with a slightly stronger notion, called a locally correctable code (LCC). In this setup, Bob must be able to not only find every symbol of Alice's message x, but also of the codeword $\mathcal{C}(x)$.

Definition 2.10 (Locally Correctable Codes (LCCs))**.** *Let $\mathcal{C} \subset \mathbb{F}_q^n$ be a code. Then $\mathcal{C}$ is (Q, ρ)-locally correctable with error probability η if there is a randomized algorithm, Δ, so that for any $w \in \mathbb{F}_q^n$ with $\delta(w, \mathcal{C}(x)) < \rho$, for each $j \in [n]$,*

$$\mathbb{P}\left\{\Delta(w,j) = w_j\right\} \geq 1 - \eta,$$

and further Δ accesses at most Q symbols of w. Here, the probability is taken over the internal randomness of the decoding algorithm Δ.

A locally correctable *linear* code gives a locally decodable code—this follows from the fact that we may always put the generator matrix in canonical form, so that the left-most $k \times k$ block is the identity. In this view, the message itself appears as part of the codeword, and so the ability to recover any symbol of the codeword is enough to recover any symbol of the message.

2.3.1 Two examples: Hadamard codes and Reed-Muller codes

It is worthwhile to consider two examples of locally decodable codes. The first example is the (q-ary) *Hadamard code.*

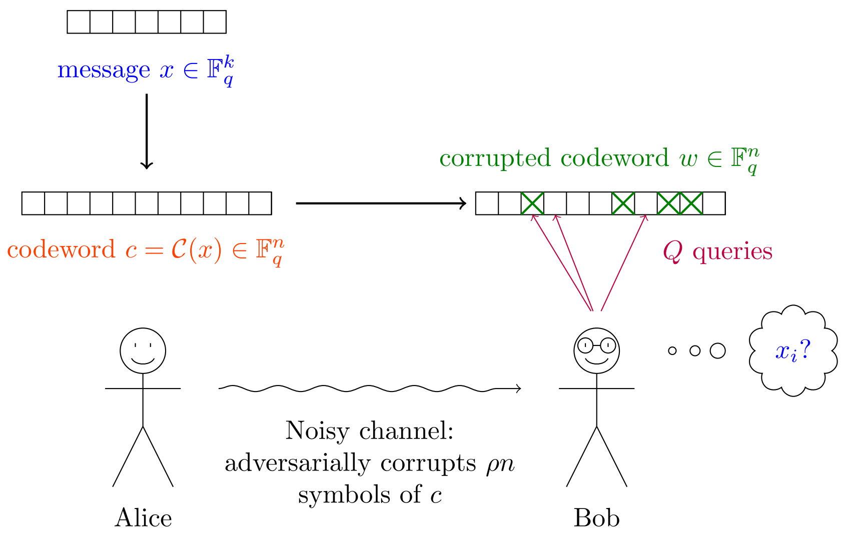

message $x \in \mathbb{F}_q^k$
corrupted codeword $w \in \mathbb{F}_q^n$
codeword $c = \mathcal{C}(x) \in \mathbb{F}_q^n$
Q queries
x_i?
Noisy channel: adversarially corrupts ρn symbols of c
Alice
Bob

Figure 2.8: The set-up for locally decodable codes. For each $i \in \{1, \ldots, k\}$, Bob must be able to guess x_i with high probability over his choice of queries. The symbols are corrupted after $\mathcal{C}(x)$ is encoded, but before Bob makes his queries.

Definition 2.11 (Hadamard code)**.** *For $n = q^k$, the Hadamard code of length n encodes messages in $x \in \mathbb{F}_q^k$ by*

$$\mathcal{C}(x) = (\langle a_i, x \rangle)_{i=1}^n \in \mathbb{F}_q^n,$$

where a_i ranges over all elements of $\mathbb{F}_q^k$.

In other words, the Hadamard code is the linear code whose generator matrix is the $k \times q^k$ matrix whose rows include every vector in $\mathbb{F}_q^k$. The rate of the Hadamard code is k/q^k; it approaches 0 as $k \to \infty$. However, the Hadamard code is $(2, \rho)$-locally correctable for any $\rho < (1 - 1/q)/2$. Algorithm 1 shows how Bob may recover $\mathcal{C}(x)_a$ for some $a \in \mathbb{F}_q^k$, by looking at only two entries of a corrupted codeword w.

Algorithm 1: Local correction algorithm for the Hadamard code

```
**Input**: Corrupted codeword $w \in \mathbb{F}_q^n$, index $a \in \mathbb{F}_q^k$.
Choose $r \in \mathbb{F}_q^k$ uniformly at random.
Choose $s = a - r$.
Query $w_s$ and $w_r$.
**return** $w_s + w_r$
```

Why does Algorithm 1 work? Suppose that Bob's two queries w_s and w_r are correct. Then

$$w_s = \mathcal{C}(x)_s = \langle x, s \rangle \quad \text{and} \quad w_r = \mathcal{C}(x)_r = \langle x, r \rangle \,,$$

and so the value returned by Algorithm 1 is

$$w_s + w_r = \langle x, s + r \rangle = \langle x, r + a - r \rangle = \langle x, a \rangle = \mathcal{C}(x)_a.$$

Thus, the algorithm works whenever these two symbols are not corrupted. While the two queries are correlated, the marginals of each are uniformly random in $\mathbb{F}_q^k$. The probability that a random query is corrupted is ρ, the fraction of corrupted symbols. Thus, by the union bound, the probability (over the choice of r) that both queries are correct is

$$\mathbb{P}\left\{ w_s = \mathcal{C}(s) \qquad \text{and} \qquad w_r = \mathcal{C}(r) \right\} \geq 1 - 2\rho.$$

Thus, as long as ρ is small enough, Alice and Bob can succeed. In particular, if $\rho < (1 - 1/q)/2$, then Alice and Bob are doing better than guessing, so we declare success. If a higher success probability is required, Alice and Bob may do several independent repetitions of Algorithm 1 and take the majority vote; this will allow them to get arbitrarily good success probability, at the cost of more queries.

Our second example is the *Reed-Muller code.*

Definition 2.12 (Reed-Muller code)**.** *The q-ary m-variate Reed-Muller code of degree $d < q - 1$, denoted $\mathrm{RM}_q(d, m)$, is a linear code $\mathcal{C} \subset \mathbb{F}_q^n$, where $n = q^m$. The message length is $k = \binom{m+d}{d}$, and we regard each message $f \in \mathbb{F}_q^k$ as a polynomial in $\mathbb{F}_q[x_1, x_2, \ldots, x_m]$ of degree at most d. Then the encoding $\mathcal{C}(f)$ of f is all of the evaluations of f over $\mathbb{F}_q^m$:*

$$\mathcal{C}(f) = (f(x))_{x \in \mathbb{F}_q^m} \,.$$

Remark 2. *Later on, we will consider Reed-Muller codes in a slightly different parameter regime (which the reader may be more familiar with), where $q = 2$ and $d \geq q - 1$ may be larger. We'll see below why we want the restriction $d < q - 1$ for locally decodable codes.*

We note that Hadamard codes are thus just the k-variate Reed-Muller code of degree 1; Reed-Muller codes can also be seen as a multivariate version of the Reed-Solomon codes we have already encountered.

Algorithm 2: Local correction algorithm for Reed-Muller codes

```
Input: Corrupted codeword $w \in \mathbb{F}_q^n$, index $a \in \mathbb{F}_q^m$.
Choose $r \in \mathbb{F}_q^m$ at random.
Let $L = \{a + \lambda r : \lambda \in \mathbb{F}_q\} \subset \mathbb{F}_q^m$ be the line through $a$ and $r$.
Query $w_b$ for $b \in L \setminus \{a\}$.
Find a univariate polynomial $g : \mathbb{F}_q \to \mathbb{F}_q$ so that
    $g(\lambda) = w_{(a+\lambda r)}$
for the most number of $\lambda$'s.
return $g(0)$
```

Algorithm 2 gives a local correction procedure for Reed-Muller codes; it requires a bit more explanation than Algorithm 1. First, observe that Algorithm 2 makes $Q = q-1$ queries, the number of points on a line in $\mathbb{F}_q^m$ (except for a itself). Second, the restriction of an m-variate polynomial of degree at most d to a line is a univariate polynomial of degree at most d. Thus, the queries $\{\mathcal{C}(f)_b : b \in L\}$ form a corrupted codeword of the q-ary Reed-Solomon code of degree d. We have seen that the distance of this code is $1 - d/q$; any two codewords disagree in at least $q - d$ places. Thus, if there are no more than $(q-d)/2-1$ corruptions in our $q-1$ queries, the polynomial g in Algorithm 2 agrees with the restriction of f to L. In fact, one may find this polynomial g efficiently. After g has been recovered, we have

$$g(0) = f(a + 0 \cdot r) = f(a) = \mathcal{C}(f)_a.$$

It remains to check when it's the case that not too many of the queries are corrupted. As with the Hadamard code, the queries of Algorithm 2 are correlated, but the marginals are uniformly random. Thus, we expect a ρ-fraction of the symbols indexed by L to be corrupted. By Markov's inequality,

$$\mathbb{P}\left\{\text{more than } \frac{q-d-2}{2} \text{ queries are corrupted}\right\} \leq \frac{2\rho(q-1)}{q-d-2},$$

and so the probability of success is at least

$$\mathbb{P}\left\{\text{algorithm 2 works }\right\} \geq 1 - \frac{2\rho}{1-\frac{d+1}{q-1}}.$$

This is better than $1/q$ whenever

$$\rho \leq \frac{1}{2}\left(1 - \frac{d+2}{q}\right).$$

There are many ways to pick parameters for Reed-Muller codes to make this algorithm work. For illustration, consider $m = 2$ and $d = q/2$. Then the rate of the corresponding Reed-Mulller code is

$$R = \frac{\binom{m+d}{d}}{q^m} = \frac{\binom{q/2+2}{2}}{q^2} = \frac{1}{8} + O(1/q).$$

The query complexity is

$$Q = q - 1 = \sqrt{n} - 1.$$

We can make the rate better by choosing d larger; notice that we can never choose $d \geq q-2$, or else the tolerable error rate ρ becomes 0, and so the rate can never become larger than 1/2.

2.3.2 Two parameter regimes

These two examples live on different sides of the spectrum: Hadamard codes have rate which tends to zero exponentially quickly, but use only two queries. Reed-Muller codes (with parameter settings like those above) have constant rate, approaching 1/2, but query complexity about $\sqrt{n}$. It is natural to ask if one can do better in either regime: if we have constant query complexity, can we have subexponential blowup in the length of the codeword? If we have query complexity that scales like n^ε, can we have rate arbitrarily close to 1? The answer to both questions—both open until recently—is yes. We briefly survey work in this direction below.

- **Constant locality.** It is known [77] that one cannot improve on the rate of the Hadamard code if we require the query complexity to be 2. However, a great surprise of the past decade is that there are 3-query LDC's with n slightly subexponential in k [10, 11, 18, 22, 25, 26, 71, 113]. These codes, called *matching vector codes,* have a strategy similar to the strategy we pursued for Hadamard codes. That is, these codes have what we will call a *smooth local reconstruction algorithm.* By this we mean an algorithm which for any symbol x_i can make a few queries so that (a) if all the queries are correct, it will correctly find x_i, and (b) while the queries may be correlated, the marginals are (close to) uniform. Then the same argument we used in the Hadamard case goes through: by a union bound, with high probability none of the queries are corrupted.

- **Locality n^ε.** As with the constant-query regime, the existence of locally decodable (or correctible) codes with rate approaching 1 for any nontrivial number of queries was an open question until recently. In the past few years, there have been two such constructions, *multiplicity codes* [79], and *lifted codes* [41]. In this thesis, we will give a third example [68], which is very different in flavor.

  The codes in [41, 79] are based on polynomials over finite fields, and at a high level the arguments are similar to the Reed-Muller argument we made above.

> One cannot hope to argue that with high probability none of the queries are corrupted; indeed, if there are n^ε queries then with very high probability about a ρ-fraction of them are corrupted. Instead, one can set things up so that the queries themselves form some other code, like the Reed-Muller queries formed a Reed-Solomon code.

Is there some way to turn a smooth local reconstruction algorithm into a locally decodable code the large-locality regime? We will return to this question in Chapter 6, where we will show how to make such an argument work. For now, we will momentarily leave the discrete world of finite fields and go over some of the (continuous) probabilistic tools we'll need.

2.4 Random tools

In this section, we briefly review some probability background that we will need for our analyses.

2.4.1 Gaussian random variables

A *Gaussian* (or *normal*) random variable $g \sim N(0, \sigma^2)$ with variance σ^2 has a probability density function

$$f(t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-t^2/2\sigma^2),$$

which is shown in Figure 2.9. One fact which we will use repeatedly about Gaussians is that they are very well concentrated. More precisely, the cumulative distribution function,

$$\mathbb{P}\{g > t\} = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{u=t}^{\infty} \exp(-u^2/2\sigma^2)\, du$$

obeys the estimate

$$\mathbb{P}\{g > t\} \leq \frac{\sigma}{t} \cdot \frac{1}{\sqrt{2\pi}} \exp(-t^2/2\sigma^2) \tag{2.9}$$

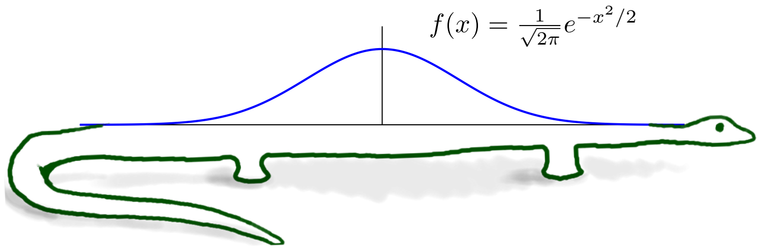


Figure 2.9: The probability density function of a standard Gaussian random variable. It looks a bit like a Brontosaurus.

for all $t > 0$. Indeed, because on the domain $u \geq t$, $(u/t) \geq 1$, we have

(2.10)
$$\frac{1}{\sqrt{2\pi\sigma^2}} \int_{u=t}^{\infty} \exp\left(\frac{-u^2}{2\sigma^2}\right) du \leq \frac{1}{\sqrt{2\pi\sigma^2}} \int_{u=t}^{\infty} \frac{u}{t} \exp\left(\frac{-u^2}{2\sigma^2}\right) du = \frac{\sigma}{t\sqrt{2\pi}} \exp\left(\frac{-t^2}{2\sigma^2}\right).$$

Another very nice fact about Gaussian random variables is that they are *stable:* linear combinations of Gaussian random variables are again Gaussian.

Fact 2.13. *Let $g_1, \ldots, g_n$ be Gaussian random variables with variances $\sigma_1^2, \ldots, \sigma_n^2$. Then the random variable $\sum_i a_i g_i$ is again a Gaussian random variable, with variance $\sum a_i^2 \sigma_i^2$.*

2.4.2 Suprema of Gaussian processes

Several times in this thesis, we will encounter the problem of estimating something of the form

$$w(T) := \mathbb{E} \sup_{t \in T} \langle g, t \rangle, \tag{2.11}$$

for some set $T \subset \mathbb{R}^n$, where $g = (g_1, \ldots, g_n) \sim N(0, I)$ is a Gaussian random vector (that is, each entry g_i of g is an independent standard Gaussian). The quantity $w(T)$ is called the *(Gaussian) mean width* of T. Figure 2.10 shows why the name makes sense.

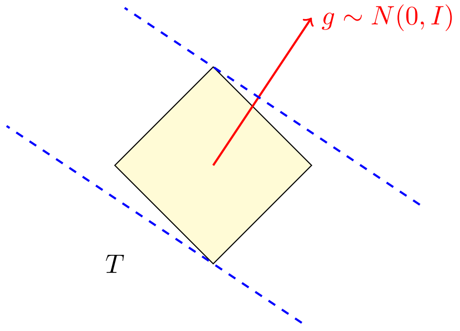


Figure 2.10: Gaussian mean width. The vector $g \sim N(0, I)$ points in a uniformly random direction, and its length (ℓ_2 norm) is very concentrated around $\sqrt{n}$. Having drawn g, the vector $t \in T$ which maximizes $\langle g, t\rangle$ is the one with the largest projection onto g, and the value of $\langle g, t\rangle$ is (proportional to) the length of that projection; as shown above, this is half of the "width" of T in the direction g. Thus, the quantity $\mathbb{E}\sup_{t\in T}\langle g, t\rangle$ can be described as the "mean width" of T, because we average the width of T in direction g, over all directions.

The most basic situation is when $T = \{\sigma_i e_i : i \in [n]\}$ is the collection of (scaled) standard basis vectors. In this case, $\mathbb{E}\sup_{t\in T}\langle g, t\rangle$ is just the expected value of the maximum of n Guassian random variables with variances $\sigma_1^2, \ldots, \sigma_n^2$. We have the following proposition.

Proposition 2.14. *Let $g_i \sim N(0, \sigma_i^2)$, for $i = 1, \ldots, n$, and suppose that $\max_i \sigma_i \leq \sigma$. Then*

$$\mathbb{E}\max_{i\in[n]}|g_i| \leq \sigma\sqrt{2\ln(n)}\cdot(1 + o(1)).$$

Proof. We have

$$\begin{aligned}\mathbb{E}\max_{i\in[n]}|g_i| &= \int_{u=0}^{\infty}\mathbb{P}\left\{\max_{i\in[n]}|g_i| > u\right\}du\\ &\leq A + \frac{2}{\sqrt{2\pi}}\int_{u=A}^{\infty} n\exp\left(\frac{-u^2}{2\sigma^2}\right)du\end{aligned}$$

for any $A \geq \sigma$ (which we will choose shortly). In the above inequality, we have used (2.9) (with the fact that $A \geq \sigma$) and the fact that for every i, $\mathbb{P}\{|g_i| > u\} = 2\mathbb{P}\{g_i > u\}$. We may estimate the integral using (2.10), so

$$\frac{2}{\sqrt{2\pi}}\int_{u=A}^{\infty}\exp\left(\frac{-u^2}{2\sigma^2}\right)du \leq \frac{2\sigma^2}{A\sqrt{2\pi}}\exp\left(-\frac{A^2}{2\sigma^2}\right).$$

Choosing $A = \sigma\sqrt{2\ln(n)}$, we get

$$\mathbb{E}\max_{i\in[n]}|g_i| \le \sigma\sqrt{2\ln(n)} + \frac{\sigma}{\sqrt{\pi\ln(n)}}.$$

$\square$

It is not hard to see that the mean width does not change when we take the convex hull of T:

$$w(T) = w(\operatorname{conv}(T)). \tag{2.12}$$

For example, Proposition 2.14 implies that the mean width of the ℓ_1 ball $B_1^n = \{x \in \mathbb{R}^n : \|x\|_1 \le 1\}$ is

$$w(B_1^n) = w(\operatorname{conv}(\{e_1, e_2, \dots, e_n\})) = w(\{e_1, e_2, \dots, e_n\}) = \Theta\left(\sqrt{\ln(n)}\right).$$

What about other sets T? For several T, $w(T)$ can be computed rather precisely. For example, $w(B_2^n) = \Theta(\sqrt{n})$. For others, it can be much trickier. We wave our hands about a general method, called a *chaining argument,* below.

Let $X_t = \langle g, t\rangle$, so we want to understand $\mathbb{E}\sup_{t\in T} X_t$. A natural approach would be a union bound: if X_t is small with probability $1-p$, then $\sup_{t\in T} X_t$ is small with probability $1 - Np$, where N is the size of (a suitable discretization of) T. However, in many situations, $p > 1/N$ is not small enough to allow for such a union bound. We mustn't give up hope: naive union bounds are often quite wasteful. If $\|s-t\|_2$ is small, then X_t and X_s are highly correlated. Treating X_t and X_s as completely unrelated (or worse) when taking the union bound is leaving something on the table.

A first attempt to take advantage of this is illustrated in Figure 2.11. Suppose that T is "clustered" (with respect to ℓ_2 distance) and write $T = S_1 \cup S_2 \ldots \cup S_\ell$. For each i, pick a representative point $t_i \in S_i$. Consider events of two types. The

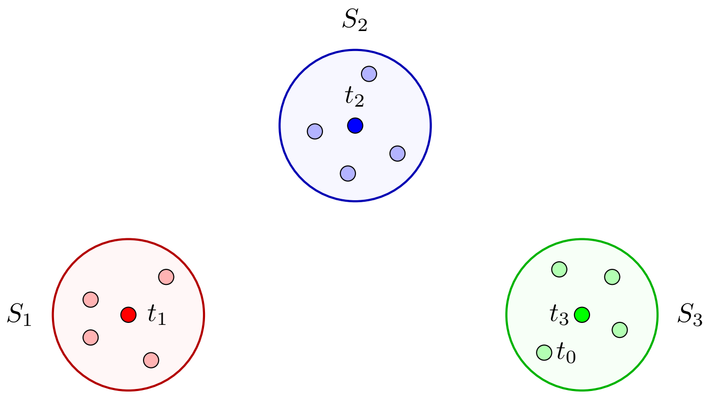


Figure 2.11: First attempt at a chaining argument. The set T can be partitioned into $S_1 \cup S_2 \cup S_3$. Suppose that $X_t = \langle g, t \rangle$ for $t \in T$, and we wish to argue that $\sup_{t \in T} |X_t| \leq M$ with high probability. A naive attempt would be to apply a union bound to the bad events that $|X_t| > M$ for all $t \in T$. A slightly more refined approach is as follows. Consider the bad event that $|X_{t_0} - X_{t_3}| > M/4$. This is very unlikely, much more unlikely than the event that $|X_{t_0}| > M$, because $X_{t_0} - X_{t_3} = \langle g, t_0 - t_3 \rangle \sim N(0, \|t_0 - t_3\|_2^2)$, and $\|t_0 - t_3\|_2$ is very small. Take a union bound over all events of this type, as well as the three events that $|X_{t_i}| > 3M/4$ for $i = 1, 2, 3$. In the favorable case that none of these events occur, we still have $X_t \leq M/4 + 3M/4 = M$ for all $t \in T$, but we have saved a little bit in the union bound. The idea of a *chaining* argument is to iterate this process, and recursively subdivide the sets S_i.

first type of event is that X_{t_i} is small. The second type of event, for $t \in S_i$, is that $|X_t - X_{t_i}|$ is small. Now this has made the situation somewhat better: if ℓ is small, then we can handle a union bound over all events of the first type, because there are not too many of them. On the other hand, there are lots of events of the second type, but they happen with much higher probability because t and t_i are "close."

The idea behind a *chaining argument* is to iterate the first attempt described above. Having made clusters $S_1, \ldots, S_\ell$, we then cluster each of the clusters, and so on. We will return to this argument in Chapter 4, where we will use it to show that Reed-Solomon codes (with random evaluation points) are near-optimally list-decodable with high probability. To put this argument in a little more context, we mention here that such chaining arguments are quite general, and in the case of Gaussian processes X_t, they in fact completely capture $\mathbb{E} \sup_{t \in T} X_t$. More precisely, Talagrand's majorizing measures theorem [104] shows that there is always a chaining argument which can estimate $\mathbb{E}\sup_{t\in T} X_t$, up to constant factors.

2.4.3 Getting to Gaussians

Above, we have outlined many of the wonderful properties about Gaussian random variables. However, this thesis is about coding theory over finite fields. The reader may be wondering how Gaussians (which are real random variables) could possibly come into the picture. There are several tricks we can use to take advantage of the tools above, even working over finite fields. The basic idea is illustrated in Figure 2.12. We will outline a few specific tricks that we need below.

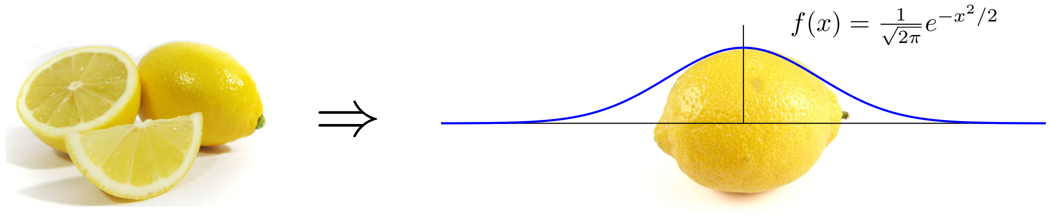


Figure 2.12: When life gives you lemons, turn them into Gaussians.

Our main tools are based on the fact that the sum of independent random variables behaves a lot like Gaussians. The first version that we use is the *Chernoff-Hoeffding bound,* which states that the tail behavior of the sum of independent random variables is very much like that a Gaussian random variable.

Theorem 2.15. *Let $X_1, \ldots, X_m$ be m independent random variables such that for every $i \in [m]$, $X_i \in [a_i, b_i]$, then for the random variable*

$$S = \sum_{i=1}^{m} X_i,$$

and any positive $v \geq 0$, we have

$$\mathbb{P}\left\{|S - \mathbb{E}[S]| \geq v\right\} \leq 2\exp\left(-\frac{2v^2}{\sum_{i=1}^{m}(b_i - a_i)^2}\right).$$

A second way to introduce Gaussians is via *symmetrization* and *comparison* arguments. These arguments simplify expressions like $\mathbb{E}\left\|\sum_{j=1}^n (X_j - \mathbb{E}X_j)\right\|$ for independent random variables X_j taking values in any Banach space. These arguments are standard—see [80], Lemma 6.3 and Equation (4.8), respectively. For completeness (and concreteness), we'll state and prove versions when the X_i are real-valued functions of some set T, and the norm is the L_∞ norm.

Lemma 2.16. *Let $T \subset \mathbb{R}^n$, and let $X_i : T \to \mathbb{R}$ for $i = 1, \ldots, m$ be independent random functions.*

$$\mathbb{E}\sup_{t\in T}\left|\sum_{j\in[n]} (X_j(t) - \mathbb{E}X_t(t))\right| \leq 2\mathbb{E}\sup_{t\in T}\left|\sum_{j\in[n]} \xi_j X_j(t)\right|,$$

where the ξ_i are independent Rademacher random variables (that is, $\xi_i = +1$ with probability $1/2$ and -1 with probability $1/2$).

Proof. Let $\mathcal{C}'$ be an independent copy of $\mathcal{C}$, and let $X'_j(t)$ denote an independent copy of $X_j(t)$. Then,

$$\begin{aligned}
\mathbb{E}_X \sup_{t\in T}\left|\sum_{j\in[n]} (X_j(t) - \mathbb{E}_X X_j(t))\right| &= \mathbb{E}_X \sup_{t\in T}\left|\sum_{j\in[n]} \left(X_j(t) - \mathbb{E}_{\mathcal{C}} X_j(t) - \mathbb{E}_{X'}\left[X'_j(t) - \mathbb{E}_{X'}X'_j(t)\right]\right)\right| \\
&\leq \mathbb{E}_X \mathbb{E}_{X'} \sup_{t\in T}\left|\sum_{j\in[n]} \left(X_j(t) - X'_j(t)\right)\right| \\
&= \mathbb{E}_\xi \mathbb{E}_X \mathbb{E}_{X'} \sup_{t\in T}\left|\sum_{j\in[n]} \xi_j (X_j(t) - X'_j(t))\right| \\
&\leq 2\mathbb{E}_\xi \mathbb{E}_X \sup_{t\in T}\left|\sum_{j\in[n]} \xi_j X_j(t)\right|.
\end{aligned}$$

Above, we used Jensen's inequality, independence, and the triangle inequality in the second, third, and fourth lines, respectively. □

Next, we replace the Rademacher random variables ξ_j with Gaussian random variables g_j using a comparison argument.

Lemma 2.17. *Let $T \subset \mathbb{R}^n$ be any set, and fix $X_j : T \to \mathbb{R}$. Let $\xi_1, \ldots, \xi_n$ be independent Rademacher random variables, and let $g_1, \ldots, g_n$ be independent standard normal random variables. Then*

$$\mathbb{E}_\xi \sup_{t \in T} \left| \sum_{j \in [n]} \xi_j X_j(t) \right| \leq \sqrt{\frac{\pi}{2}} \mathbb{E}_g \sup_{t \in T} \left| \sum_{j \in [n]} g_j X_j(t) \right|.$$

Proof. We have

$$\begin{aligned} \mathbb{E}_g \sup_{t \in T} \left| \sum_{j \in [n]} g_j X_j(t) \right| &= \mathbb{E}_g \mathbb{E}_\xi \sup_{t \in T} \left| \sum_{j \in [n]} \xi_j |g_j| X_j(t) \right| \\ &\geq \mathbb{E}_\xi \sup_{t \in T} \left| \sum_{j \in [n]} \xi_j \mathbb{E}_g |g_j| X_j(t) \right| \qquad \text{by Jensen's inequality} \\ &= \mathbb{E}_\xi \sup_{t \in T} \left| \sum_{j \in [n]} \xi_j \sqrt{\frac{2}{\pi}} X_j(t) \right|. \end{aligned}$$

Above, we used the fact that for a standard normal random variable g_j, $\mathbb{E}|g_j| = \sqrt{2/\pi}$. □

Together, Lemma 2.16 and Lemma 2.17 imply that

$$\mathbb{E} \sup_{t \in T} \left| \sum_{j \in [n]} (X_j(t) - \mathbb{E} X_j(t)) \right| \leq C \mathbb{E} \sup_{t \in T} \left| \sum_{j \in [n]} g_j X_j(t) \right|,$$

for some constant C. This will allow us to use our observations above about the mean width. Indeed, the expression on the right hand side is the Gaussian mean width of the set

$$\{(X_1(t), \ldots, X_n(t)) \, : \, t \in T\} \subset \mathbb{R}^n.$$

When manipulating Gaussian processes of the form (2.11), the following *contraction principle* will also come in handy.

Lemma 2.18 (Corollary 3.17 in [80]). *Let $T \subset \mathbb{R}^n$ be a bounded set, and let $g_1, \ldots, g_n$ be i.i.d. standard normal random variables. Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a contraction (that is,*

$|\varphi(x) - \varphi(y)| \le |x - y|$ for all x, y in $\mathbb{R}$) with $\varphi(0) = 0$. Then for all non-negative convex increasing functions F on $\mathbb{R}_+$,

$$\mathbb{E} F\left(\frac{1}{2}\sup_{t\in T}\left|\sum_i g_i \varphi(t_i)\right|\right) \le \mathbb{E} F\left(2\sup_{t\in T}\left|\sum_{i=1}^n g_i t_i\right|\right).$$

In the case when F is the identity and $\varphi(x) = |x|$, this implies that

$$\mathbb{E}\sup_{t\in T}\left|\sum_{i=1}^n g_i |t_i|\right| \le 4\mathbb{E}\sup_{t\in T}\left|\sum_{i=1}^n g_i t_i\right|. \tag{2.13}$$

2.5 Overview of notation

We conclude this chapter with a brief overview of the notation. We reserve n for the block length of a code, and k for the dimension of a (linear) code. We will use R for the rate of a code. For $x, y \in \mathbb{F}_q^n$, $\delta(x, y)$ will denote relative Hamming distance. For an integer r, $[r]$ will denote the set $[r] = \{1, 2, \ldots, r\} \subset \mathbb{Z}$. Generally, g will denote a Gaussian random variable, and ξ will denote a Rademacher random variable. We will use $\mathbb{E}$ to denote the expectation operator, and $\mathbb{P}\{\cdot\}$ for probabilities. For $x \in \mathbb{R}^n$, $\|x\|_p$ will denote the ℓ_p norm of x:

$$\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}, \qquad \|x\|_\infty = \max_i |x_i|.$$

For vectors x, y in $\mathbb{R}^n$ or $\mathbb{F}_q^n$, $\langle x, y\rangle$ will denote the inner product

$$\langle x, y\rangle = \sum_{i=1}^n x_i y_i.$$

We care more about the dependence on key parameters (like n and ε as n grows large and ε grows small) than on the constant factors. To that end, we will use C or C_i to denote absolute constants, which do not depend on any of the parameters of interest. We will use standard asymptotic notation to hide these constants. Precisely, for functions $f(x), g(x)$, we say $f = O(g)$ as $x \to \infty$ if there is some constant C (independent of the inputs to f and g) so that $f(x) \leq Cg(x)$ for sufficiently large x. Similarly, $f = \Omega(g)$ means there is a constant C so that $f(x) \geq Cg(x)$ for suffiently large x, and $f = \Theta(g)$ means that both $f = O(g)$ and $f = \Omega(g)$. The notation $f = o(g)$ (resp. $f = \omega(g)$) means that for all constants C and for all x_0, there is some $x \geq x_0$ so that $f(x) \leq Cg(x)$ (resp. $f(x) \geq Cg(x)$). We will also occasionally use notation like $f \lesssim g, f \gtrsim g$, and $f \approx g$ to mean $f = O(g), f = \Omega(g), f = \Theta(g)$, respectively. Usually the asymptotics will be in terms of the block size n of a code, which will tend to infinity. In the list-decoding setting, we consider asymptotics as the error rate $\rho = 1 - 1/q - \varepsilon$ as $\varepsilon \to 0$ and (sometimes) $q \to \infty$. In the local-decoding setting we will consider what happens as the rate $R \to 1$. We will make it clear what parameters we consider when each case arises.

CHAPTER 3

List Decoding: small alphabets

In our discussion in Chapter 2 of list decoding, we saw that random codes were nearly optimally list-decodable (Theorem 2.4), and wondered if the same would hold for random *linear* codes, not just for random codes. In this chapter, we return to this question, in the case when q is small (that is, constant).

More precisely, Theorem 2.4 implies that a random q-ary code of rate $\Omega(\varepsilon^2)$ is list decodable up to radius $1 - 1/q - \varepsilon$, with list sizes on the order of $1/\varepsilon^2$. When we tried to extend the argument to random linear codes in Chapter 2, we ended up with Proposition 2.6, which was unsatisfying: the list sizes were on the order of q^{1/ε^2}. This was basically the best we could do until recently. However, in 2013, Cheraghchi, Guruswami, and Velingker [20] made substantial progress. They exploited a connection between list decodability of random linear codes and the *Restricted Isometry Property,* a property of matrices which comes up in compressed sensing (an area of signal processing). Using this connection, they showed that a random linear code of rate $\Omega(\varepsilon^2/\log^3(1/\varepsilon))$ achieves the list decoding properties above, with constant probability. In this chapter, we improve on their result to show that in fact we may take the rate to be $\Omega(\varepsilon^2)$, which is optimal for constant q (up to constant factors), and further that the success probability is $1 - o(1)$, rather than constant. As an added benefit, our proof is quite simple.

Our argument extends beyond random linear codes. We will return to the full generality of our approach in Chapter 5, but here we will state a corollary for *randomly punctured codes* over small alphabet sizes. Using this generalization, we show that randomly punctured Reed-Muller codes have the same list decoding properties as the original codes, even when the rate is improved to a constant.

3.1 Introduction

We recall that a *random linear code* $\mathcal{C} \subset \mathbb{F}_q^n$ is a random subspace of $\mathbb{F}_q^n$. The *dimension* of $\mathcal{C}$, which we will denote by k, is just the dimension of the subspace, and the *rate* is defined to be $R = k/n$. We are interested in the trade-off between the rate R and the list-decoding radius ρ, for polynomial list-sizes $L = \mathrm{poly}(1/\varepsilon)$. In particular, we would like to answer the following question:

> **Question:** Do random linear codes have the same list-decoding radius of random codes? More precisely, are random linear codes of rate $\Omega(\varepsilon^2)$ (probably) (ρ, L)-list-decodable for $\rho = 1 - 1/q - \varepsilon$ and $L = 1/\varepsilon^2$?

As mentioned in the previous chapters, understanding the trade-offs in list decoding is interesting not just for communication, but also for a wide array of applications in complexity theory. List decodable codes can be used for hardness amplification of boolean functions and for constructing hardcore predicates from one-way functions, and they can be used to construct randomness extractors, expanders, and pseudorandom generators [43, 102, 108]. Beyond that, understanding the behavior of linear codes, and in particular random linear codes, is also of interest: decoding a random linear code is related to the problem of learning with errors, a fundamental problem in both learning theory [15, 30] and cryptography [91].

For most of these applications, the parameter regime of interest in when the error rate ρ is very large. We saw in Section 2.2.1 that the largest we can hope for ρ to be is $1 - 1/q$. Thus, we study what happens when we back off just a little bit, to $\rho = 1 - 1/q - \varepsilon$. For the work in this chapter, the computations will work out slightly more nicely if instead we consider $\rho = (1 - {}^1\!/_q)(1 - \varepsilon)$, so we'll do that.

3.1.1 Related work

Random linear codes are a very natural class of codes, and there have been many works devoted to their list-decodability. Recall, we have already made such an attempt in Chapter 2, Proposition 2.6. In order to set the stage, we first recall how we arrived at that result.

In Chapter 2, in the proof of Theorem 2.4, we saw a proof that general random codes are optimally list-decodable. The basic idea was that, for a set Λ of possible messages and any received word z, there is only a very small probability that all of the codewords corresponding to Λ are close to z. This probability is small enough to allow for a union bound over the $q^n \cdot \binom{N}{L}$ choices for Λ and z. However, as we pointed out after the proof of Theorem 2.4, this argument crucially exploits the independence between the encodings of distinct messages. If we begin with a random linear code, then codewords are no longer independent, and the above argument fails. We managed to find away around this in Proposition 2.6: by considering only the linearly independent messages in Λ, we made the argument work, but we got exponentially large list sizes of $q^{\Omega(1/\varepsilon)}$.

This exponential dependence on ε can actually be removed for a *constant* fraction of errors, by a careful analysis of the dependence between codewords corresponding to linearly dependent messages. When ρ is *constant*, rather than tending to $1 - 1/q$, Guruswami, Håstad, and Kopparty [44] show that a random linear code of rate

$1 - H_q(\rho) - C_{\rho,q}/L$ is (ρ, L)-list decodable, where $H_q(x) = x\log_q(q-1) - x\log_q(x) - (1-x)\log_q(1-x)$ is the q-ary entropy. This matches lower bounds of Rudra and Guruswami-Narayanan [49, 95]. However, for $\rho = (1 - {}^{1}/_{q})(1-\varepsilon)$, the constant $C_{\rho,q}$ depends exponentially on ε, and this result quickly degrades.

When $\rho = (1 - {}^{1}/_{q})(1-\varepsilon)$, Proposition 2.6, originally due to [116], gave the best upper bounds for random linear codes with rate $\Omega(\varepsilon^2)$ came until recently. Closing the exponential gap in the list sizes between random linear codes and general random codes was posed by [28]. Some progress was made in 2002 by Guruswami, Håstad, Sudan, and Zuckerman [45], who proved the existence of a binary linear code with rate $\Omega(\varepsilon^2)$ and list size $O(1/\varepsilon^2)$. However, this result only holds for binary codes, and further the proof does not show that most linear codes have this property. Cheraghchi, Guruswami, and Velingker (henceforth CGV) recently made substantial progress on closing the gap between random linear codes and general random codes [20]. Using a connection between list decodability of random linear codes and the Restricted Isometry Property (RIP) from compressed sensing, they proved the following theorem.

Theorem 3.1. (Theorem 12 in [20]) *Let q be a prime power, and let $\varepsilon, \gamma > 0$ be constant parameters. Then for all large enough integers n, a random linear code $\mathcal{C} \subseteq \mathbb{F}_q^n$ of rate R, for some*

$$R \geq C\frac{\varepsilon^2}{\log(1/\gamma)\log^3(q/\varepsilon)\log(q)}$$

is $\left((1 - {}^{1}/_{q})(1-\varepsilon), O(1/\varepsilon^2)\right)$-list decodable with probability at least $1-\gamma$.

It is known that the rate cannot exceed $O(\varepsilon^2)$; this follows from the list decoding capacity theorem, Theorem 2.4, and we stated it in Corollary 2.5. Further, the recent lower bounds of Guruswami and Vadhan [61] and Blinovsky [13, 14] show that the list size L must be at least $\Omega_q(1/\varepsilon^2)$. Thus, Theorem 3.1 has nearly optimal dependence on ε, leaving a polylogarithmic gap.

3.1.2 Contributions of Chapter 3

The extra logarithmic factors in the result of CGV stem from the difficulty in proving that the RIP is likely to hold for randomly subsampled Fourier matrices. Removing these logarithmic factors is considered to be a difficult problem. In this work, we show that while the RIP is a sufficient condition for list decoding, it may not be necessary. We formulate a different sufficient condition for list decodability: while the RIP is about controlling the ℓ_2 norm of Φx, for a matrix Φ and a sparse vector x with $\|x\|_2 = 1$, our sufficient condition amounts to controlling the ℓ_1 norm of Φx with the same conditions on x. Next, we show, using (easy) techniques from high dimensional probability, that this condition does hold with overwhelming probability for random linear codes, with no extra logarithmic dependence on ε. The punchline, and our main result, is the following theorem.

Theorem 3.2. *Let q be a prime power, and fix $\varepsilon > 0$. Then for all large enough integers n, a random linear code $\mathcal{C} \subseteq \mathbb{F}_q^n$ of rate R, for*

$$R \geq C \frac{\varepsilon^2}{\log(q)}$$

is $\left(\left(1 - {}^1\!/_q\right)(1-\varepsilon), O(1/\varepsilon^2)\right)$-list decodable with probability at least $1 - o(1)$. Above, C is an absolute constant.

There are three differences between Theorem 3.1 and Theorem 3.2. First, the dependence on ε in Theorem 3.2 is optimal. Second, the dependence on q is also improved by several log factors, although it is still not quite correct—we will return to this in Chapter 4. Finally, the success probability in Theorem 3.2 is $1 - o(1)$, compared to a constant success probability in Theorem 3.1. As an additional benefit, the proof on Theorem 3.2 is relatively short, while the proof of the RIP result in [20] is quite difficult.

After proving Theorem 3.2, we then generalize our approach to apply to not-necessarily-uniform ensembles of linear codes. We formulate a more general version of Theorem 3.2, and give examples of codes to which it applies. Our main example is linear codes $\mathcal{C}$ of rate $\Omega(\varepsilon^2)$ whose generator matrix is chosen by randomly sampling the columns of a generator matrix of a linear code $\mathcal{C}_0$ of nonconstant rate. Ignoring details about repeating columns, $\mathcal{C}$ can be viewed as randomly punctured version of $\mathcal{C}_0$. Random linear codes fit into this framework when $\mathcal{C}_0$ is taken to be $\mathrm{RM}_q(1, k)$, the q-ary Reed-Muller code of degree one and dimension k. We extend this in a natural way by taking $\mathcal{C}_0 = \mathrm{RM}(r, m)$ to be any (binary) Reed-Muller code.[1] It has recently been shown [40, 76] that $\mathrm{RM}(r, m)$ is list-decodable up to $1/2 - \varepsilon$, with exponential but nontrivial list sizes. However, $\mathrm{RM}(r, m)$ is not a "good" code, in the sense that it does not have constant rate. In the same spirit as our main result, we show that when $\mathrm{RM}(r, m)$ is punctured down to rate $O(\varepsilon^2)$, with high probability the resulting code is list decodable up to radius $1/2 - \varepsilon$ with asymptotically no loss in list size.

3.1.3 Overview of the approach

The CGV proof of Theorem 3.1 proceeds in three steps. The first step is to prove an average-distance Johnson bound, *a la* Theorem 2.8. The second step is a translation of the coding theory setting to a setting suitable for the RIP: a code $\mathcal{C}$ is encoded as a matrix Φ whose columns correspond to codewords of $\mathcal{C}$. This encoding has the property that if Φ had the RIP with good parameters, then $\mathcal{C}$ is

[1] We saw Reed-Muller codes (Definition 2.12) earlier in the context of locally decodable codes. The parameter settings we are interested in for list-decoding are a little different—we will return to these later—but the definition is the same.

list decodable with similarly good parameters. Finally, the last and most technical step is proving that the matrix Φ does indeed have the Restricted Isometry Property with the desired parameters.

In this work, we use the second step from the CGV analysis (the encoding from codes to matrices), but we bypass the other steps. While both the average case Johnson bound and the improved RIP analysis for Fourier matrices are clearly of independent interest, our analysis will be much simpler, and obtains the correct dependence on ε.

3.1.4 Chapter organization

In Section 3.2, we fix some notation and recall some definitions, and also introduce the simplex encoding map from the second step of the CGV analysis. In Section 3.3, we state our sufficient condition and show that it implies list decoding, which is straightforward. We take a detour in Section 3.3.1 to note that the sufficiency of our condition in fact implies the sufficiency of the Restricted Isometry Property directly, providing an alternative proof of Theorem 11 in [20]. In Section 3.4 we prove that our sufficient condition holds, and conclude Theorem 3.2. Finally, in Section 3.5, we discuss the generality of our result, and show that it applies to other ensembles of linear codes.

3.2 A few more definitions

First, we recall some of the notation we'll need. Throughout, we will be interested in linear, q-ary, codes $\mathcal{C}$ with length n and size $|\mathcal{C}| = N$. We use the notation $[q] = \{0, \ldots, q-1\}$, and for a prime power q, $\mathbb{F}_q$ denotes the finite field with q elements. When notationally convenient, we identify $[q]$ with $\mathbb{F}_q$; for our purposes, this identification may be arbitrary. We let $\omega = e^{2\pi \mathbf{i}/q}$ denote the primitive q^{th} root of unity, and we use $\Sigma_L \subset \{0,1\}^N$ to denote the space of L-sparse binary vectors. For two vectors $x, y \in [q]^n$, the *relative Hamming distance* between them is

$$\delta(x,y) = \frac{1}{n} \left|\{i \,:\, x_i \neq y_i\}\right|.$$

Throughout, C_i denotes numerical constants.

We recall Definition 2.3 of list-decodability: a code is list-decodable if any possible received word z does not have too many codewords close to it. For convenience, we repeat the definition here.

Definition 3.3. *A code $\mathcal{C} \subseteq \mathbb{F}_q^n$ is (ρ, L)-list decodable if for all $z \in \mathbb{F}_q^n$,*

$$\left|\{c \in \mathcal{C} \,:\, \delta(c,w) \leq \rho\}\right| \leq L.$$

A code is *linear* if the set $\mathcal{C}$ of codewords is of the form $\mathcal{C} = \{xG \mid x \in \mathbb{F}_q^k\}$, for a $k \times n$ generator matrix G. We say that $\mathcal{C}$ is a *random linear code of rate* R if the image of the generator matrix G is a random subspace of dimension $k = Rn$.

Below, it will be convenient to work with generator matrices G chosen uniformly at random from $\mathbb{F}_q^{k\times n}$, rather than with random linear subspaces of dimension k. These are not the same, as there is a small but positive probability that G chosen this way will not have full rank. However, we observe that

$$\mathbb{P}\left\{\operatorname{rank}(G) < k\right\} = \prod_{r=0}^{k-1}\left(1 - q^{r-n}\right) = 1 - o(1). \tag{3.1}$$

Now suppose that $\mathcal{C}$ is a random linear code of rate $R = k/n$, and $\mathcal{C}'$ is a code with a random $k \times n$ generator matrix G. Let E be the event that $\mathcal{C}$ is (ρ, L)-list decodable for some ρ and L, and let E' be the corresponding event for $\mathcal{C}'$. By symmetry, we have

$$\begin{aligned}\mathbb{P}\left\{E\right\} &= \mathbb{P}\left\{E' \mid \mathrm{rank}(G) = k\right\} \\ &\geq \mathbb{P}\left\{E' \wedge \mathrm{rank}(G) = k\right\} \\ &\geq 1 - \mathbb{P}\left\{\overline{E'}\right\} - \mathbb{P}\left\{\mathrm{rank}(G) < k\right\} \\ &= \mathbb{P}\left\{E'\right\} - o(1),\end{aligned}$$

where we have used (3.1) in the final line. Thus, to prove Theorem 3.2, it suffices to show that $\mathcal{C}'$ is list decodable, and so going forward we will consider a code $\mathcal{C}$ with a random $k \times n$ generator matrix. For notational convenience, we will also treat $\mathcal{C} = \left\{xG \mid x \in \mathbb{F}_q^k\right\}$ as a multi-set, so that in particular we always have $N = |\mathcal{C}| = q^k$. Because by the above analysis the parameter of interest is now k, not $|\mathcal{C}|$, this will be innocuous.

We make use the simplex encoding used in the CGV analysis, which maps the code $\mathcal{C}$ to a complex matrix Φ.

Definition 3.4 (Simplex encoding from [20]).
Define a map $\varphi : [q] \to \mathbb{C}^{q-1}$ by $\varphi(x)(\alpha) = \omega^{x\alpha}$ for $\alpha \in \{1, \ldots, q-1\}$. We extend this map to a map $\varphi : [q]^n \to \mathbb{C}^{n(q-1)}$ in the natural way by concatenation. Further, we extend φ to act on sets $\mathcal{C} \subset [q]^n$: $\varphi(\mathcal{C})$ is the $n(q-1) \times N$ matrix whose columns are $\varphi(c)$ for $c \in \mathcal{C}$.

Notice that when $q = 2$, the simplex encoding $\Phi = \varphi(\mathcal{C})$ is the same as the matrix Φ in our proof of the average-radius, average-distance Johnson bound in Theorem 2.8.

Suppose that $\mathcal{C}$ is a q-ary linear code with random generator matrix $G \in \mathbb{F}_q^{k \times n}$, as above. Consider the $n \times N$ matrix M which has the codewords as columns. The rows of this matrix are independent—each row corresponds to a column t of the random generator matrix G. To sample a row r, we choose $t \in \mathbb{F}_q^k$ uniformly at random (with replacement), and let $r = (\langle t, x\rangle)_{x\in\mathbb{F}_q^k}$. Let T denote the random multiset with elements in $\mathbb{F}_q^k$ consisting of the draws t. To obtain $\Phi = \varphi(\mathcal{C})$, we replace each symbol β of M with its simplex encoding $\varphi(\beta)$, regarded as a column vector. Thus, each row of Φ corresponds to a vector $t \in T$ (a row of the original matrix M, or a column of the generator matrix G), and an index $\alpha \in \{1, \ldots, q-1\}$ (a coordinate of the simplex encoding). We denote this row by $f_{t,\alpha}$.

We use the following facts about the simplex encoding, also from [20]:

1. For $x, y \in [q]^n$,

$$\langle \varphi(x), \varphi(y)\rangle = (q-1)n - q\delta(x,y)n. \tag{3.2}$$

2. If $\mathcal{C}$ is a linear code with a uniformly random generator matrix, the columns of Φ are orthogonal in expectation. That is, for $x, y \in \mathbb{F}_q^n$, indexed by $i, j \in \mathbb{F}_q^k$ respectively, we have

$$\begin{aligned}\mathbb{E}d(x,y) &= \frac{1}{n}\mathbb{E}\sum_{t\in T}\mathbf{1}_{\langle t,i\rangle\neq\langle t,j\rangle}\\ &= \mathbb{P}\left\{\langle t,i\rangle \neq \langle t,j\rangle\right\}\\ &= \begin{cases}1-\frac{1}{q} & i\neq j\\ 0 & i=j\end{cases}\end{aligned}$$

Combined with (3.2), we have

$$\begin{aligned}\mathbb{E}\langle\varphi(x),\varphi(y)\rangle &= (q-1)n - qn\,\mathbb{E}\delta(x,y)\\ &= \begin{cases}(q-1)n & x=y\\ 0 & x\neq y\end{cases}\end{aligned}$$

This implies that

$$\mathbb{E}\|\Phi x\|_2^2 = \sum_{i,j\in[N]} x_i x_j \mathbb{E}\langle\varphi(c_i),\varphi(c_j)\rangle = (q-1)n\|x\|^2. \tag{3.3}$$

3.3 Sufficient conditions for list decodability

Suppose that $\mathcal{C}$ is a linear code as above, and let $\Phi = \varphi(\mathcal{C}) \in \mathbb{C}^{n(q-1)\times N}$ be the complex matrix associated with $\mathcal{C}$ by the simplex encoding. We first translate Definition 2.3 into a linear algebraic statement about Φ. The identity (3.2) implies that $\mathcal{C}$ is $(\rho, L-1)$ list decodable if and only if for all $w \in \mathbb{F}_q^n$, for all sets $\Lambda \subset \mathcal{C}$ with $|\Lambda| = L$, there is at least one codeword $c \in \Lambda$ so that $d(w,c) > \rho$, that is, so that

$$\langle \varphi(c), \varphi(w) \rangle < (q-1)n - q\rho n.$$

Translating the quantifiers into appropriate max's and min's, we observe

Observation 3.5. *A code $\mathcal{C} \in [q]^n$ is $(\rho, L-1)$-list decodable if and only if*

$$\max_{w\in[q]^n} \max_{\Lambda\subset\mathcal{C}, |\Lambda|=L} \min_{c\in\Lambda} \langle \varphi(w), \varphi(c) \rangle < (q-1)n - q\rho n.$$

When $\rho = (1 - 1/q)(1-\varepsilon)$, $\mathcal{C}$ is $(\rho, L-1)$-list decodable if and only if

$$\max_{w\in[q]^n} \max_{\Lambda\subset\mathcal{C}, |\Lambda|=L} \min_{c\in\Lambda} \langle \varphi(w), \varphi(c) \rangle < (q-1)n\varepsilon. \tag{3.4}$$

We seek sufficient conditions for (3.4). Below is the one we will find useful:

Lemma 3.6. *Let $\mathcal{C} \in \mathbb{F}_q^n$ be a q-ary linear code, and let $\Phi = \varphi(\mathcal{C})$ as above. Suppose that*

$$\frac{1}{L} \max_{x\in\Sigma_L} \|\Phi x\|_1 < (q-1)n\varepsilon. \tag{3.5}$$

Then (3.4) holds, and hence $\mathcal{C}$ is $((1 - 1/q)(1-\varepsilon), L-1)$-list decodable.

Proof. We always have

$$\min_{c\in\Lambda} \langle \varphi(w), \varphi(c) \rangle \le \frac{1}{L} \sum_{c\in\Lambda} \langle \varphi(w), \varphi(c) \rangle,$$

so

$$\begin{aligned}
\max_{w\in[q]^n}\max_{|\Lambda|=L}\min_{c\in\Lambda}\langle\varphi(w),\varphi(c)\rangle &\le \frac{1}{L}\max_{w\in[q]^n}\max_{|\Lambda|=L}\sum_{c\in\Lambda}\langle\varphi(w),\varphi(c)\rangle \\
&= \frac{1}{L}\max_{w\in[q]^n}\max_{x\in\Sigma_L}\varphi(w)^T\Phi x \\
&\le \frac{1}{L}\max_{w\in[q]^n}\|\varphi(w)\|_\infty\max_{x\in\Sigma_L}\|\Phi x\|_1 \\
&= \frac{1}{L}\max_{x\in\Sigma_L}\|\Phi x\|_1.
\end{aligned}$$

Thus it suffices to bound the last line by $(q-1)n\varepsilon$. □

Remark 3. *There are two inequalities in the proof above. The first is passing from list-decodability to* average-radius *list-decodability, which we mentioned in Chapter 2. The second is the more serious inequality, and it is here that we give up on large q. Indeed, for $q = 2$, the second inequality in the proof of Lemma 3.6 is an equality, and nothing is lost. As q grows, this becomes more and more lossy.*

3.3.1 Aside: the Restricted Isometry Property

A matrix A has the *Restricted Isometry Property* (RIP) if, for some constant δ and sparsity level s,

$$(1-\delta)\|x\|_2^2 \le \|Ax\|_2^2 \le (1+\delta)\|x\|_2^2$$

for all s-sparse vectors x. The best constant $\delta = \delta(A,k)$ is called the *Restricted Isometry Constant*. The RIP is an important quantity in compressed sensing—an area of signal processing—and much work has gone into understanding it. See [31] for an excellent overview of compressed sensing, including the RIP.

CGV have shown that if $\frac{1}{\sqrt{n(q-1)}}\varphi(\mathcal{C})$ has the RIP with appropriate parameters, $\mathcal{C}$ is list decodable. The proof that the RIP is a sufficient condition follows, after some computations, from an average-distance Johnson bound. While the average-distance Johnson bound is interesting on its own, in this section we note that Lemma 3.6 implies the sufficiency of the RIP immediately. Indeed, by Cauchy-Schwarz,

$$\begin{aligned}\frac{1}{L}\max_{x\in\Sigma_L}\|\Phi x\|_1 &\leq \frac{\sqrt{n(q-1)}}{L}\max_{x\in\Sigma_L}\|\Phi x\|_2\\ &\leq \frac{\sqrt{n(q-1)}}{L}\left(\sqrt{n(q-1)}(1+\delta)\max_{x\in\Sigma_L}\|x\|_2\right)\\ &\leq \frac{n(q-1)}{\sqrt{L}}(1+\delta),\end{aligned}$$

where $\Phi = \varphi(\mathcal{C})$, and $\delta = \delta(\tilde{\Phi}, L)$ is the restricted isometry constant for $\tilde{\Phi} = \frac{1}{\sqrt{n(q-1)}}\Phi$ and sparsity L. By Lemma 3.6, this implies that

$$\frac{\delta+1}{\sqrt{L}} < \varepsilon$$

also implies (3.4), and hence $\left(\left(1-1/q\right)\left(1-\varepsilon\right), L-1\right)$-list decodability. Setting $\delta = 1/2$, we may conclude the following statement:

> For any code $\mathcal{C} \subset [q]^n$, if $\frac{1}{\sqrt{n(q-1)}}\varphi(\mathcal{C})$ has the RIP with contant $1/2$ and sparsity level L, then $\mathcal{C}$ is $\left(\left(1-1/q\right)\left(1-3/2\sqrt{L}\right), L-1\right)$-list decodable.

This precisely recovers Theorem 11 from [20].

3.4 Random linear codes are optimally list-decodable over small alphabets

We wish to show that, when $\Phi = \varphi(\mathcal{C})$ for a random linear code $\mathcal{C}$, (3.5) holds with high probability. Thus, we need to bound $\max_{x\in\Sigma_L}\|\Phi x\|_1$. We write

$$\max_{x\in\Sigma_L}\|\Phi x\|_1 \leq \max_{x\in\Sigma_L}\mathbb{E}\|\Phi x\|_1 + \max_{x\in\Sigma_L}\left|\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1\right|, \tag{3.6}$$

and we will bound each term separately. First, we observe that $\mathbb{E}\|\Phi x\|_1$ is correct.

Lemma 3.7. *Let $\mathcal{C} \subset \mathbb{F}_q^n$ be a linear q-ary code with a random generator matrix. Let $\Phi = \varphi(\mathcal{C})$ as above. Then for any $x \in \Sigma_L$,*

$$\frac{1}{L}\mathbb{E}\|\Phi x\|_1 \le \frac{n(q-1)}{\sqrt{L}}.$$

Proof. The proof is a straighforward consequence of (3.3). For any $x \in \Sigma_L$, we have

$$\begin{aligned}\mathbb{E}\|\Phi x\|_1 &\le \sqrt{n(q-1)}\mathbb{E}\|\Phi x\|_2 \\ &\le \sqrt{n(q-1)}\left(\mathbb{E}\|\Phi x\|_2^2\right)^{1/2} \\ &= n(q-1)\sqrt{L}\end{aligned}$$

using (3.3) and the fact that $\|x\|_2 = \sqrt{L}$. □

Next, we control the deviation of $\|\Phi x\|_1$ from $\mathbb{E}\|\Phi x\|_1$, uniformly over $x \in \Sigma_L$. We do not require the vectors t_j be drawn uniformly at random anymore, so long as they are selected independently.

Lemma 3.8. *Let $\mathcal{C} \subset \mathbb{F}_q^n$ be q-ary linear code, so that the columns $t_1, \ldots, t_n$ of the generator matrix are independent. Then*

$$\frac{1}{L}\mathbb{E}\max_{x\in\Sigma_L}\left|\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1\right| \le C_0(q-1)\sqrt{n\ln(N)}$$

with probability $1 - 1/\mathrm{poly}(N)$, for an absolute constant C_0.

Remark 4. *As noted above, we do not make any assumptions on the distribution of the vectors $t_1, \ldots, t_n$, other than that they are chosen independently. In fact, we do not even require the code to be linear—it is enough for the vectors $v_i = (c(i))_{c\in\mathcal{C}} \in [q]^N$ to be independent. However, as we only consider linear codes in this work, we stick with our statement in order to keep the notation consistent.*

As a warm-up to the proof, which involves a few too many symbols, consider first the case when $q = 2$, and suppose that we wish to succeed with constant probability. Then the rows f_t of Φ are rows of the Hadamard matrix, chosen independently. By standard symmetrization and comparison arguments (as we saw in Section 2.4, and which we will make more precise below), it suffices to bound

$$\begin{aligned}
\frac{1}{L}\mathbb{E}\max_{x\in\Sigma_L}\sum_{t\in T} g_t\left\langle f_t, x\right\rangle &= \frac{1}{L}\mathbb{E}\max_{x\in\Sigma_L}\left\langle g, \Phi x\right\rangle \\
&\leq \mathbb{E}\max_{x\in B_1^N}\left\langle g, \Phi x\right\rangle \\
&= \mathbb{E}\max_{y\in\Phi B_1^N}\left\langle g, y\right\rangle,
\end{aligned}$$

where above $g = (g_1, g_2, \ldots, g_n)$ is a vector of i.i.d. standard normal random variables, and B_1^N denotes the ℓ_1 ball in $\mathbb{R}^N$. The last line is the *Gaussian mean width* of ΦB_1^N, which we discussed in Section 2.4 (Equation (2.11)). Fortunately, it is easy to estimate the mean width of ΦB_1^N, which is a polytope contained in the convex hull of $\pm\varphi(c)$ for $c \in \mathcal{C}$, (that is, the columns of Φ and their opposites). As in (2.12), taking convex hulls does not change the mean width, and so

$$\begin{aligned}
\mathbb{E}\max_{y\in\Phi B_1^N}\left\langle g, y\right\rangle &= \mathbb{E}\max_{c\in\mathcal{C}}\left\langle g, \varphi(c)\right\rangle \\
&\leq 3\sqrt{\log|\mathcal{C}|}\sqrt{\mathbb{E}\left\langle g, \varphi(c)\right\rangle^2} \\
&= 3\|c\|_2\sqrt{\log(N)} \\
&= 3\sqrt{n\log(N)}
\end{aligned}$$

which is what we wanted. Above, we used Fact 2.13 that $\langle g, \varphi(c)\rangle \sim N(0, \|\varphi(c)\|_2^2)$, and then Proposition 2.14.

For general q and failure probability $o(1)$, there is slightly more notation, but the proof idea is the same. We will need the following bound on moments of maxima of Gaussian random variables.

Lemma 3.9. *Let $X_1, \ldots, X_N$ be standard normal random variables (not necessarily independent). Then*

$$\left(\mathbb{E} \max_{i \leq N} |X_i|^p\right)^{1/p} \leq C_1 N^{1/p} \sqrt{p}$$

for some absolute constant C_1.

We remark that while Lemma 3.9 is clearly suboptimal for small p (compare to the bound we got for $p = 1$ above), we will apply it with $p \sim \ln(N)$ and this will give us the desired results.

Proof. Let $Z = \max_{i \leq N} |X_i|$. Then

$$\mathbb{P}\left\{Z > s\right\} \leq N \exp(-s^2/2)$$

for $s \geq 1$. Integrating,

$$\begin{aligned}
\mathbb{E}|Z|^p &= \int \mathbb{P}\left\{Z^p > s\right\} ds \\
&= \int \mathbb{P}\left\{Z^p > t^p\right\} pt^{p-1}\, dt \\
&\leq 1 + N \int_1^\infty \exp(-t^2/2) pt^{p-1}\, dt \\
&\leq 1 + Np2^{p/2}\Gamma(p/2) \\
&\leq 1 + (Np)\left(p^{p/2}\right).
\end{aligned}$$

Thus,

$$(\mathbb{E}|Z|^p)^{1/p} \leq C_1 N^{1/p} \sqrt{p}.$$

for some absolute constant C_1. □

Now we may prove the lemma.

Proof of Lemma 3.8. We recall the notation from the facts in Section 3.2: the rows of Φ are $f_{t,\alpha}$ for $t \in T$, where T is a random multiset of size n with elements chosen independently from $\mathbb{F}_q^d$, and $\alpha \in \mathbb{F}_q^*$.

To control the largest deviation of $\|\Phi x\|_1$ from its expectation, we will control the p^{th} moments of this deviation—eventually we will choose $p \sim \ln(N)$. By a symmetrization argument followed by a comparison principle (Lemma 6.3 and Equation (4.8), respectively, in [80]), for any $p \geq 1$,

$$\begin{aligned}
&\mathbb{E}\max_{x\in\Sigma_L} |\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1|^p \\
&\qquad = \mathbb{E}\max_{x\in\Sigma_L}\left|\sum_{t\in T}\sum_{\alpha\in\mathbb{F}_q^*}\left(|\langle f_{t,\alpha}, x\rangle| - \mathbb{E}|\langle f_{t,\alpha}, x\rangle|\right)\right|^p \\
&\qquad \leq C_2\mathbb{E}_T\mathbb{E}_g\max_{x\in\Sigma_L}\left|\sum_{t\in T} g_t \sum_{\alpha\in\mathbb{F}_q^*}|\langle f_{t,\alpha}, x\rangle|\right|^p \\
&\qquad \leq C_2\mathbb{E}_T\mathbb{E}_g\max_{x\in\Sigma_L}\left|(q-1)\max_{\alpha\in\mathbb{F}_q^*}\sum_{t\in T} g_t|\langle f_{t,\alpha}, x\rangle|\right|^p \\
&\qquad \leq C_2 4^p (q-1)^p \mathbb{E}_T\mathbb{E}_g\max_{x\in\Sigma_L}\max_{\alpha\in\mathbb{F}_q^*}\left|\sum_{t\in T} g_t\langle f_{t,\alpha}, x\rangle\right|^p, \qquad (3.7)
\end{aligned}$$

where the g_t are i.i.d. standard normal random variables, and we dropped the absolute values at the cost of a factor of four by a contraction principle (Equation (2.13)). Above, we used the independence of the vectors $f_{t,\alpha}$ for a fixed α to apply the symmetrization.

For fixed α, let Φ_α denote Φ restricted to the rows $f_{t,\alpha}$ that are indexed by α. Similarly, for a column $\varphi(c)$ of Φ, let $\varphi(c)_\alpha$ denote the restriction of that column to the rows indexed by α. Conditioning on T and fixing $\alpha \in \mathbb{F}_q^*$, let

$$X(x,\alpha) := \sum_{t\in T} g_t\langle f_{t,\alpha}, x\rangle = \langle g, \Phi_\alpha x\rangle .$$

Let B_1^N denote the ℓ_1 ball in $\mathbb{R}^N$. Since $\Sigma_L \subset LB_1^N$, we have

$$\Phi_\alpha(\Sigma_L) \subset L\Phi_\alpha(B_1^N) = \operatorname{conv}\{\pm L\varphi(c)_\alpha : c \in \mathcal{C}\}.$$

Thus, we have

$$\begin{aligned}
&\mathbb{E}_g \max_{x\in\Sigma_L} \max_{\alpha\in\mathbb{F}_q^*} |X(x,\alpha)|^p \\
&\qquad = \mathbb{E}_g \max_{y\in\Phi_\alpha\Sigma_L} \max_{\alpha\in\mathbb{F}_q^*} |\langle g, y\rangle|^p \\
&\qquad \le L^p\, \mathbb{E}_g \max_{\pm c\in\mathcal{C}} \max_{\alpha\in\mathbb{F}_q^*} |\langle g, \varphi(c)_\alpha\rangle|^p, \qquad (3.8)
\end{aligned}$$

using the fact that $\max_{x\in\mathrm{conv}(S)} F(x) = \max_{x\in S} F(x)$ for any convex function F. Using Lemma 3.9, and the fact that $\langle g, \varphi(c)_\alpha\rangle$ is Gaussian with variance $\|\varphi(c)_\alpha\|_2^2 = n$,

$$\begin{aligned}
&L^p\, \mathbb{E}_g \max_{\pm c\in\mathcal{C}} \max_{\alpha\in\mathbb{F}_q^*} |\langle g, \varphi(c)_\alpha\rangle|^p \\
&\qquad \le \left(C_1\, L\, \sqrt{np}(2N(q-1))^{1/p}\right)^p. \qquad (3.9)
\end{aligned}$$

Together, (3.7), (3.8), and (3.9) imply

$$\begin{aligned}
&\mathbb{E} \max_{x\in\Sigma_L} |\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1|^p \\
&\qquad \le C_2 4^p (q-1)^p \mathbb{E}_T \left(C_1 L\sqrt{np}(2N(q-1))^{1/p}\right)^p \\
&\qquad \le \left(4C_2^{1/p} C_1 (q-1)^{(1+1/p)} L\sqrt{np}(2N)^{1/p}\right)^p \\
&\qquad =: Q(p)^p.
\end{aligned}$$

Finally, we set $p = \ln(N)$, so we have

$$Q(\ln(N)) \le C_3(q-1)L\sqrt{n\ln(N)},$$

for an another constant C_3. Then Markov's inequality implies

$$\mathbb{P}\left\{\max_{x\in\Sigma_L} |\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1| > eQ(\ln(N))\right\} \le \frac{1}{N}.$$

We conclude that with probability at least $1 - o(1)$,

$$\frac{1}{L} \max_{x\in\Sigma_L} |\|\Phi x\|_1 - \mathbb{E}\|\Phi x\|_1| \le C_0(q-1)\sqrt{n\ln(N)},$$

for $C_0 = eC_3$. □

Now we may prove Theorem 3.2.

Proof of Theorem 3.2. Lemmas 3.7 and 3.8, along with (3.6), imply that

$$\frac{1}{L}\max_{x\in\Sigma_L}\|\Phi x\|_1 \le \frac{n(q-1)}{\sqrt{L}} + C_0(q-1)\sqrt{n\ln(N)}$$

with probability $1-o(1)$. Thus, if

$$(q-1)\left(\frac{n}{\sqrt{L}} + C_0\sqrt{n\ln(N)}\right) < (q-1)n\varepsilon \tag{3.10}$$

holds, the condition (3.5) also holds with probability $1-o(1)$. Setting $L = \left({}^{2}\!/\!{}_{\varepsilon}\right)^2$ and $n = \frac{4C_0^2\ln(N)}{\varepsilon^2}$ satisfies (3.10), so Lemma 3.6 implies that $\mathcal{C}$ is $\left(\left(1-{}^{1}\!/\!{}_{q}\right)(1-\varepsilon), 4/\varepsilon^2\right)$-list decodable, with k equal to

$$\log_q(N) = \frac{n\varepsilon^2}{(2C_0)^2\ln(q)}.$$

With the remarks from Section 3.2 following the definition of random linear codes, this concludes the proof. □

3.5 Generalization to randomly punctured codes

In this section, we show that our approach above applies not just to random linear codes, but to many ensembles. In our proof of Theorem 3.2, we required only that the expectation of $\|\Phi x\|_1$ be about right, and that the columns of the generator matrix were chosen independently, so that Lemma 3.8 implies concentration. The fact that $\|\Phi x\|_1$ was about right followed from the condition (3.3), which required that, within sets $\Lambda \subset \mathcal{C}$ of size L, the average pairwise distance is, in expectation, large. We formalize this observation in the following lemma, which can be substituted for Lemma 3.7.

Lemma 3.10. *Let $\mathcal{C} = \{c_1, \ldots, c_N\} \subset [q]^n$ be a (not necessarily uniformly) random code*[2] *so that for any $\Lambda \subset [N]$ with $|\Lambda| = L$,*

$$\frac{1}{\binom{L}{2}} \mathbb{E} \sum_{i<j\in\Lambda} \delta(c_i, c_j) \geq (1 - 1/q)(1 - \eta). \tag{3.11}$$

Then for all $x \in \Sigma_L$,

$$\frac{1}{L}\mathbb{E}\|\varphi(\mathcal{C})x\|_1 \leq n(q-1)\sqrt{\frac{1}{L} + \frac{2\eta\binom{L}{2}}{L^2}}.$$

Proof. Fix $x \in \Sigma_L$, and let Λ denote the support of x. Then, using (3.2),

$$\begin{aligned}
\frac{1}{L}\mathbb{E}\|\varphi(\mathcal{C})x\|_1 &\leq \frac{\sqrt{n(q-1)}}{L}\left(\mathbb{E}\|\varphi(\mathcal{C})\|_2^2\right)^{1/2} \\
&= \frac{\sqrt{n(q-1)}}{L}\left(\mathbb{E}\sum_{i,j\in\Lambda}\langle\varphi(c_i), \varphi(c_j)\rangle\right)^{1/2} \\
&= \frac{\sqrt{n(q-1)}}{L}\left(\mathbb{E}\sum_{i,j\in\Lambda}(q-1)n - qn\,\delta(c_i, c_j)\right)^{1/2} \\
&\leq \frac{\sqrt{n(q-1)}}{L}\left(L(q-1)n + 2\binom{L}{2}n(q-1)\eta\right)^{1/2} \\
&= n(q-1)\sqrt{\frac{1}{L} + \frac{2\eta\binom{L}{2}}{L^2}},
\end{aligned}$$

as claimed. $\square$

Thus, we may prove a statement analogous to Theorem 3.2 about any distribution on linear codes whose generator matix has independent columns, which satisfies (3.11). Where might we find such distributions? Notice that if the expectation is removed, (3.11) is precisely what we needed for the average-distance Johnson bound (Theorem 2.8 in this thesis, or Theorem 8 in [20]) to work, and so any code $\mathcal{C}_0$ to which the average-distance Johnson bound applies attains (3.11). However, such a

[2]$\mathcal{C}$ need not be linear, so we switch the alphabet from $\mathbb{F}_q$ to $[q]$ to emphasize that the field structure is not important.

code $\mathcal{C}_0$ might have substantially suboptimal rate—we can improve the rate, and still satisfy (3.11), by forming generator matrix for a new code $\mathcal{C}$ from a random set of columns of the generator matrix of $\mathcal{C}_0$.

Definition 3.11. *Fix a code $\mathcal{C}_0 \subset [q]^{n_0}$, and define an ensemble $\mathcal{C} = \mathcal{C}(\mathcal{C}_0) \subset [q]^n$ as follows. To draw $\mathcal{C}$, choose a random multiset $T = \{t_1, \ldots, t_n\}$ of size n by drawing elements of $[n_0]$ independently with replacement. Then let*

$$\mathcal{C} = \{(c_{t_1}, \ldots, c_{t_n}) \,:\, c \in \mathcal{C}_0\} .$$

We will call $\mathcal{C}$ a random sampled *version of $\mathcal{C}_0$, with block length n.*

Remark 5 (Sampling vs. Puncturing)**.** *We note that the operation of randomly* sampling *a code (a term we just made up) is very similar to that of randomly* puncturing *a code (a term with a long and illustrious history). The only difference is that we sample with replacement, while a randomly punctured code can be viewed as a code where the sampling is done without replacement. These two distributions are basically the same in the parameter regimes we consider: as such we will (and have) occasionally abuse(d) language and refer(ed) to the operation in Definition 3.11 "puncturing."*

We also notice that since all we need is independence of the symbols, the results would follow if we retained each coordinate in $[n_0]$ independently with probability n/n_0. This would actually be a punctured code, except that the length would now be a random variable, with expected length n.

Replacing Lemma 3.7 with Lemma 3.10 in the proof of Theorem 3.2 immediately implies that randomly sampled codes are list decodable with high probability, if the original code $\mathcal{C}$ has good average distance.

Corollary 3.12. *Let $\mathcal{C}_0 = \{c_1, \ldots, c_N\} \subset \mathbb{F}_q^{n_0}$ be any linear code with*

$$\frac{1}{\binom{L}{2}} \sum_{i<j\in\Lambda} \delta(c_i, c_j) \geq \left(1 - \frac{1}{q}\right)(1-\eta)$$

for all sets $\Lambda \subset [N]$ of size L. Set

$$\varepsilon^2 := 4\left(\frac{1}{L} + \eta\left(1 - \frac{1}{L}\right)\right).$$

There is some $R = \Omega(\varepsilon^2)$ so that if $\mathcal{C} = \mathcal{C}(\mathcal{C}_0)$ is as in Definition 3.11 with rate R, then $\mathcal{C}$ is $((1 - 1/q)(1-\varepsilon), L-1)$-list decodable with probability $1 - o(1)$.

The average-distance Johnson bound implies that if $\mathcal{C}$ is as in the statement of Corollary 3.12, then the original code $\mathcal{C}_0$ is $((1 - 1/q)(1-\varepsilon), O(1/\varepsilon^2))$-list decodable, for ε as above. Thus, Corollary 3.12 implies that $\mathcal{C}$ has the same list decodability properties as $\mathcal{C}_0$, but perhaps a much better rate.

As a example of this construction, consider the family of (binary) degree r Reed-Muller codes, $\mathrm{RM}(r,m) \subset \mathbb{F}_2^m$. $\mathrm{RM}(r,m)$ can be viewed as the set of degree r, m-variate polynomials over $\mathbb{F}_2$. It is easily checked that $\mathrm{RM}(r,m)$ is a linear code of dimension $k = 1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}$ and minimum relative distance 2^{-r}. The random sampling $\mathcal{C}$ of $\mathrm{RM}(r,m)$ is a natural class of codes: decoding $\mathcal{C}$ is equivalent to learning a degree r polynomial over $\mathbb{F}_2^m$ from random samples, in the presence of (worst case) noise.

We cannot hope for short list sizes in this case, but we can hope for nontrivial ones. Kaufman, Lovett, and Porat [76] have given tight asymptotic bounds on the list sizes for $\mathrm{RM}(r,m)$ for all radii, and in particular have shown that $\mathrm{RM}(r,m)$ is list decodable up to $1/2 - \varepsilon$ with list sizes on the order of $\varepsilon^{\Theta_r(m^{r-1})}$. As $|\mathrm{RM}(r,m)|$ is exponential in m^r, this is a nontrivial bound. We will show that randomly sampled Reed-Muller codes, with rate $\Omega(\varepsilon^2)$, have basically the same list decoding parameters as their un-punctured progenitors.

Proposition 3.13. *Let $\mathcal{C} = \mathcal{C}(\mathrm{RM}(r,m))$ be as in Definition 3.11, with rate $O(\varepsilon^2)$. Then $\mathcal{C}$ is $(1/2(1-\varepsilon), L(\varepsilon))$-list decodable with probability $1 - o(1)$, where*

$$L(\varepsilon) = \left(\frac{1}{\varepsilon}\right)^{O_r(m^{r-1})},$$

where O_r hides constants depending only on r.

Proof. We aim to find η so that (3.11) is satisfied. As usual, let $N = |\mathrm{RM}(r,m)|$. We borrow a computation from the proof of Lemma 6 in [20]. Let $A = A(\varepsilon)$ be the number of codewords of $\mathrm{RM}(r,m)$ with relative weight at most $1/2(1-\varepsilon^2)$. Let $L = A/\varepsilon^2$ and choose a set $\Lambda \subset [N]$ of size L. By linearity, for each codeword c_i with $i \in \Lambda$, there are at most $A-1$ codewords c_j within $1/2(1-\varepsilon^2)$ of c_i, out of $L-1$ choices for c_j. Thus, the sum of the relative distances over $j \neq i$ is at most $(L-A) \cdot 1/2(1-\varepsilon^2)$. This implies

$$\begin{aligned}
\frac{1}{\binom{L}{2}} \sum_{i<j\in\Lambda} \delta(c_i,c_j) &\geq \frac{L-A}{L-1}\left(\frac{1}{2}(1-\varepsilon^2)\right) \\
&= \left(1 - \frac{A-1}{L-1}\right)\left(\frac{1}{2}(1-\varepsilon^2)\right) \\
&\geq \frac{1}{2}\left(1 - \varepsilon^2 - \frac{A-1}{L-1}\right) \\
&= \frac{1}{2}\left(1 - O(\varepsilon^2)\right),
\end{aligned}$$

using the choice $L = A/\varepsilon^2$ in the final line. Thus, in Corollary 3.12, we may take $\eta = O(\varepsilon^2)$.

We conclude that the randomly punctured code $\mathcal{C}(\mathrm{RM}(r,m))$ of rate $O(\varepsilon^2)$ is $(1/2(1-\varepsilon), L-1)$ list decodable, with list size L on the order of A/ε^2. It remains to estimate $A = A(\varepsilon)$. It is shown in [76] that

$$A = A(\varepsilon) = \left(\frac{1}{\varepsilon}\right)^{\Theta_r(m^{r-1})},$$

which finishes the proof. □

Another popular ensemble of linear codes is the *Wozencraft ensemble* [74, 109], which encodes an element $x \in \mathbb{F}_{q^k}$ as $(x, \alpha_1 x, \alpha_2 x, \ldots, \alpha_r x)$ for uniformly random $\alpha_j \in \mathbb{F}_{2^k}$. In this case, the symbols within a codeword are not all independent, so Lemma 3.10 does not apply. However, the techniques above extend immediately to imply that a code from this ensemble (with $r \sim k/\varepsilon^2$) is $((1 - 1/q)(1 - \varepsilon), O(1/\varepsilon))$-list decodable with rate ε^2/k. (Previously, the only known result about the list decodability of the Wozencraft ensemble follows from the Johnson bound, which implies a rate on the order of ε^4 for the same radius, so for very small ε this is better). It would be interesting to see if this argument could be modified to obtain constant rate for the Wozencraft ensemble, or for other ensembles of linear codes.

3.6 Conclusion

In this chapter, we have shown that a random linear code of rate $\Omega\left(\frac{\varepsilon^2}{\log(q)}\right)$ is $((1 - 1/q)(1 - \varepsilon), O(1/\varepsilon))$-list decodable with probability $1 - o(1)$. Our result improves the results of [20] in three ways. First, we remove the logarithmic dependence on ε in the rate, achieving the optimal dependence on ε. Second, it improves the dependence on q in the rate, from $1/\log^4(q)$ to $1/\log(q)$. Finally, we show that list decodability holds with probability $1 - o(1)$, rather than with constant probability. As an added benefit, the proof is relatively short and straightforward. For constant alphabet sizes q, this closes a question asked by [28]: Random linear codes are (up to constant factors, with high probability) optimally list-decodable. For $q > 2$, this work is the first to establish even existence of such codes.

We also extended our argument to randomly punctured codes (modulo Remark 5). As an example, we considered Reed-Muller codes, and showed that they retain their combinatorial list decoding properties with high probability when randomly punctured down to constant rate.

However, some questions remain. While these results are optimal for constant q, they are not correct if q is allowed to grow with ε. We recall Corollary 2.5, which gave upper bounds on the rate R when $\rho = (1 - 1/q)(1 - \varepsilon)$: we had

$$R \leq 1 - H_q(1 - 1/q - \varepsilon) \leq \min \left\{ \varepsilon, \frac{q\varepsilon^2}{\log(q)} \right\}.$$

In particular, our dependence on q in Theorem 3.2 is off by a factor of q. Additionally, when q is large, say, larger than $1/\varepsilon^2$, then our quadratic dependence on ε is not correct. In Chapter 4, we will address these questions, and see how to extend the argument in this chapter to large alphabet sizes.

Acknowledgements

The results in this chapter first appeared as [111]. I thank Atri Rudra and Martin Strauss for very helpful conversations when working on this.

CHAPTER 4

List Decoding: large alphabets and Reed-Solomon codes

When we last left our heroes (Alice and Bob) in Chapter 3, they had been able (combinatorially speaking) to communicate using a random linear code, provided the alphabet size was constant. However, the chapter ended on a cliff-hanger of sorts. If Alice and Bob want to communicate over a larger alphabet, say, $q \gg 1/\varepsilon^2$, the results of Chapter 3 wouldn't help much. There are several reasons to consider larger alphabet sizes. In addition to the complexity-theoretic applications mentioned in Chapter 2, our primary motivation for the work in the current chapter was the Reed-Solomon codes of Definition 2.1. Because the symbols of Reed-Solomon codes are indexed by the evaluation points $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$, we must have $q \geq n$ to define them; in particular, q cannot be constant if we are going to allow $n \to \infty$. As a final piece of motivation, we like to resolve cliff-hangers.

4.1 Introduction

We will continue our exploration of list decoding, this time over larger alphabet sizes. We recall that our goal is to understand list-decodability in the parameter regime where $\rho = 1 - 1/q - \varepsilon$ is very large. The optimal rate to correct ρ fraction of errors is given by Theorem 2.4 and Corollary 2.5:

$$R^*(q,\varepsilon) := 1 - H_q(1 - 1/q - \varepsilon) \le \min\left\{\varepsilon, \frac{q\varepsilon^2}{\log(q)}\right\}.$$

As we mentioned in Chapter 2, for complexity applications it is often enough to design a code with rate $\Omega(R^*(q,\varepsilon))$ with the same error correction capability.

In Chapter 3, we got the right dependence on ε, when q was constant. In this chapter, we will also try to get the right dependence on q. That is, we seek to correct a $\rho = 1 - 1/q - \varepsilon$ fraction of errors, with rate $\Omega(R^*(q,\varepsilon))$. The quest for such codes comes in two flavors: one can ask about the list decodability of a specific family of codes, or one can ask for the most general conditions which guarantee list decodability. The results in this chapter address open problems of both flavors, discussed more below.

Specific families of codes with near-optimal rate. There has been significant effort directed at designing *efficiently*-decodable codes with optimal rate. This has led to the study of very specific families of list-decodable codes. The first non-trivial progress towards this goal was due to work of Sudan [101] and Guruswami-Sudan [57] who showed that *Reed-Solomon* (RS) codes [1] can be list decoded efficiently from $1-\varepsilon$ fraction of errors with rate ε^2. This matches the Johnson bound (Theorem 2.7).

The work of Guruswami and Sudan held the record for seven years, during which time RS codes enjoyed the best known tradeoff between rate and fraction of correctable errors. However, Parvaresh and Vardy showed that a variant of Reed-Solomon codes can beat the Johnson bound [87]. This was then improved by Guruswami and Rudra who achieved the optimal rate of ε with Folded Reed-Solomon codes [51]. Since then this optimal rate result has been achieved with other

[1]Recall Definition 2.1: an RS code encodes a low-degree univariate polynomial f over $\mathbb{F}_q$ as a list of evaluations $(f(\alpha_1), \ldots, f(\alpha_n))$ for a predetermined set of $n \le q$ *evaluation points* in $\mathbb{F}_q$.

codes: derivative codes [62], multiplicity codes [78], folded Algebraic Geometric (AG) codes [63] as well as *subcodes* of RS and AG codes [64]. There has also been a lot of recent work on reducing the runtime and list size for folded RS codes [23, 48, 62].

Even though many of the recent developments on list decoding are based on Reed-Solomon codes, there has been no non-trivial progress on the list decodability of Reed-Solomon codes themselves since the work of Guruswami-Sudan. This is true even if we only ask for combinatorial (not necessarily efficient) decoding guarantees, and even for rates only slightly beyond the Johnson bound. The question of whether or not Reed-Solomon codes can be list decoded beyond the Johnson bound was our main motivation for this work:

Question 4.1. *Are there Reed-Solomon codes which can be combinatorially list decoded from a $1-\varepsilon$ fraction of errors, with rate $\omega\left(\varepsilon^2\right)$?*

This question, which has been well-studied, is interesting for several reasons. First, Reed-Solomon codes themselves are arguably the most well-studied codes in the literature. Secondly, there are complexity applications where one needs to be able to list decode Reed-Solomon codes in particular: e.g. the average-case hardness of the permanent [17]. Finally, the Johnson bound is a natural barrier and it is an interesting to ask whether it can be overcome by natural codes.[2]

There have been some indications that Reed-Solomon codes might *not* be list decodable beyond the Johnson bound. Guruswami and Rudra [50] showed that for a generalization of list decoding called list recovery, the Johnson bound indeed gives the correct answer for RS codes. Further, Ben-Sasson et al. [12] showed that for RS code where the evaluation set is all of $\mathbb{F}_q$, the correct answer is close to the

[2]We note that it is easy to come up with codes that have artificially small distance and hence can beat the Johnson bound; it is also known that Reed-Muller codes (Definition 2.12) can be list decoded beyond the Johnson bound [39, 40].

Johnson bound. In particular, they show that to correct $1-\varepsilon$ fraction of errors with polynomial list sizes, the RS code with $\mathbb{F}_q$ as its evaluation points cannot have rate better than $\varepsilon^{2-\gamma}$ for any constant $\gamma > 0$. However, this result leaves open the possibility that one could choose the evaluation points carefully and obtain an RS code which can be combinatorially list decoded significantly beyond the Johnson bound.

Resolving the above possibility has been open since [56]: see e.g. [43, 94, 108] for explicit formulations of this question.

Large families of codes with near-optimal rate. While the work on list decodability of specific families of codes have typically also been accompanied with list decoding algorithms, results on larger classes of codes are typically combinatorial. Two classic results along these lines are (i) that random (linear) codes have optimal rate with high probability, and (ii) the fact, following from the Johnson bound, that any code with distance $1-1/q-\varepsilon^2$ can be list decoded from $1-1/q-\varepsilon$ fraction of errors.

Results of the second type are attractive since they guarantee list decodability for any code, deterministically, as long as the code has large enough distance. Unfortunately, it is known that the Johnson bound is tight for some codes [55], and so we cannot obtain a stronger form of (ii). However, one can hope for a result of the first type for list decodability, based on distance. More specifically, it is plausible that most *puncturings* of a code with good distance can beat the Johnson bound.

In Chapter 3, we obtained such a result for constant q. In particular, we showed that any code with distance $1-1/q-\varepsilon^2$ has many puncturings of rate $\Omega(\varepsilon^2/\log q)$ that are list decodable from a $1-1/q-\varepsilon$ fraction of errors. This rate is optimal up to constant factors when q is small, but is far from the optimal bound of $R^*(q,\varepsilon)$ for larger values of q, even when q depends only on ε and is otherwise constant. This leads to our second motivating question, the cliff-hanger at the end of Chapter 3:

Question 4.2. *Is it true that any code with distance* $1-1/q-\varepsilon^2$ *has many puncturings of rate* $\widetilde{\Omega}(R^*(q,\varepsilon))$ *that can list decode from* $1-1/q-\varepsilon$ *fraction of errors?*

4.1.1 Contributions of Chapter 4

We answer Questions 4.1 and 4.2 in the affirmative. Our main result addresses Question 4.2. We show that random puncturings of any code with distance $1-1/q-\varepsilon^2$ can list decode from $1-1/q-\varepsilon$ fraction of errors with rate

$$\frac{\min\{\varepsilon, q\varepsilon^2\}}{\log(q)\log^5(1/\varepsilon)}.$$

A corollary of this is that random linear codes are list decodable from $1-1/q-\varepsilon$ fraction of errors with the same rate. This improves upon our answers of Chapter 3 for $q \gtrsim \log^5(1/\varepsilon)$, and is optimal up to polylogarithmic factors.

Our main result also implies a positive answer to Question 4.1, and we show that there do exist RS codes that are list decodable beyond the Johnson bound. In fact, most sets of evaluation points will work: we show that if an appropriate number of evaluation points are chosen at random, then with constant probability the resulting RS code is list decodable from $1-\varepsilon$ fraction of errors with rate

$$\frac{\varepsilon}{\log(q)\log^5(1/\varepsilon)}.$$

This beats the Johnson bound for

$$\varepsilon \leq \widetilde{O}\left(\frac{1}{\log(q)}\right).$$

Finally, we prove some new average-distance, average-radius Johnson bounds, which we will need for our main results. We saw such a bound for $q=2$ in Theorem

2.8, and here we extend it to large alphabets. The proofs of these bounds are very similar to some of the proofs of the standard Johnson bound, and the fact that these proofs extend to the average case appears to be folklore. However, it's probably worth writing them down, so we'll do that in this chapter.

Relationship to impossibility results. Before we get into the details, we digress a bit to explain why our result on Reed-Solomon codes does not contradict the known impossibility results on this question. The lower bound of [50] works for list recovery but does not apply to our results about list decoding.[3] The lower bound of [12] does work for list decoding, but critically needs the set of evaluation points to be all of $\mathbb{F}_q$ (or more precisely the evaluation set should contain particularly structured subsets $\mathbb{F}_q$). Since we pick the evaluation points at random, this property is no longer satisfied. Finally, Cheng and Wan [19] showed that *efficiently* solving the list decoding problem for RS codes from $1 - \varepsilon$ fraction of errors with rate $\Omega(\varepsilon)$ would imply an efficient algorithm to solve the discrete log problem. However, this result does not rule out the list size being small (which is what our results imply), just that computing the list quickly is unlikely.

4.1.2 Chapter Organization

Our main technical result addresses Question 4.2 and states that a randomly punctured code[4] will retain the list decoding properties of the original code as long as the original code has good distance. Our results for RS codes (answering Question 4.1) and random linear codes follow by starting from the RS code evaluated on all of $\mathbb{F}_q$ and the q-ary Hadamard code, respectively.

[3]Our results can be extended to the list recovery setting, and the resulting parameters obey the lower bound of [50].

[4]Technically, our construction is slightly different than randomly punctured codes: see Remark 6.

We'll go over notation and review definitions in Section 4.2. In Section 4.3 we'll prove some average-radius, average-distance Johnson bounds which we will need. Preliminaries over with, we give a more detailed technical overview of our approach in Section 4.4. In Section 4.5 we state our main result, Theorem 4.6, about randomly punctured codes, and we apply it to Reed-Solomon codes and random linear codes. The remainder of the paper, Sections 4.6 and 4.7, are devoted to the proof of Theorem 4.6. Finally, we conclude with Section 4.8.

4.2 Yet more definitions

Motivated by Reed-Solomon codes, we consider random ensembles of linear codes $\mathcal{C} \subset \mathbb{F}_q^n$, where the field size q is large. We recall that a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ is *linear* if it forms a subspace of $\mathbb{F}_q^n$. Equivalently, $\mathcal{C} = \left\{ x^T G : x \in \mathbb{F}_q^k \right\}$ for a *generator matrix* $G \in \mathbb{F}_q^{k \times n}$. We will study the list decodability of these codes, up to "large" error rates $1 - 1/q - \varepsilon$, which is $1 - \Theta(\varepsilon)$ when $q \gtrsim 1/\varepsilon$. We recall Definition 2.3 and say that a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ is (ρ, L)-*list-decodable* if for all $z \in \mathbb{F}_q^n$, the number of codewords $c \in \mathcal{C}$ with $\delta(z, c) \leq \rho$ is at most L. As usual, $\delta(z, c)$ denotes the relative Hamming distance between z and c. We will actually study a slightly stronger notion of list decodability, which we waved our hands about in Chapters 2 and 3 and which was explicitly studied in [49]. We say that a code $\mathcal{C} \subset \mathbb{F}_q^n$ is (ρ, L)-*average-radius list-decodable* if for all $z \in \mathbb{F}_q^n$ and all sets Λ of $L+1$ codewords $c \in \mathcal{C}$, the average distance between elements of Λ and z is at least ρ. Notice that standard list decoding can be written in this language with the average replaced by a maximum.

As before, we are interested in the trade-off between ε, L, and the rate of the code $\mathcal{C}$. The *rate* of a linear code $\mathcal{C}$ is defined to be $\dim(\mathcal{C})/n$, where $\dim(\mathcal{C})$ refers to the dimension of $\mathcal{C}$ as a subspace of $\mathbb{F}_q^n$. As in the small-alphabet case in Chapter 3, We'll consider ensembles of linear codes where the generator vectors are independent; this includes random linear codes and Reed Solomon codes with random evaluation points. More precisely, a distribution on the matrices G induces a distribution on linear codes. We say that such a distribution on linear codes $\mathcal{C}$ has *independent symbols* if the columns of the generator matrix G are selected independently.

We will be especially interested in codes with randomly sampled symbols, where a new code (with a shorter block length) is created from an old code by including a few symbols of the codeword at random. We recall Definition 3.11 of a *randomly sampled* code: suppose that $\mathcal{C}_0 \subset \mathbb{F}_q^{n_0}$ is a q-ary code with block length n_0. Form a new code $\mathcal{C} \subset \mathbb{F}_q^n$ from $\mathcal{C}_0$ by choosing n symbols uniformly at random, with replacement from $[n_0]$. That is, choose a multiset $\{t_1, \ldots, t_n\} \subset [n_0]$ by choosing each $i_j \in [n_0]$ independently, uniformly. Then for each $x \in \mathbb{F}_q^k$, define $\mathcal{C}(x)$ by

$$\mathcal{C}(x) = (\mathcal{C}_0(x)_{t_1}, \mathcal{C}_0(x)_{t_2}, \ldots, \mathcal{C}_0(x)_{t_n}).$$

Notice that randomly sampled codes have independent symbols by definition.

Remark 6 (Sampling vs. Puncturing)**.** *We make the same remark here as we did in Chapter 3, Remark 5. That is, the operation of randomly* sampling *a code is very similar to that of randomly* puncturing *a code. The only difference is that we sample with replacement, while a randomly punctured code can be viewed as a code where the sampling is done without replacement. Our method of sampling is convenient for our analysis because of the independence. However, for the parameter regimes we will work in, collisions are overwhelmingly unlikely, and the distribution on randomly sampled codes is indeed very similar to that of randomly punctured codes.*

A few more bits of notation: as in Chapter 3, the size of $\mathcal{C}$ will be $|\mathcal{C}| = N$, and throughout we will consider linear codes $\mathcal{C} \subseteq \mathbb{F}_q^n$ of block length n and message length

k, with generator matrices $G \in \mathbb{F}_q^{k \times n}$. For a message $x \in \mathbb{F}_q^k$, we will write $c = \mathcal{C}(x)$ for the encoding $\mathcal{C}(x) = x^T G$. We will be interested in subsets $\Lambda \subseteq \mathbb{F}_q^k$ of size L (the *list size*), which we will identify, when convenient, with the corresponding subset of $\mathcal{C}$.

For $x, y \in \mathbb{F}_q^n$, let $\mathrm{agr}(x, y) = n(1 - \delta(x, y))$ be the number of symbols in which x and y agree. For a vector $v = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$ and a set $S \subseteq [n]$, we will use v_S to denote the restriction of v to the coordinates indexed by S. We use log to denote the logarithm base 2, and ln to denote the natural log.

4.3 Average-radius Johnson bounds

Now, we'll prove two average-radius, average-distance variants on the Johnson bound. These two statements are based on two proofs of of the (standard) Johnson bound, found in [59] and [84], respectively. It appears to be folklore that such statements are true (and follow from the proofs in the two works cited above), but we include them below for completeness.

Theorem 4.3. *Let $\mathcal{C} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ be any code. Then for all $\Lambda \subset \mathbb{F}_q^k$ of size L and for all $z \in \mathbb{F}_q^n$, and for all $\varepsilon \in (0, 1)$,*

$$\sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) \leq \frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1 + \varepsilon^2\right)\left(1 - \frac{1}{q}\right) - \frac{n}{2L\varepsilon} \sum_{x \neq y \in \Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y)).$$

Remark 7. *As with Theorem 2.8, a "normal" Johnson-bound (a la Theorem 2.7) follows by bounding $\delta(\mathcal{C}(x), \mathcal{C}(y)) \geq \delta(\mathcal{C})$ for all x, y, and by bounding $\sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) \geq L \min_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z)$.*

Proof. Fix a $z \in \mathbb{F}_q^n$. The crux of the proof is to map the relevant vectors over $\mathbb{F}_q^n$ to vectors in $\mathbb{R}^{nq}$ as follows. Given a vector $u \in \mathbb{F}_q^n$, let $u' \in \mathbb{R}^{nq}$ denote the concatenation

$$u' = (e_{u_1}, e_{u_2}, \ldots, e_{u_n}),$$

where $e_{u_i} \in \{0,1\}^q$ is the vector which is one in the u_i'th index and zero elsewhere. (Above, we fix an arbitrary mapping of $\mathbb{F}_q$ to $[q]$). In particular, for an $x \in \Lambda$, we will use $\mathcal{C}'(x)$ to denote the mapping of the codeword $\mathcal{C}(x)$. Finally let $v \in \mathbb{R}^{nq}$ be

$$v = \varepsilon \cdot z' + \left(\frac{1-\varepsilon}{q}\right) \cdot \mathbf{1},$$

where $\mathbf{1}$ denotes the all-ones vector.

Given the definitions above, it can be verified that the identities below hold for every $x \neq y \in \Lambda$:

$$\langle \mathcal{C}'(x), v \rangle = \varepsilon \cdot \mathrm{agr}(\mathcal{C}(x), z) + \frac{(1-\varepsilon)n}{q}, \tag{4.1}$$

$$\langle v, v \rangle = \frac{n}{q} + \varepsilon^2 \left(1 - \frac{1}{q}\right) n, \tag{4.2}$$

$$\langle \mathcal{C}'(x), \mathcal{C}'(y) \rangle = n(1 - \delta(\mathcal{C}(x), \mathcal{C}(y)), \tag{4.3}$$

and

$$\langle \mathcal{C}'(x), \mathcal{C}'(x) \rangle = n. \tag{4.4}$$

Now consider the following sequence of relations:

(4.5)
$$0 \le \left\langle \sum_{x\in\Lambda} (\mathcal{C}'(x) - v)\, , \sum_{x\in\Lambda} (\mathcal{C}'(x) - v) \right\rangle$$
$$= \sum_{x,y\in\Lambda} \langle \mathcal{C}'(x), \mathcal{C}'(y)\rangle - \sum_{x,y\in\Lambda} (\langle \mathcal{C}'(x), v\rangle + \langle \mathcal{C}'(y), v\rangle) + \sum_{x,y\in\Lambda} \langle v, v\rangle$$
$$= \sum_{x\in\Lambda} \langle \mathcal{C}'(x), \mathcal{C}'(x)\rangle + \sum_{x\neq y\in\Lambda} \langle \mathcal{C}'(x), \mathcal{C}'(y)\rangle - 2L\cdot \sum_{x\in\Lambda} \langle \mathcal{C}'(x), v\rangle + \sum_{x,y\in\Lambda} \langle v, v\rangle$$

(4.6)
$$= nL + n \sum_{x\neq y\in\Lambda} (1 - \delta(\mathcal{C}(x), \mathcal{C}(y)))$$
$$- 2L\cdot \sum_{x\in\Lambda} \left(\varepsilon \cdot \operatorname{agr}(\mathcal{C}(x), z) + \frac{(1-\varepsilon)n}{q} \right) + L^2 \cdot \left(\frac{n}{q} + \varepsilon^2 \left(1 - \frac{1}{q}\right) n \right)$$
$$= nL^2 \cdot \left(1 + \frac{1}{q} + \varepsilon^2 \left(1 - \frac{1}{q}\right) - \frac{2(1-\varepsilon)}{q} \right) - n \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y)) - 2L\varepsilon \cdot \sum_{x\in\Lambda} \operatorname{agr}(\mathcal{C}(x), z)$$

(4.7)
$$= nL^2 \cdot \left((1+\varepsilon^2)\left(1 - \frac{1}{q}\right) + \frac{2\varepsilon}{q} \right) - n \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y)) - 2L\varepsilon \cdot \sum_{x\in\Lambda} \operatorname{agr}(\mathcal{C}(x), z)$$

In the above, (4.5) follows from the fact that the norm of a vector is always positive and (4.6) follows from (4.1), (4.2), (4.3) and (4.4).

Equation (4.7) then implies that

$$2L\varepsilon \cdot \sum_{x\in\Lambda} \operatorname{agr}(\mathcal{C}(x), z) \le nL^2 \cdot \left((1+\varepsilon^2)\left(1 - \frac{1}{q}\right) + \frac{2\varepsilon}{q} \right) - n \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y)),$$

which implies the statement after rearranging terms. □

Next, we prove a second average-radius variant of the Johnson bound, which has been copied almost verbatim from [84].

Theorem 4.4. *Let $\mathcal{C} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ be any code. Then for all $\Lambda \subset \mathbb{F}_q^k$ of size L and for* *all* $z \in \mathbb{F}_q^n$,

$$\sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) \le \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x),\mathcal{C}(y))}\right).$$

Proof. For every $j \in [n]$, define

$$a_j = |\,\{x \in \Lambda | \mathcal{C}(x)_j = z_j\}\,|.$$

Note that

$$\sum_{j=1}^{n} a_j = \sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z), \tag{4.8}$$

and

$$\begin{aligned}
\sum_{j=1}^{n} \binom{a_j}{2} &= \frac{1}{2}\cdot \sum_{j=1}^{n} \sum_{x\neq y\in\Lambda} \mathbf{1}_{\mathcal{C}(x)_j = z_j} \mathbf{1}_{\mathcal{C}(y)_j = z_j} \\
&\le \sum_{j=1}^{n} \sum_{x\neq y\in\Lambda} \mathbf{1}_{\mathcal{C}(x)_j = \mathcal{C}(y)_j} \\
&= \frac{1}{2}\cdot \sum_{x\neq y\in\Lambda} \mathrm{agr}(\mathcal{C}(x),\mathcal{C}(y)) \\
&= \frac{L(L-1)n}{2} - \frac{n}{2}\sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x),\mathcal{C}(y)). \qquad (4.9)
\end{aligned}$$

Next, note that by the Cauchy-Schwartz inequality,

$$\sum_{j=1}^{n} \binom{a_i}{2} = \frac{1}{2}\left(\sum_{j=1}^{n} a_j^2 - \sum_{j=1}^{n} a_j\right) \ge \frac{1}{2n}\left(\sum_{j=1}^{n} a_j\right)^2 - \frac{1}{2}\sum_{j=1}^{n} a_j.$$

Combining the above with (4.8) and (4.9) implies that

$$\left(\sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z)\right)^2 - n\cdot \sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) - \left(n^2L(L-1) - n^2 \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x),\mathcal{C}(y))\right) \le 0,$$

which in turn implies (by the fact that the sum we care about lies in between the two roots of the quadratic equation) that

$$\sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) \le \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x\neq y\in\Lambda} \delta(\mathcal{C}(x),\mathcal{C}(y))}\right),$$

which completes the proof. □

4.4 Overview of approach

In this section, we give a technical overview of our argument, and point out where it differs from previous approaches, and in particular from the approach in Chapter 3. The main difficulty that arose in Chapter 3, and again arises here, is that the codewords are not independent. When the codewords *are* independent, as with a general random code, we saw in the proof of Theorem 2.4 that optimal list-decodability follows from a simple union bound: for a given set of messages Λ and a received word z, the probability that z lies close to the encodings of all messages in Λ is extremely small. However, without independence, this probability is not so small, and this approach fails.

In Proposition 2.6, we got around this by considering only the linearly independent messages in Λ, but at the cost of exponentially large list sizes. The exponential dependence on ε can actually be removed for a *constant* fraction of errors, by a careful analysis of the dependence between codewords corresponding to linearly dependent messages [44]. However, such techniques do not seem to work in the large-error regime that we consider. Instead, in Chapter 3, we avoided analyzing the dependence between the codewords by (impicitly) doing the union bound in a smarter way. By considering the geometry of these sets Λ, we used a mean-width argument to take advantage of the fact that the well-behaved-ness of the all of the Λ followed from the well-behaved-ness of a few extreme cases. We could indeed afford a union bound over these few cases.

However, the argument in Chapter 3 did not scale well with q; we pointed out in Remark 3 where we gave up on large alphabet sizes. Handling large alphabets is necessary for the application to Reed-Solomon codes. In this chapter, we'll follow the same basic idea of Chapter 3—avoiding the naive union bound using techniques from high-dimensional probability—but we will handle large alphabets. Instead of a simple mean-width argument, we'll have to get our hands a little dirtier and do a chaining argument, like we outlined in Chapter 2, Section 2.4. We outline the approach in slightly more detail below.

As before, our proof actually establishes *average-radius* list decodability, which has the advantage of linearizing the problem. However, instead of using the simplex embedding to formulate a sufficient condition, like we did in of Section 3.3 in the previous chapter, we will be a little more direct. After some rearranging (which is encapsulated in Proposition 4.5), it turns out that it's sufficient to control

$$\sum_{c\in\Lambda} \mathrm{agr}(z,c) = \sum_{c\in\Lambda}\sum_{j=1}^{n} \mathbf{1}_{c_j=z_j}$$

uniformly over all $\Lambda \subseteq \mathcal{C}$ and all $z \in \mathbb{F}_q^n$.

We will show that this is true in expectation; that is, we will bound

$$\mathbb{E}\max_{\Lambda,z}\sum_{c\in\Lambda}\sum_{j=1}^{n} \mathbf{1}_{c_j=z_j}. \tag{4.10}$$

The proof proceeds in two steps. The first (more straightforward) step is to argue that if the expectation and the maximum over Λ were reversed in (4.10), then we would have the control we need. To that end, we introduce a parameter

$$\mathcal{E} = \max_{|\Lambda|=L}\mathbb{E}\max_{z\in\mathbb{F}_q^n}\sum_{c\in\Lambda}\sum_{j=1}^{n} \mathbf{1}_{c_j=z_j}.$$

It is not hard to see that the received word z which maximizes the agreement is the one which, for each j, agrees with the plurality of the c_j for $c \in \Lambda$. That is,

$$\max_{z\in\mathbb{F}_q^n}\sum_{c\in\Lambda}\sum_{j=1}^{n} \mathbf{1}_{c_j=z_j} = \sum_{j=1}^{n}\max_{\alpha\in\mathbb{F}_q}|\{c\in\Lambda \,:\, c_j=\alpha\}| =: \sum_{j=1}^{n}\mathrm{plurality}_j\left(\Lambda\right).$$

Thus, to control $\mathcal{E}$, we must understand the expected pluralities. For our applications, this follows from standard Johnson-bound type arguments.

Of course, it is generally not okay to switch expectations and maxima; we must also argue that the quantity inside the maximum does not deviate too much from its mean in the worst case. This is the second and more complicated step of our argument. We must control the deviation

$$\sum_{j=1}^{n} \big(\mathrm{plurality}_j(\Lambda) - \mathbb{E}\mathrm{plurality}_j(\Lambda)\big) \tag{4.11}$$

uniformly over all Λ of size L. By the assumption of independent symbols (that is, independently chosen evaluation points for the Reed-Solomon code, or independent generator vectors for random linear codes), each summand in (4.11) is independent.

Sums of independent random variables tend to be reasonably concentrated, but, as pointed out above, because the codewords are not independent there is no reason that the pluralities themselves need to be particularly well-concentrated. Thus, we cannot handle a union bound over all $\Lambda \subseteq \mathcal{C}$ of size L. Instead, we use a chaining argument to deal with the union bound; the idea is that if the set Λ is close to the set Λ' (say they overlap significantly), then we should not have to union bound over both of them as though they were unrelated. Our main theorem, Theorem 4.6, bounds the deviation (4.11), and thus bounds (4.10) in terms of $\mathcal{E}$. We control $\mathcal{E}$ in the Corollaries 4.7 and 4.8, and then explain the consequences for Reed-Solomon codes and random linear codes in Sections 4.5.2 and 4.5.3.

We prove Theorem 4.6 in Section 4.6. To carry out the intuition above, we first pass to the language of Gaussian processes, as per Figure 2.12. Through some standard tricks from high dimensional probability (the symmetrization and comparison arguments that we saw in Section 2.4), it will suffice to instead bound the Gaussian process

$$X(\Lambda) = \sum_{j=1}^{n} g_j \mathrm{plurality}_j(\Lambda). \tag{4.12}$$

uniformly over all Λ of size L, where the g_j are independent standard normal random variables.

Now, we condition on $\mathcal{C}$, considering only the randomness over the Gaussians. We control this process in Theorem 4.9, the proof of which is contained in Section 4.7. The process (4.12) induces a metric on the space of sets Λ: Λ is close to Λ' if the vectors of their pluralities are close, in ℓ_2 distance. Indeed, if Λ is close to Λ' in this sense, then the corresponding increment $X(\Lambda) - X(\Lambda')$ is small with high probability. Now the situation is more in line with that discussed in Section 2.4 in Chapter 2, and our intuition about "wasting" the union bound on close-together Λ and Λ' can be made precise. In particular, Dudley's theorem [80, 104] bounds the supremum of the process in terms of the size of ε-nets with respect to this distance.

Thus, our proof of Theorem 4.9 boils down to constructing nets on the space of Λ's. In fact, our nets are quite simple—smaller nets consist of all of the sets of size $L/2^t$, for $t = 1, \ldots, \log(L)$. However, showing that the width of these nets is small is trickier. Our argument actually uses the structure of the chaining argument that is at the heart of the proof of Dudley's theorem: instead of arguing that the width of the net is small, we argue that each successive net cannot have points that are too far from the previous net, and thus build the "chain" step-by-step. With some work, one can abtract out a distance argument and apply Dudley's theorem as a black box. However, at the point that we are explicitly constructing the chains, it actually takes a bit longer to package things up for Dudley's theorem than to write out the chaining argument directly. To this end, (and to keep the dissertation self-contained), we unwrap Dudley's theorem in Section 4.7.2, as part of our proof.

We construct and control our nets in Lemma 4.10, which we prove in Section 4.7.3. Briefly, the idea is as follows. In order to show that a set Λ of size $L/2^t$ is "close" to some set Λ' of size $L/2^{t+1}$, we use the probabilistic method. We choose a set $\Lambda' \subseteq \Lambda$ at random, and argue that in expectation (after some appropriate normalization), the two are "close." Thus, the desired Λ' exists. However, the expected distance of Λ to Λ' in fact depends on the quantity

$$Q_t = \max_{|\Lambda| = L/2^t} \sum_{j=1}^{n} \mathrm{plurality}_j(\Lambda).$$

For $t = 0$, this is the quantity that we were trying to control in the first place in (4.10). Carrying this quantity through our argument, we are able to solve for it at the end and obtain our bound.

Controlling Q_t for $t > 0$ requires a bit of delicacy. In particular, as defined above $Q_{\log(L)}$ is deterministically equal to n, which it turns out is too large for our applications. To deal with this, we actually chain over not just the Λ, but also the set of the symbols $j \in [n]$ that we consider.[5] In fact, if we did not do this trick, we would recover (with some extra logarithmic factors) our results from Chapter 3.

Our argument has a similar flavor to some existing arguments in other domains, for example [92, 93], where a quantity analogous to Q_0 arises, and where analogous nets will work; indeed, those works are a major inspiration for our approach. There are a few main differences between that work and what we do here, although it is possible that one could re-frame our argument to mimic those. The first difference is that our proof of distance is structurally quite different; we actually prove distance step-by-step, by constructing the chains. The second difference is the trick described above, where we chain over the symbols $j \in [n]$ as well as the sets Λ. This is the part that makes the argument obnoxious to repackage for Dudley's theorem. One informal way to describe this trick is to say that we use qualitatively different chains for different sets Λ; how the set $I \subset [n]$ of evaluation points changes over the chain

[5]In particular we lied a little bit above, and (4.12) is not actually the Gaussian process we end up analyzing.

depends on the initial set Λ. In this sense, our argument smells a bit more like the "generic chaining" of [104].

4.5 Main theorem

In this section, we state our main technical result, Theorem 4.6. To begin, we first give a sufficient condition for list decodability, which is weaker than the sufficient condition we gave in Chapter 3. This sufficient condition is known as *average-radius* list-decodability. It's been implicitly studied for a long time (indeed, we used it implicitly in Chapter 3, and this is the condition that our average-radius Johnson bounds of Section 4.3 show), and it was first explicitly studied in [49]. All of our results in this chapter will actually show average-radius list decodability, and the following proposition shows that this will imply the standard notion of list decodability.

Proposition 4.5. *Suppose that*

$$\max_{z \in \mathbb{F}_q^n} \max_{\Lambda \subset \mathbb{F}_q^k, |\Lambda| = L} \sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) < nL\left(\varepsilon + \frac{1}{q}\right).$$

Then $\mathcal{C}$ is $(1 - 1/q - \varepsilon, L-1)$-list decodable.

Proof. By definition, $\mathcal{C}$ is $(1 - {}^1\!/_q - \varepsilon, L-1)$-list decodable if for any $z \in \mathbb{F}_q^n$ and any set $\Lambda \subset \mathbb{F}_q^k$ of size L, there is at least one message $x \in \Lambda$ so that $\mathrm{agr}(\mathcal{C}(x), z)$ is at most $n\left(\varepsilon + {}^1\!/_q\right)$, that is, if

$$\max_{z \in \mathbb{F}_q^n} \max_{|\Lambda| = L} \min_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) < n\left(\varepsilon + \frac{1}{q}\right).$$

Since the average is always larger than the minimum, it suffices for

$$\max_{z \in \mathbb{F}_q^n} \max_{|\Lambda| = L} \sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) < Ln\left(\varepsilon + \frac{1}{q}\right),$$

as claimed. □

Our main theorem gives conditions on ensembles of linear codes under which $\mathbb{E}\max_{z,\Lambda}\sum_{x\in\Lambda}\mathrm{agr}(\mathcal{C}(x),z)$ is bounded. Thus, it gives conditions under which Proposition 4.5 holds.

Theorem 4.6. *Fix $\varepsilon > 0$. Let $\mathcal{C}$ be a random linear code with independent symbols. Let*

$$\mathcal{E} = \max_{\Lambda\subset\mathbb{F}_q^k, |\Lambda|=L} \mathbb{E}_{\mathcal{C}} \max_{z\in\mathbb{F}_q^k}\left(\sum_{x\in\Lambda}\mathrm{agr}(\mathcal{C}(x),z)\right).$$

Then

$$\mathbb{E}_{\mathcal{C}}\max_{z\in\mathbb{F}_q^n}\max_{\Lambda\subset\mathbb{F}_q^k,|\Lambda|=L}\sum_{x\in\Lambda}\mathrm{agr}(\mathcal{C}(x),z) \le \mathcal{E} + Y + \sqrt{\mathcal{E}Y},$$

where

$$Y = C_0 L\log(N)\log^5(L)$$

for an absolute constant C_0.

Together with Proposition 4.5, Theorem 4.6 implies results about the list decodability of random codes with independent symbols, which we present next.

Remark 8. *We have chosen the statement of the theorem which gives the best bounds for Reed-Solomon codes, where $q \gg L$ is a reasonable parameter regime. An inspection of the proof shows that we may replace one $\log(L)$ factor with $\min\{\log(L),\log(q)\}$.*

Before we prove Theorem 4.6, we derive some consequences of it for randomly sampled codes, in terms of the distance of the original code. We work out two corollaries to this effect in Section 4.5.1 below. Our motivating examples are Reed-Solomon codes with random evaluation points, and random linear codes, which both fit within this framework. Indeed, Reed-Solomon codes with random evaluation points are obtained by sampling symbols from the Reed-Solomon code with block length $n = q$, and a random linear code is a randomly sampled Hadamard code.

We'll discuss the implications and optimality for the two motivating examples below in Sections 4.5.2 and 4.5.3 respectively.

4.5.1 Codes with good distance have abundant optimally-list-decodable puncturings

We'll prove two statements. The first holds for all q, but only yields the correct list size when q is small. The second holds for $q \gtrsim 1/\varepsilon^2$, and gives an improved list size in this regime. As discussed below in Section 4.5.3, our results are nearly optimal in both regimes. The proofs of both results follow from the average-radius Johnson bounds of Section 4.3; they amount to controlling the quantity $\mathcal{E}$. We state both results first, and then prove them.

The following result is intended for use with small q.

Corollary 4.7 (Small q)**.** *Let $\mathcal{C}_0$ be a linear code over $\mathbb{F}_q$ with distance $1 - \frac{1}{q} - \frac{\varepsilon^2}{2}$. Suppose that*

$$n \geq \frac{C_0 \log(N) \log^5(L)}{\min\{\varepsilon, q\varepsilon^2\}},$$

and choose $\mathcal{C}$ to be a randomly sampled version of $\mathcal{C}_0$, of block length n. Then, with constant probability over the choice of $\mathcal{C}$, the code $\mathcal{C}$ is $(1 - 1/q - \varepsilon', 2/\varepsilon^2)$-list decodable, where $\varepsilon' = \left(2 + \sqrt{2}\right)\varepsilon$.

Corollary 4.7 holds for all values of q, but the list size $L \gtrsim \varepsilon^{-2}$ is suboptimal when $q \gtrsim 1/\varepsilon$. To that end, we include the following corollary, which holds when $q \gtrsim 1/\varepsilon^2$ and attains the "correct" list size.[6]

Corollary 4.8 (Large q)**.** *Suppose that $q > 1/\varepsilon^2$, and that ε is sufficiently small. Let $\mathcal{C}_0$ be a linear code over $\mathbb{F}_q$ with distance $1 - \varepsilon^2$. Let*

$$n \geq \frac{2C_0 \log(N) \log^5(L)}{\varepsilon},$$

[6]As discussed below, we do not know good lower bounds on list sizes for large q; by "correct" we mean matching the performance of a general random code.

and choose $\mathcal{C}$ to be a randomly sampled version of $\mathcal{C}_0$, of block length n. Then, with constant probability over the choice of $\mathcal{C}$, the code $\mathcal{C}$ is $(1-\varepsilon', 1/\varepsilon)$-list decodable, where $\varepsilon' = 5\varepsilon$.

Remark 9 (Average-radius list decodability)**.** *We remark that the proofs of both Corollaries 4.7 and 4.8 go through Proposition 4.5, and thus actually show average-radius list decodability, not just list decodability. In particular, the applications to both Reed-Solomon codes and random linear codes hold under this stronger notion as well.*

We prove Corollaries 4.7 and 4.8 below.

Proof of Corollary 4.7. Suppose that $L \geq 2/\varepsilon^2$ and that the distance of $\mathcal{C}_0$ is at least $1 - 1/q - \varepsilon^2/2$. We need an average-radius version of the Johnson bound, which we provide in Theorem 4.3 in Appendix 4.3. By Theorem 4.3, for any $z \in \mathbb{F}_q^n$ and for all $\Lambda \subset \mathbb{F}_q^k$ of size L,

$$\sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) \leq \frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1+\varepsilon^2\right)\left(1-\frac{1}{q}\right) - \frac{n}{2L\varepsilon}\sum_{x\neq y\in\Lambda}\delta(\mathcal{C}(x),\mathcal{C}(y)). \tag{4.13}$$

By Theorem 4.6, it suffices to control $\mathcal{E}$. Since the right hand side above does not depend on z,

$$\begin{aligned}
\mathcal{E} &= \max_{|\Lambda|=L} \mathbb{E}_{\mathcal{C}} \max_{z\in\mathbb{F}_q^k} \sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) \\
&\leq \max_{|\Lambda|=L} \mathbb{E}_{\mathcal{C}} \max_{z\in\mathbb{F}_q^k} \left(\frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1+\varepsilon^2\right)\left(1-\frac{1}{q}\right) - \frac{n}{2L\varepsilon}\sum_{x\neq y\in\Lambda}\delta(\mathcal{C}(x),\mathcal{C}(y))\right) \\
&= \max_{|\Lambda|=L} \left(\frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1+\varepsilon^2\right)\left(1-\frac{1}{q}\right) - \frac{n}{2L\varepsilon}\sum_{x\neq y\in\Lambda}\mathbb{E}_{\mathcal{C}}\delta(\mathcal{C}(x),\mathcal{C}(y))\right) \\
&\leq \frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1+\varepsilon^2\right)\left(1-\frac{1}{q}\right) - \frac{n(L-1)\left(1-\frac{1}{q}-\frac{\varepsilon^2}{2}\right)}{2\varepsilon} \qquad (4.14)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{nL}{q} + \frac{nL\varepsilon}{2}\left(\frac{3}{2} - \frac{1}{q}\right) + \frac{n\left(1 - \frac{1}{q} - \frac{\varepsilon^2}{2}\right)}{2\varepsilon} \\
&\le \frac{nL}{q} + \frac{3nL\varepsilon}{4} + \frac{n}{2\varepsilon} \\
&\le nL\left(\frac{1}{q} + \varepsilon\right). \qquad (4.15)
\end{aligned}
$$

In the above, (4.14) follows from the fact that the original code had (relative) distance $1-1/q-\varepsilon^2/2$ and that in the construction of $\mathcal{C}$ from $\mathcal{C}_0$, pairwise Hamming distances are preserved in expectation. Finally, (4.15) follows from the assumption that $L \ge 2/\varepsilon^2$.

Recall from the statement of Theorem 4.6 that we have defined

$$Y = C_0 L \log(N) \log^5(L),$$

so the assumption on n implies that

$$Y \le nL \min\{\varepsilon, q\varepsilon^2\}.$$

Suppose that $q\varepsilon \le 1$, so that $Y \le nLq\varepsilon^2$. Plugging this along with (4.15) into Theorem 4.6, we obtain

$$
\begin{aligned}
\mathbb{E}_{\mathcal{C}} \max_{z\in\mathbb{F}_q^n} \max_{\Lambda\subset\mathbb{F}_q^k, |\Lambda|=L} \sum_{x\in\Lambda} \operatorname{agr}(\mathcal{C}(x), z) &\le \mathcal{E} + Y + \sqrt{\mathcal{E}Y} \\
&\le nL\left(\frac{1}{q}+\varepsilon\right) + nLq\varepsilon^2 + nL\sqrt{q\varepsilon^2\left(\frac{1}{q}+\varepsilon\right)} \\
&= nL\left(\frac{1}{q} + \varepsilon\left(1 + q\varepsilon + \sqrt{1+q\varepsilon}\right)\right) \\
&\le nL\left(\frac{1}{q} + \varepsilon\left(2+\sqrt{2}\right)\right),
\end{aligned}
$$

using the assumption that $q\varepsilon \le 1$ in the final line. Thus, Proposition 4.5 implies that $\mathcal{C}$ is $\left(1 - {}^1/_q - (2+\sqrt{2})\varepsilon, 2/\varepsilon^2\right)$-list-decodable.

On the other hand, suppose that $q\varepsilon \geq 1$, so that $Y \leq nL\varepsilon$. Then following the same outline, we have

$$\begin{aligned}
\mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^n} \max_{\Lambda \subset \mathbb{F}_q^k, |\Lambda| = L} \sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) &\leq \mathcal{E} + Y + \sqrt{\mathcal{E}Y} \\
&\leq nL\left(\frac{1}{q} + \varepsilon\right) + nL\varepsilon + nL\sqrt{\varepsilon\left(\frac{1}{q}+\varepsilon\right)} \\
&= nL\left(\frac{1}{q} + \varepsilon\left(2 + \sqrt{\frac{1}{q\varepsilon} + 1}\right)\right) \\
&\leq nL\left(\frac{1}{q} + \varepsilon\left(2 + \sqrt{2}\right)\right),
\end{aligned}$$

using the assumption that $q\varepsilon \geq 1$ in the final line. Thus, in this case as well, $\mathcal{C}$ is $\left(1 - {}^1\!/_q - (2+\sqrt{2})\varepsilon, 2/\varepsilon^2\right)$-list-decodable.

This completes the proof of Corollary 4.7. □

Proof of Corollary 4.8. As with Corollary 4.7, we need an average-radius version of the Johnson bound. In this case, we follow a proof of the Johnson bound from [84], which gives a better dependence on ε in the list size when q is large. For completeness, our average-radius version of the proof is given in Appendix 4.3, Theorem 4.4.

We proceed with the proof of Corollary 4.8. By Theorem 4.4, for any $z \in \mathbb{F}_q^n$ and for all $\Lambda \subset \mathbb{F}_q^k$ of size L,

$$\sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) \leq \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x \neq y \in \Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y))}\right). \tag{4.16}$$

By Theorem 4.6, it suffices to control $\mathcal{E}$. Since the right hand side above does not depend on z,

$$\begin{aligned}
\mathcal{E} &= \max_{|\Lambda|=L} \mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^k} \sum_{x \in \Lambda} \mathrm{agr}(\mathcal{C}(x), z) \\
&\leq \max_{|\Lambda|=L} \mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^k} \left(\frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x \neq y \in \Lambda} \delta(\mathcal{C}(x), \mathcal{C}(y))}\right)\right) \qquad (4.17)
\end{aligned}$$

$$\leq \max_{|\Lambda|=L} \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x\neq y\in\Lambda} \mathbb{E}_{\mathcal{C}}\delta(\mathcal{C}(x),\mathcal{C}(y))}\right) \tag{4.18}$$

$$\leq \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1) - 4n^2 \sum_{x\neq y\in\Lambda} (1-\varepsilon^2)}\right) \tag{4.19}$$

$$\leq \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L(L-1)\varepsilon^2}\right)$$

$$< \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2L^2\varepsilon^2}\right)$$

$$\leq 2nL\varepsilon. \tag{4.20}$$

In the above, (4.17) follows from (4.16). (4.18) follows from Jensen's inequality. (4.19) follows from the fact that the original code had (relative) distance $1-\varepsilon^2$ and that in the construction of $\mathcal{C}$ from $\mathcal{C}_0$, pairwise Hamming distances are preserved in expectation. Finally, (4.20) follows from the assumption that $L \geq 1/\varepsilon$.

Now, Theorem 4.6 implies that

$$\begin{aligned}\mathbb{E}_{\mathcal{C}} \max_{z\in\mathbb{F}_q^n} \max_{\Lambda\subset\mathbb{F}_q^k, |\Lambda|=L} \sum_{x\in\Lambda} \mathrm{agr}(\mathcal{C}(x), z) &\leq \mathcal{E} + Y + \sqrt{\mathcal{E}Y} \\ &\leq 2\left(\mathcal{E}+Y\right) \\ &\leq 2\left(2nL\varepsilon + Y\right) \\ &\leq 5nL\varepsilon\end{aligned}$$

where as before

$$Y = C_0 L \log(N) \log^5(L)$$

and where we used the choice of n in the final line. Choose $\varepsilon' = 5\varepsilon$, so that whenever $5\varepsilon > 1/q$, Proposition 4.5 applies and completes the proof. Because we have chosen $\varepsilon > 1/\sqrt{q}$ (which is necessary in order for $\mathcal{C}_0$ to have distance $1-\varepsilon^2$), the condition that $5\varepsilon > 1/q$ holds for sufficiently small ε. $\square$

Next, we'll show how to apply Corollaries 4.7 and 4.8 to our headline results, about Reed-Solomon codes and random linear codes.

4.5.2 Most Reed-Solomon codes are list-decodable beyond the Johnson bound

Our results imply that a Reed-Solomon code with random evaluation points is, with high probability, list decodable beyond the Johnson bound. Recall Definition 2.1 of Reed-Solomon codes: For $q \geq n$, and an integer k, and let $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}_q$ be a list of "evaluation points." The corresponding Reed-Solomon code $\mathcal{C} \subset \mathbb{F}_q^n$ encodes a polynomial (message) $f \in \mathbb{F}_q[x]$ of degree at most $k-1$ as

$$\mathcal{C}(f) = (f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_n)) \in \mathbb{F}_q^n.$$

Note that there are q^k polynomials of degree at most $k-1$, and thus $|\mathcal{C}| = q^k$.

For Reed-Solomon codes, we are often interested in the parameter regime when $q \geq n$ is quite large. In particular, below we will be especially interested in the regime when $q \gg 1/\varepsilon^2$, and so we will use Corollary 4.8 for this application. To apply Corollary 4.8, let $\mathcal{C}_0$ be the Reed-Solomon code of block length q (that is, every point in $\mathbb{F}_q$ is evaluated), and choose the n evaluation points $(\alpha_i)_{i=1}^n$ for $\mathcal{C}$ independently from $\mathbb{F}_q$. We will choose the block length n so that

$$n \lesssim \frac{\log(N)\log^5(1/\varepsilon)}{\varepsilon}.$$

As we discussed in Chapter 2, the generator matrix for $\mathcal{C}$ will have full rank, and so the rate of $\mathcal{C}$ is at least

$$R \gtrsim \frac{\varepsilon}{\log(q)\log^5(1/\varepsilon)}. \tag{4.21}$$

Before we investigate the result of Corollary 4.8, let us pause to observe what the Johnson bound predicts for $\mathcal{C}$. The distance of $\mathcal{C}$ is exactly $1-(k-1)/n$. Indeed, any two polynomials of degree $k-1$ agree on at most $k-1$ points, and this is attained by, say, the zero polynomial and any polynomial with k distinct roots in $\{\alpha_1, \ldots, \alpha_n\}$. Thus, letting $\varepsilon = (k-1)/n$, the Johnson bound predicts that $\mathcal{C}$ has rate ε, distance $1-\varepsilon$, and is list decodable up to $1 - O(\sqrt{\varepsilon})$, with polynomial list sizes.

Now, we compare this to the result of Corollary 4.8. The distance of $\mathcal{C}_0$ is $1 - (k-1)/q$, so as long as $q \gtrsim k/\varepsilon^2$, we may apply Corollary 4.8. Then, Corollary 4.8 implies that the resulting Reed-Solomon code $\mathcal{C}$ has rate

$$\Omega\left(\frac{\varepsilon}{\log(q)\log^5(1/\varepsilon)}\right),$$

distance $1-\varepsilon$, and is list decodable up to radius $1-5\varepsilon$, with list sizes at most $1/\varepsilon$.

In particular, the tolerable error rate may be as large as $1 - O(\varepsilon)$, rather than $1 - O(\sqrt{\varepsilon})$, and the rate suffers only by logarithmic factors.

4.5.3 Near-optimal bounds for random linear codes over large alphabets

In addition to implying that most Reed-Solomon codes are list decodable beyond the Johnson bound, Corollaries 4.7 and 4.8 provide the best known bounds on random linear codes over large fields; this improves on the results of Chapter 3 for large q. Our new results are tight up to constant factors.

Suppose that $\mathcal{C}_0$ is the Hadamard code over $\mathbb{F}_q$ of dimension k; that is, the generator matrix of $\mathcal{C}_0 \in \mathbb{F}_q^{k \times q^k}$ has all the elements of $\mathbb{F}_q^k$ as its columns. The relative distance of $\mathcal{C}_0$ is $1 - {}^1\!/_q$, and so we may apply the corollaries with any $\varepsilon > 0$ that we choose.

To this end, fix $\varepsilon > 0$, and let $\mathcal{C}$ be a randomly sampled version of $\mathcal{C}_0$, of block length

$$n = \frac{2C_0 \log(q^k)\log^5(1/\varepsilon)}{\varepsilon}.$$

Regime	Best known rate for random linear codes	Upper bound on rate	Best known list size for random linear codes	Lower bound on list size
	$\frac{\varepsilon^2}{\log(q)}$, Chapter 3	$\frac{q\varepsilon^2}{\log(q)}$	$\frac{1}{\varepsilon^2}$ [20], Chapter 3, and Cor. 4.7	$\frac{1}{q^5\varepsilon^2}$ [61]
$q = \log^5(1/\varepsilon)$	$\frac{q\varepsilon^2}{\log(q)\log^5(1/\varepsilon)}$ Cor. 4.7			
$q = 1/\varepsilon$	$\frac{\varepsilon}{\log(q)\log^5(1/\varepsilon)}$ Cors. 4.7, 4.8	$1 - H_q\left(1 - \frac{1}{q} - \varepsilon\right)$		
$q = 1/\varepsilon^2$		ε	$\frac{1}{\varepsilon}$ Cor. 4.8	

Figure 4.1: The state of affairs for q-ary random linear codes. Above, the list decoding radius is $1 - 1/q - \varepsilon$, and we have suppressed constant factors.

It is not hard to see that the generator matrix of $\mathcal{C}$ will have full rank with high probability, and so the rate of $\mathcal{C}$ will be at least

$$R = k/n = \frac{\min\{\varepsilon, q\varepsilon^2\}}{2C_0 \log(q) \log^5(1/\varepsilon)}. \tag{4.22}$$

By Corollary 4.7, $\mathcal{C}$ is list decodable up to error radius $1 - {}^1/_q - O(\varepsilon)$, with list sizes at most $2/\varepsilon^2$. When $q \gtrsim 1/\varepsilon^2$, Corollary 4.8 applies, and we get the same result with an improved list size of $1/\varepsilon$.

We compare these results to known results on random linear codes in Figure 4.1. The best known results on the list decodability of random linear codes, from [111], state that a random linear code of rate on the order of $\varepsilon^2/\log(q)$ is $(1 - 1/q - \varepsilon, O(1/\varepsilon^2))$-list decodable. This is optimal (up to constant factors) for constant q, but it is suboptimal for large q. In particular, the bound on the rate is surpassed by our bound (4.22) when $q \gtrsim \log^5(1/\varepsilon)$.

When the error rate is $1 - 1/q - \varepsilon$, the optimal information rate for list decodable codes is given by the list decoding capacity theorem, which implies that we must have $R \leq 1 - H_q(1 - 1/q - \varepsilon)$. This expression behaves differently for different parameter regimes; in particular, when $q \leq 1/\varepsilon$ and ε is sufficiently small, we have

$$1 - H_q(1 - 1/q - \varepsilon) = \frac{q\varepsilon^2}{2\log(q)(1 - 1/q)} + O(\varepsilon^3),$$

while when $q \geq 2^{\Omega(1/\varepsilon)}$, the optimal rate is linear in ε. For the first of these two regimes—and indeed whenever $q \leq 1/\mathrm{poly}(\varepsilon)$—our bound (4.22) is optimal up to polylogarithmic factors in $1/\varepsilon$. In the second regime, when q is exponentially large, our bound slips by an additional factor of $\log(q)$.

For the $q \leq 1/\varepsilon^2$ regime, our list size of $1/\varepsilon^2$ matches existing results, and when q is constant it matches the lower bounds of [61]. For $q \geq 1/\varepsilon^2$, our list size of $1/\varepsilon$ is the best known. There is a large gap between the lower bound of [61] and our upper bounds for large q. However, there is evidence that the most of discrepancy is due to the difficulty of obtaining lower bounds on list sizes. Indeed, a (general) random code of rate $1 - H_q(1 - 1/q - \varepsilon) - 1/L$ is list-decodable with list size L, implying that $L = O(1/\varepsilon)$ is the correct answer for $q \gtrsim 1/\varepsilon$. Thus, while our bound seems like it is probably weak for q super-constant but smaller than $1/\varepsilon^2$, it seems correct for $q \gtrsim 1/\varepsilon^2$.

4.6 Proof of Theorem 4.6: reduction to Gaussian processes

In this section, we prove Theorem 4.6. For the reader's convenience, we restate the theorem here.

Theorem (Theorem 4.6, restated)**.** *Fix $\varepsilon > 0$. Let $\mathcal{C}$ be a random linear code with* *independent symbols. Let*

$$\mathcal{E} = \max_{\Lambda \subset \mathbb{F}_q^k, |\Lambda| = L} \mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^k} \left(\sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) \right).$$

Then

$$\mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^n} \max_{\Lambda \subset \mathbb{F}_q^k, |\Lambda| = L} \sum_{x \in \Lambda} \operatorname{agr}(c(x), z) \leq \mathcal{E} + Y + \sqrt{\mathcal{E} Y},$$

where

$$Y = C_0 L \log(N) \log^5(L)$$

for an absolute constant C_0.

To begin, we introduce some notation.

Notation 4.1. *For a set $\Lambda \subseteq \mathbb{F}_q^k$, let $\mathbf{pl}_j^{\mathcal{C}}$ denote the (fractional)* plurality *of index $j \in [n]$:*

$$\mathbf{pl}_j^{\mathcal{C}}(\Lambda) = \frac{1}{|\Lambda|} \max_{\alpha \in \mathbb{F}_q} \left| \{ x \in \Lambda \,:\, \mathcal{C}(x)_j = \alpha \} \right|.$$

For a set $I \subseteq [n]$, let

$$\mathbf{pl}_I^{\mathcal{C}}(\Lambda) \in [0,1]^n$$

be the the vector $(\mathbf{pl}_j^{\mathcal{C}}(\Lambda))_{j=1}^n$ restricted to the coordinates in I, with the remaining coordinates set to zero. When $\mathcal{C}$ is fixed, we will drop the superscript for notational clarity.

Rephrasing the goal in terms of our new notation, the quantity we wish to bound is

$$\mathbb{E}_{\mathcal{C}} \max_{z \in \mathbb{F}_q^n} \max_{|\Lambda| = L} \sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) = L \cdot \mathbb{E}_{\mathcal{C}} \max_{|\Lambda| = L} \sum_{j \in [n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda). \tag{4.23}$$

Moving the expectation inside the maximum recovers the quantity

$$\mathcal{E} = L \cdot \max_{|\Lambda| = L} \mathbb{E}_{\mathcal{C}} \sum_{j \in [n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda),$$

which appears in the statement of Theorem 4.6. Since Theorem 4.6 outsources a bound on $\mathcal{E}$ to the user (in our case, Corollaries 4.7 and 4.8), we seek to control the worst deviation

$$\begin{aligned}\mathcal{F} &:= L \cdot \mathbb{E}_{\mathcal{C}} \max_{|\Lambda|=L} \left| \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) - \mathbb{E}_{\mathcal{C}} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right| \\ &= L \cdot \mathbb{E}_{\mathcal{C}} \max_{|\Lambda|=L} \left| \sum_{j\in[n]} \left(\mathbf{pl}_j^{\mathcal{C}}(\Lambda) - \mathbb{E}_{\mathcal{C}}\, \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right) \right| . \qquad (4.24)\end{aligned}$$

Indeed, let

$$Q = Q(\mathcal{C}) = \max_{|\Lambda|=L} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda),$$

so that $L \cdot \mathbb{E}_{\mathcal{C}} Q$ is the quantity in (4.23). Then,

$$\begin{aligned}\mathbb{E}_{\mathcal{C}} Q &= \mathbb{E}_{\mathcal{C}} \max_{|\Lambda|=L} \left(\sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) - \mathbb{E}_{\mathcal{C}} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) + \mathbb{E}_{\mathcal{C}} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right) \\ &\le \mathbb{E}_{\mathcal{C}} \max_{|\Lambda|=L} \left| \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) - \mathbb{E}_{\mathcal{C}} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right| + \max_{|\Lambda|=L} \mathbb{E}_{\mathcal{C}} \sum_{j\in[n]} \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \\ &= \frac{1}{L} \left(\mathcal{F} + \mathcal{E} \right), \qquad (4.25)\end{aligned}$$

so getting a handle on $\mathcal{F}$ would be enough. With that in mind, we return our attention to (4.24). By the assumption of independent symbols, the summands in (4.24) are independent. By a standard symmetrization argument followed by a comparison argument (Lemmas 2.16 and 2.17, respectively), we may bound

$$\frac{1}{L}\mathcal{F} = \mathbb{E}_{\mathcal{C}} \max_{|\Lambda|=L} \left| \sum_{j\in[n]} \left(\mathbf{pl}_j^{\mathcal{C}}(\Lambda) - \mathbb{E}_{\mathcal{C}}\, \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right) \right| \qquad (4.26)$$

$$\le \sqrt{2\pi}\, \mathbb{E}_{\mathcal{C}} \mathbb{E}_g \max_{|\Lambda|=L} \left| \sum_{j\in[n]} g_j\, \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right| \qquad (4.27)$$

Above, g_j are independent standard normal random variables.

Let

$$\mathcal{S}_0 = \{[n]\} \times \left\{ \Lambda \subset \mathbb{F}_q^k \,:\, |\Lambda| = L \right\}, \qquad (4.28)$$

so that we wish to control

$$\mathbb{E}_{\mathcal{C}}\mathbb{E}_g \max_{(I,\Lambda)\in\mathcal{S}_0} \left| \sum_{j\in I} g_j \, \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \right| .$$

At this stage, maximimizing I over the one-element collection $\{[n]\}$ may seem like a silly use of notation, but we will use the flexibility as the argument progresses.

Condition on the choice of $\mathcal{C}$ until further notice, and consider only the randomness over the Gaussian random vector $g = (g_1, \ldots, g_n)$. In particular, this fixes $Q = Q(\mathcal{C})$, and also fixes the function $\mathbf{pl}^{\mathcal{C}}$. In order to take advantage of (4.26), we will study the Gaussian process

$$X(I,\Lambda) = \sum_{j\in I} g_j \, \mathbf{pl}_j^{\mathcal{C}}(\Lambda) \tag{4.29}$$

indexed by $(I,\Lambda) \in \mathcal{S}_0$. The bulk of the proof of Theorem 4.6 is the following theorem, which controls the expected supremum of $X(I,\Lambda)$, in terms of Q.

Theorem 4.9. *Condition on the choice of $\mathcal{C}$. Then*

$$\mathbb{E}_g \max_{(I,\Lambda)\in\mathcal{S}_0} |X(I,\Lambda)| \le C_3 \sqrt{Q \log(N) \log^5(L)}$$

for some constant C_3.

We will prove Theorem 4.9 in Section 4.7. First, let us show how it implies Theorem 4.6. By (4.26), and applying Theorem 4.9, we have

$$\begin{aligned}
\mathcal{F} &\le \sqrt{2\pi}\, L\, \mathbb{E}_{\mathcal{C}}\mathbb{E}_g \max_{(I,\Lambda)\in\mathcal{S}_0} \left| \sum_{j\in I} g_j \, \mathbf{pl}_j^{\mathcal{C}}(z,\Lambda) \right| \\
&\le C_3\sqrt{2\pi}\, L\, \mathbb{E}_{\mathcal{C}} \left[\sqrt{Q \log(N) \log^5(L)} \right] \\
&\le C_3\sqrt{2\pi}\, L \sqrt{\mathbb{E}_{\mathcal{C}} Q \, \log(N) \log^5(L)}
\end{aligned}$$

Using the fact (4.25) that $\mathbb{E}_{\mathcal{C}} Q \le \frac{1}{L}(\mathcal{E}+\mathcal{F})$,

$$\begin{aligned}
\mathcal{F} &\le C_3\sqrt{2\pi} \sqrt{L(\mathcal{E}+\mathcal{F}) \log(N) \log^5(L)} \\
&=: \sqrt{Y(\mathcal{E}+\mathcal{F})},
\end{aligned}$$

where

$$Y := C_3^2 2\pi L \log(N) \log^5(L).$$

Solving for $\mathcal{F}$, this implies that

$$\mathcal{F} \leq \frac{Y + \sqrt{Y^2 + 4Y\mathcal{E}}}{2} \leq Y + \sqrt{Y\mathcal{E}}.$$

Then, from (4.25) and the definition of Q (recall that $L \cdot \mathbb{E}_{\mathcal{C}} Q$ is the quantity in (4.23)),

$$\begin{aligned} \mathbb{E}_{\mathcal{C}} \max_{I,\Lambda} \sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) &= L\mathbb{E}_{\mathcal{C}} Q \\ &\leq \mathcal{E} + \mathcal{F} \\ &\leq \mathcal{E} + Y + \sqrt{Y\mathcal{E}}, \end{aligned}$$

as claimed. This proves Theorem 4.6.

4.7 Proof of Theorem 4.9: controlling a Gaussian process

In this section, we prove Theorem 4.9. Recall that the goal was to control the Gaussian process (4.29) given by

$$X(I, \Lambda) = \sum_{j \in I} g_j \, \mathbf{pl}_j^{\mathcal{C}}(\Lambda).$$

Recall also that we are conditioning on the choice of $\mathcal{C}$. Because of this, for notational convenience, we will drop the superscript on $\mathbf{pl}^{\mathcal{C}}$, and additionally identify $\Lambda \subset \mathbb{F}_q^k$ with the corresponding set of codewords $\{\mathcal{C}(x) : x \in \Lambda\} \subset \mathcal{C}$. That is, for this section, we will imagine that $\Lambda \subset \mathcal{C}$ is a set of codewords.

Notation 4.2. *When the code $\mathcal{C}$ is fixed (in particular, for the entirety of Section 4.7), we will identify $\Lambda \subset \mathbb{F}_q^k$ with $\Lambda \subset \mathcal{C}$, given by*

$$\Lambda \leftarrow \{\mathcal{C}(x) : x \in \Lambda\}.$$

To control the Gaussian process (4.29), we will use a chaining argument. We outlined the basic intuition of such an argument in Section 2.4. More precisely, we will define a series of nets, $\mathcal{S}_t \subset 2^{[n]} \times 2^{\mathcal{C}}$ and write, for any $(I_0, \Lambda_0) \in \mathcal{S}_0$,

$$|X(I_0, \Lambda_0)| \leq \left(\sum_{t=0}^{t_{\max}-1} |X(\pi_t(I_0, \Lambda_0)) - X(\pi_{t+1}(I_0, \Lambda_0))| \right) + |X(\pi_{t_{\max}}(I_0, \Lambda_0))| \,,$$

where $\pi_t(I_0, \Lambda_0) \in \mathcal{S}_t$ and $t_{\max} \in \mathbb{Z}$ will shortly be determined, $\pi_0(I_0, \Lambda_0) = (I_0, \Lambda_0)$. Then we will argue that each step in this "chain" (that is, each summand in the first term) is small with high probability, and union bound over all possible chains.

For Gaussian processes, such chaining arguments come in standard packages, for example Dudley's integral inequality [80], or Talagrand's generic chaining inequality [104]. We choose to unpack the argument for two reasons. The first and main reason is that our choice of nets is informed by the structure of the chaining argument. Thus, it is clearer to define the nets in the context of the complete argument. The second reason is to make the exposition self-contained.

We remark that, due to the nature of our argument, it is convenient for us to start with the large nets indexed by small t, and the small nets indexed by large t; this is in contrast with convention.

4.7.1 Defining the nets

We will define nets $\mathcal{S}_t$, for each t recursively. Begin by defining $\mathcal{S}_0$ as in (4.28), and let $\pi_0 : \mathcal{S}_0 \to \mathcal{S}_0$ be the identity map. Given $\mathcal{S}_t$, we will define $\mathcal{S}_{t+1}$, as well as the maps $\pi_{t+1} : \mathcal{S}_0 \to \mathcal{S}_{t+1}$. Our maps π_t will satisfy the guarantees of the following lemma.

Lemma 4.10. *Fix a parameter $\eta = 1/\log(L)$, and suppose $c_0 < L < N/2$ is sufficiently large, for some constant c_0. Let*

$$t_{\max} = \frac{\log(L) - 2\log(1/\eta) - 2}{\log(2/(1-\eta))}. \tag{4.30}$$

Then there is a sequence of maps

$$\pi_t : \mathcal{S}_0 \to 2^{[n]} \times 2^{\mathcal{C}}$$

for $t = 0, \ldots, t_{\max}$ *so that* π_0 *is the identity map and so that the following hold.*

1. *For all* $(I_0, \Lambda_0) \in \mathcal{S}_0$*, and for all* $t = 0, \ldots, t_{\max}$*, the pair* $(I_t, \Lambda_t) = \pi_t(I_0, \Lambda_0)$ *obeys*

   $$\sum_{j \in I_t} \mathbf{pl}_j(\Lambda_t) \leq Q_t := (1+\eta)^t \, Q. \tag{4.31}$$

   and

   $$\left(\frac{1-\eta}{2}\right)^t L \leq |\Lambda_t| \leq \left(\frac{1+\eta}{2}\right)^t L. \tag{4.32}$$

2. *For all* $(I_0, \Lambda_0) \in \mathcal{S}_0$*, and for all* $t = 0, \ldots, t_{\max} - 1$*, the pair* $(I_{t+1}, \Lambda_{t+1}) = \pi_{t+1}(I_0, \Lambda_0)$ *obeys*

   $$\left\| \mathbf{pl}_{I_t}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1}) \right\|_2 \leq \frac{C_4 \sqrt{Q_t \log(L)}}{\eta \sqrt{|\Lambda_t|}} \tag{4.33}$$

   for some constant C_4.

3. *For all* $t = 0, \ldots, t_{\max}$*, define*

   $$\mathcal{S}_t := \left\{ \pi_t(I_0, \Lambda_0) \, : \, (I_0, \Lambda_0) \in \mathcal{S}_0 \right\}.$$

   Then, for $t \geq 1$*, the size of the net* $\mathcal{S}_t$ *satisfies*

   $$|\mathcal{S}_t| \leq C_6 \binom{N}{eL/2^t} \binom{N}{eL/2^{t-1}}, \tag{4.34}$$

   for some constant C_6*, while* $|\mathcal{S}_0| = \binom{N}{L}$.

4.7.2 Proof of Theorem 4.9 from Lemma 4.10: a chaining argument

Before we prove Lemma 4.10, we will show how to use it to prove Theorem 4.9. This part of the proof follows the standard proof of Dudley's theorem [80], and can be skipped by the reader already familiar with it.[7] As outlined above, we will use a chaining argument to control the Gaussian process in Theorem 4.9. We wish to control

$$\mathbb{E} \max_{(I,\Lambda)\in\mathcal{S}_0} |X(I,\Lambda)| .$$

For any $(I_0, \Lambda_0) \in \mathcal{S}_0$, write

$$\begin{aligned} |X(I_0,\Lambda_0)| &\le \left(\sum_{t=0}^{t_{\max}-1} |X(\pi_t(I_0,\Lambda_0)) - X(\pi_{t+1}(I_0,\Lambda_0))| \right) + |X(\pi_{t_{\max}}(I_0,\Lambda_0))| \\ &=: S(I_0,\Lambda_0) + |X(\pi_{t_{\max}}(I_0,\Lambda_0))| , \end{aligned} \tag{4.35}$$

where Lemma 4.10 tells us how to pick $(I_t, \Lambda_t) := \pi_t(I_0, \Lambda_0)$, and where we have used the fact that $\pi_0(I_0, \Lambda_0) = (I_0, \Lambda_0)$.

Each increment

$$X(\pi_t(I_0,\Lambda_0)) - X(\pi_{t+1}(I_0,\Lambda_0)) = \sum_{j=1}^{n} g_j \left[\mathbf{1}_{j\in I_t}\, \mathbf{pl}_j(\Lambda_t) - \mathbf{1}_{j\in I_{t+1}}\, \mathbf{pl}_j(\Lambda_{t+1}) \right]$$

[7] Assuming that the reader is willing to take our word on the calculations.

is a Gaussian random variable (see Fact 2.13) with variance

$$\begin{aligned}
\sum_{j=1}^{n} &\left(\mathbf{1}_{j\in I_t}\, \mathbf{pl}_j(\Lambda_t) - \mathbf{1}_{j\in I_{t+1}}\, \mathbf{pl}_j(\Lambda_{t+1}) \right)^2 \\
&= \left\| \, \mathbf{pl}_{I_t}(\Lambda_t) - \, \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1}) \right\|_2^2 \\
&\le \frac{C_4^2 Q_t \log(L)}{\eta^2 |\Lambda_t|} \qquad \text{by (4.33)} \\
&\le \frac{C_4^2 Q_t \log(L)}{\eta^2 \left(\frac{1-\eta}{2}\right)^t L} \qquad \text{by (4.32)} \\
&\le \frac{C_4^2 (1+\eta)^t Q \log(L)}{\eta^2 \left(\frac{1-\eta}{2}\right)^t L} \qquad \text{by (4.31)} \\
&\le \left(\frac{C_4}{\eta} \right)^2 \left(\frac{Q \log(L) (2(1+2\eta))^t}{L} \right) \qquad \text{using } \eta \le 1/2. \\
&\le \left(\frac{eC_4}{\eta} \right)^2 \left(\frac{Q \log(L) 2^t}{L} \right) \qquad \text{using } \eta = 1/\log(L) \text{ and } t_{\max} \le \log(L).
\end{aligned}$$

Thus, for each $0 \le t < t_{\max}$, and for any $u, a_t \ge 0$,

$$\begin{aligned}
\mathbb{P}\left\{ |X(\pi_t(z,\Lambda)) - X(\pi_{t+1}(z,\Lambda))| > u \cdot a_t \right\} &\le \exp\left(\frac{-u^2 \cdot a_t^2}{2 \sum_{j=1}^n \left(\mathbf{1}_{j\in I_t}\, \mathbf{pl}_j(\Lambda_t) - \mathbf{1}_{j\in I_{t+1}}\, \mathbf{pl}_j(\Lambda_{t+1}) \right)^2} \right) \\
&\le \exp\left(\frac{-u^2 \cdot a_t^2}{2 \left(\frac{eC_4}{\eta} \right)^2 \left(\frac{Q \log(L) 2^t}{L} \right)} \right) \\
&=: \exp\left(\frac{-u^2 \cdot a_t^2}{\delta_t^2} \right). \qquad (4.36)
\end{aligned}$$

In the above, we used the fact that for a Gaussian variable g with variance σ, $\mathbb{P}\{|g| > u\} \le \exp(-u^2/(2\sigma^2))$. Now we union bound over all possible "chains" (that is, sequences $\{\pi_t(I_0,\Lambda_0)\}_t$) to bound the probability that there exists a $(I_0,\Lambda_0) \in \mathcal{S}_0$ so that the first term $S(I_0,\Lambda_0)$ in (4.35) is large. Consider the event that for all $(I_0,\Lambda_0) \in \mathcal{S}_0$,

$$|X(\pi_t(I_0,\Lambda_0)) - X(\pi_{t+1}(I_0,\Lambda_0))| \le u \cdot a_t,$$

for a_t to be determined shortly. In the favorable case that this event occurs, the first term in (4.35) is bounded by

$$S(I_0, \Lambda_0) = \sum_{t=0}^{t_{\max}-1} |X(\pi_t(I_0, \Lambda_0)) - X(\pi_{t+1}(I_0, \Lambda_0))| \le u \cdot \sum_{t=0}^{t_{\max}-1} a_t,$$

for all (I_0, Λ_0). Let

$$N_t = \begin{cases} C_6 \binom{N}{eL/2^t} \binom{N}{eL/2^{t-1}} & t \ge 1 \\ \binom{N}{L} & t = 0 \end{cases} \tag{4.37}$$

be our bound on $|\mathcal{S}_t|$, given by (4.34) in Lemma 4.10. Then probability that the above good event fails to occur is at most, by the union bound,

$$\mathbb{P}\left\{ \max_{(I_0,\Lambda_0)\in\mathcal{S}_0} S(I_0, \Lambda_0) > u \cdot \sum_{t=0}^{t_{\max}-1} a_t \right\} \le \sum_{t=0}^{t_{\max}-1} N_t N_{t+1} \exp\left(\frac{-u^2 \cdot a_t^2}{\delta_t^2} \right).$$

Indeed, there are at most $N_t N_{t+1}$ possible "steps" between $\pi_t(I_0, \Lambda_0)$ and $\pi_{t+1}(I_0, \Lambda_0)$, and the probability that any step at level t fails is given by (4.36).

Choose

$$a_t = \sqrt{2 \ln (N_t N_{t+1})}\, \delta_t. \tag{4.38}$$

This choice will imply that

$$\mathbb{E} \max_{(I_0,\Lambda_0)\in\mathcal{S}_0} S(I_0, \Lambda_0) \le 2 \sum_{t=1}^{t_{\max}-1} a_t. \tag{4.39}$$

Indeed, to establish (4.39), we may follow a (standard) computation similar to that of Proposition 2.14 that we saw in Chapter 2. Let $A = \sum_{t=1}^{t_{\max}-1}$. Then

$$\begin{aligned} \mathbb{E} \max_{(I,\Lambda)\in\mathcal{S}_0} S(I, \Lambda) &= \int_{u=0}^{\infty} \mathbb{P}\left\{ \max_{(I,\Lambda)} S(I, \Lambda) > u \right\} du \\ &\le A + \int_{u=A}^{\infty} \sum_{t=0}^{t_{\max}-1} N_t N_{t+1} \exp\left(\frac{-u^2 \cdot a_t^2}{\delta_t^2 A^2} \right) du \\ &= A + \int_{u=A}^{\infty} \sum_{t=0}^{t_{\max}-1} N_t N_{t+1} \exp\left(\frac{-2u^2 \ln (N_t N_{t+1})}{A^2} \right) du \\ &\le A + \sum_{t=0}^{t_{\max}-1} N_t N_{t+1} \int_{u=A}^{\infty} \exp\left(\frac{-2u^2 \ln (N_t N_{t+1})}{A^2} \right) du. \end{aligned}$$

Repeating the trick (2.10), we estimate

$$\int_{u=A}^{\infty} \exp\left(\frac{-2u^2\ln\left(N_tN_{t+1}\right)}{A^2}\right) \le \frac{A}{4\ln\left(N_tN_{t+1}\right)}\exp\left(-2\ln\left(N_tN_{t+1}\right)\right) \le \frac{A}{4N_t^2N_{t+1}^2}.$$

Plugging this in, we get

$$\mathbb{E}\max_{(I,\Lambda)\in\mathcal{S}_0} S(I,\Lambda) \le A\left(1+\frac{1}{4}\sum_{t=0}^{t_{\max}-1}\frac{1}{N_tN_{t+1}}\right) \le 2A.$$

In the last inequality, we used the definition of $N_t = C_6\binom{N}{eL/2^t}\binom{N}{eL/2^{t+1}}$ if $t \ge 1$ and $N_0 = \binom{N}{L}$. In particular, we have used the fact that $N_t \ge 2$ for our setting of parameters. This establishes (4.39).

Now, plugging in our definition (4.38) of a_t and then of δ_t and N_t (Equations (4.36) and (4.37), respectively),

$$\begin{aligned}
\mathbb{E}\max_{(z,\Lambda)\in\mathcal{S}_0} S(I_0,\Lambda_0) &\le 2\sum_{t=0}^{t_{\max}-1}\sqrt{2\ln\left(N_tN_{t+1}\right)}\,\delta_t \\
&\lesssim \sum_{t=0}^{t_{\max}-1}\sqrt{\frac{L}{2^t}\log(N)}\left(\frac{1}{\eta}\sqrt{\frac{Q\log(L)2^t}{L}}\right) \\
&= t_{\max}\left(\frac{\sqrt{Q\log(N)\log(L)}}{\eta}\right) \\
&\le \log^2(L)\sqrt{Q\log(N)\log(L)}, \qquad (4.40)
\end{aligned}$$

after using the choice of $\eta = 1/\log(L)$ and $t_{\max} \le \log(L)$ in the final line.

With the first term $S(I_0,\Lambda_0)$ of (4.35) under control by (4.40), we turn to the second term, and we now bound the probability that the final term $X(\pi_{t_{\max}}(z,\Lambda))$ is large. Let $(I_{\max},\Lambda_{\max}) = \pi_{t_{\max}}(I_0,\Lambda_0)$, so we wish to bound the Gaussian random variable

$$X(\pi_{t_{\max}}(I_0,\Lambda_0)) = \sum_{j\in I_{\max}} g_j\,\mathbf{pl}_j(\Lambda_{\max}).$$

As with the increments in $S(I_0,\Lambda_0)$, we will first bound the variance of $X(\pi_{t_{\max}}(I_0,\Lambda_0))$. By (4.31), we know that

$$\sum_{j\in I_{\max}} \mathbf{pl}_j(\Lambda_{\max}) \le Q_{t_{\max}} \le eQ.$$

Further, since $\mathbf{pl}_j(\Lambda_{\max})$ is a fraction, we always have

$$\mathbf{pl}_j(\Lambda_{\max}) \leq 1.$$

By Hölder's inequality,

$$\sum_{j \in I_{\max}} \mathbf{pl}_j(\Lambda_{\max})^2 \leq \left(\sum_{j \in I_{\max}} \mathbf{pl}_j(\Lambda_{\max}) \right) \left(\max_{j \in I_{\max}} \mathbf{pl}_j(\Lambda_{\max}) \right) \leq eQ.$$

Thus, for each $(I_0, \Lambda_0) \in \mathcal{S}_0$, $X(\pi_{t_{\max}}(I_0, \Lambda_0))$ is a Gaussian random variable with variance at most eQ (using Fact 2.13). We recall the choice from (4.30) of

$$t_{\max} = \frac{\log(L) - 2\log(1/\eta) - 2}{1 + \log(1/(1-\eta))} \geq \log(L) - 2\log\log(L) - C_7, \tag{4.41}$$

for some constant C_7, for sufficiently large L. Because there are $|\mathcal{S}_{t_{\max}}| \leq \binom{N}{eL/2^{t_{\max}}}$ of these, Proposition 2.14 says that

$$\begin{aligned} \mathbb{E} \max_{(I_0,\Lambda_0) \in \mathcal{S}_0} |X(\pi_{t_{\max}}(I_0, \Lambda_0))| &\lesssim \sqrt{\ln |\mathcal{S}_{t_{\max}}|} \cdot \sqrt{Q} \\ &\lesssim \sqrt{\frac{LQ \log(N)}{2^{t_{\max}}}} \\ &\lesssim \log(L) \sqrt{Q \log(N)}, \end{aligned}$$

using the choice of $t_{\max}$ (and the bound on it in (4.41)) in the final line. Finally, putting together the two parts of (4.35), we have

$$\begin{aligned} \mathbb{E} \max_{(I_0,\Lambda_0) \in \mathcal{S}_0} X(I_0, \Lambda_0) &\lesssim \log^2(L) \sqrt{Q \log(N) \log(L)} + \log(L) \sqrt{Q \log(N)} \\ &\lesssim \log^2(L) \sqrt{Q \log(N) \log(L)}. \end{aligned} \tag{4.42}$$

This completes the proof of Theorem 4.9 (assuming Lemma 4.10).

4.7.3 Proof of Lemma 4.10: the desired nets exist

Finally, we prove Lemma 4.10. We proceed inductively. In addition to the conclusions of the lemma, we will maintain the inductive hypotheses

$$I_{t+1} \subseteq I_t \qquad \text{and} \qquad \Lambda_{t+1} \subseteq \Lambda_t \tag{4.43}$$

for all t.

For the base case, $t = 0$, we set $\pi_0(I_0, \Lambda_0) = (I_0, \Lambda_0)$. The definition of Q guarantees (4.31), and the definition of $\mathcal{S}_0$ guarantees (4.32). By definition $|\mathcal{S}_0| \leq \binom{N}{L}$. Further, since by definition $I_0 = [n]$, the first part of (4.43) is automatically satisfied. (We are not yet in a position to verify the base case for the second part of (4.43), having not yet defined Λ_1, but we will do so shortly).

We will need to keep track of how the pluralities $\mathbf{pl}_j(\Lambda_t)$ change, and for this we need the following notation.

Notation 4.3. *For $\alpha \in \mathbb{F}_q$ and $\Lambda \subset \mathcal{C}$, let*

$$v_j(\alpha, \Lambda) = \frac{|\{c \in \Lambda : c_j = \alpha\}|}{|\Lambda|}$$

be the fraction of times the symbol α appears in the j'th symbol in Λ.

Now we define $\mathcal{S}_t$ for $t \geq 1$. Suppose we are given $(I_t, \Lambda_t) = \pi_t(I_0, \Lambda_0) \in \mathcal{S}_t$ satisfying the hypotheses of the lemma. We need to produce $(I_{t+1}, \Lambda_{t+1}) \in \mathcal{S}_{t+1}$, and we will use the probabilistic method. We will choose I_{t+1} deterministically based on Λ_t. Then we will choose Λ_{t+1} randomly, based on Λ_t, and show that with positive probability, (I_{t+1}, Λ_{t+1}) obey the desired conclusions. Then we will fix a favorable draw of (I_{t+1}, Λ_{t+1}) and call it $\pi_{t+1}(I_0, \Lambda_0)$.

We choose I_{t+1} to be the "heavy" coordinates,

$$I_{t+1} := \left\{ j : |\Lambda_t| \, \mathbf{pl}_j(\Lambda_t) \geq \gamma \right\},$$

for

$$\gamma := \frac{4c_1 \log(L)}{(1-\eta)^2 \eta^2}, \tag{4.44}$$

where c_1 is a suitably large constant to be fixed later. Notice that I_{t+1} depends only on Λ_t (and on $\mathcal{C}$, which for the moment is fixed).

Now consider drawing $\Lambda_{t+1} \subset \Lambda_t$ at random by including each element of Λ_t in Λ_{t+1} independently with probability $1/2$. We will choose some Λ_{t+1} from the support of this distribution.

Before we fix Λ_{t+1}, observe that we are already in a position to establish (4.43). Indeed, the second part of (4.43) holds for all t, because $\Lambda_{t+1} \subseteq \Lambda_t$ by construction. To establish the first part of (4.43) for $t, t+1$, we use that $\Lambda_t \subseteq \Lambda_{t-1}$ (by induction, using (4.43) for $t-1, t$), and this implies that for all $j \in I_{t+1}$,

$$\begin{aligned}
\gamma &\leq |\Lambda_t| \, \mathbf{pl}_j(\Lambda_t) \\
&= \max_\alpha |\{c \in \Lambda_t \, : \, c_j = \alpha\}| \\
&\leq \max_\alpha |\{c \in \Lambda_{t-1} \, : \, c_j = \alpha\}| \\
&= |\Lambda_{t-1}| \, \mathbf{pl}_j(\Lambda_{t-1}),
\end{aligned}$$

and hence $j \in I_t$. Thus,

$$I_{t+1} \subseteq I_t. \tag{4.45}$$

Before we move on to the other inductive hypotheses, stated in Lemma 4.10, we must fix a "favorable" draw of Λ_{t+1}. In expectation, Λ_{t+1} behaves like Λ_t, and so the hope is that the "step"

$$\mathbf{pl}_{I_t}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})$$

is small. We quantify this in the following lemma.

Lemma 4.11. *For all j,*

$$\mathbb{E}\left[|\Lambda_{t+1}| |\, \mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1})| \right] \leq \sqrt{C_5 |\Lambda_t| \log(L) \, \mathbf{pl}_j(\Lambda_t)}$$

and

$$\mathbb{E}\left[|\Lambda_{t+1}|^2 (\, \mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1}))^2 \right] \leq C_5 |\Lambda_t| \log(L) \, \mathbf{pl}_j(\Lambda_t)$$

for some constant C_5.

Proof. The second statement implies the first, by Jensen's inequality, so we prove only the second statement. For each $\alpha \in \mathbb{F}_q$, and each $j \in [n]$, consider the random variable

$$\begin{aligned} Y_j(\alpha) &:= |\Lambda_{t+1}| \left(v_j(\alpha, \Lambda_{t+1}) - v_j(\alpha, \Lambda_t) \right) \\ &= \sum_{c \in \Lambda_t : c_j = \alpha} \left(\xi_c - \frac{|\Lambda_{t+1}|}{|\Lambda_t|} \right) \\ &= \sum_{c \in \Lambda_t : c_j = \alpha} \left(\xi_c - \frac{1}{2} \right) + \sum_{c \in \Lambda_t : c_j = \alpha} \left(\frac{1}{2} - \frac{|\Lambda_{t+1}|}{|\Lambda_t|} \right) \\ &= \sum_{c \in \Lambda_t : c_j = \alpha} \left(\xi_c - \frac{1}{2} \right) + v_j(\alpha, \Lambda_t) \sum_{c \in \Lambda_t} \left(\frac{1}{2} - \xi_c \right) \\ &=: Z_j(\alpha) + W_j(\alpha), \end{aligned}$$

where above ξ_c is 1 if $c \in \Lambda_{t+1}$ and 0 otherwise. Both $Z_j(\alpha)$ and $W_j(\alpha)$ are sums of independent mean-zero random variables, and we use Chernoff bounds to control them. First, $Z_j(\alpha)$ is a sum of $|\Lambda_t| v_j(\alpha, \Lambda_t)$ independent mean-zero random variables, and a Chernoff bound (Theorem 2.15) yields

$$\mathbb{P}\left\{ |Z_j(\alpha)| > u \right\} \le 2 \exp\left(\frac{-2u^2}{|\Lambda_t| v_j(\alpha, \Lambda_t)} \right) \le 2 \exp\left(\frac{-2u^2}{|\Lambda_t|\, \mathbf{pl}_j(\Lambda_t)} \right).$$

Similarly, $W_j(\alpha)$ is a sum of $|\Lambda_t|$ independent mean-zero random variables, each contained in

$$\left[-\frac{v_j(\alpha, \Lambda_t)}{2}, \frac{v_j(\alpha, \Lambda_t)}{2} \right] \subseteq \left[-\frac{\mathbf{pl}_j(\Lambda_t)}{2}, \frac{\mathbf{pl}_j(\Lambda_t)}{2} \right],$$

and we have

$$\mathbb{P}\left\{ |W_j(\alpha)| > u \right\} \le 2 \exp\left(\frac{-2u^2}{|\Lambda_t|\, \mathbf{pl}_j(\Lambda_t)^2} \right) \le 2 \exp\left(\frac{-2u^2}{|\Lambda_t|\, \mathbf{pl}_j(\Lambda_t)} \right),$$

using the fact that $\mathbf{pl}_j(\Lambda_t) \le 1$. Together,

$$\mathbb{P}\left\{ |Y_j(\alpha)| > u \right\} \le \mathbb{P}\left\{ |W_j(\alpha)| > u/2 \right\} + \mathbb{P}\left\{ |Z_j(\alpha)| > u/2 \right\} \le 4 \exp\left(\frac{-u^2}{2\, \mathbf{pl}_j(\Lambda_t) |\Lambda_t|} \right),$$

Let

$$T_j = \{\alpha \in \mathbb{F}_q \,:\, \exists c \in \Lambda_t, c_j = \alpha\}$$

be the set of symbols that show up in the j'th coordinates of Λ_t. Then

$$|T_j| \leq \min\{q, |\Lambda_t|\} \leq L.$$

By the union bound, and letting $v = u^2$,

$$\mathbb{P}\left\{\max_{\alpha\in\mathbb{F}_q} Y_j(\alpha)^2 > v\right\} = \mathbb{P}\left\{\max_{\alpha\in T_j} Y_j(\alpha)^2 > v\right\} \leq 4L\exp\left(\frac{-v}{2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|}\right). \tag{4.46}$$

Next, we show that if all of the $Y_j(\alpha)$ are under control, then so are the pluralities $\mathbf{pl}_j(\Lambda_t)$. For any four numbers A, B, C, D with $A \leq B$ and $C \leq D$, we have

$$|B - D| \leq \max\{|B - C|, |D - A|\}. \tag{4.47}$$

Indeed, we have

$$B-D \leq (B-D)+(D-C) = B-C \qquad \text{and} \qquad D-B \leq (D-B)+(B-A) = D-A.$$

The claim (4.47) follows. Now, for fixed j, let

$$\alpha = \mathrm{argmax}_{\sigma\in T_j} v_j(\sigma, \Lambda_t) \qquad \text{and} \qquad \beta = \mathrm{argmax}_{\sigma\in T_j} v_j(\sigma, \Lambda_{t+1}),$$

so that

$$|\Lambda_{t+1}|v_j(\alpha, \Lambda_{t+1}) \leq |\Lambda_{t+1}|v_j(\beta, \Lambda_{t+1}) \qquad \text{and} \qquad |\Lambda_{t+1}|v_j(\beta, \Lambda_t) \leq |\Lambda_{t+1}|v_j(\alpha, \Lambda_t).$$

By (4.47), we have

$$\begin{aligned}|\Lambda_{t+1}||\,\mathbf{pl}_j(\Lambda_{t+1}) - \mathbf{pl}_j(\Lambda_t)| &= |\Lambda_{t+1}||v_j(\beta, \Lambda_{t+1}) - v_j(\alpha, \Lambda_t)| \\ &\leq |\Lambda_{t+1}|\max\{|v_j(\alpha, \Lambda_t) - v_j(\alpha, \Lambda_{t+1})|, |v_j(\beta, \Lambda_t) - v_j(\beta, \Lambda_{t+1})|\} \\ &\leq \max_{\alpha\in T_j} |Y_j(\alpha)|.\end{aligned}$$

Thus, the probability that $|\,\mathbf{pl}_j(\Lambda_{t+1}) - \mathbf{pl}_j(\Lambda_t)|$ is large is no more than the probability that $\max_{\alpha\in T_j}|Y_j(\alpha)|$ is large, and we conclude from (4.46) that

$$\mathbb{P}\left\{|\Lambda_{t+1}|^2(\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1}))^2 > v\right\} \le 4L\exp\left(\frac{-v}{2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|}\right).$$

Integrating, we bound the expectation by

$$\begin{aligned}\mathbb{E}|\Lambda_{t+1}|^2(\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1}))^2 &= \int_0^\infty \mathbb{P}\left\{\max_{\alpha\in T_j} Y_j(\alpha)^2 > v\right\}\,dv\\ &\le A + 4L\int_A^\infty \exp\left(\frac{-v}{2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|}\right)\,dv\\ &= A + 4L\cdot 2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|\cdot\exp\left(\frac{-A}{2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|}\right)\end{aligned}$$

for any $A \ge 0$. Choosing $A = 2\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|\ln(4L)$ gives

$$\mathbb{E}|\Lambda_{t+1}|^2(\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1}))^2 \le 2|\Lambda_t|\,\mathbf{pl}_j(\Lambda_t)\,(\ln(4L)+1)\,.$$

Setting C_5 correctly proves the second item in Lemma 4.11, and the first follows from Jensen's inequality. □

The next lemma uses Lemma 4.11 to argue that a number of good things happen all at once.

Lemma 4.12. *There is some $\Lambda_{t+1} \subseteq \Lambda_t$ so that:*

1.

$$\left(\frac{1-\eta}{2}\right)^{t+1} L \le \left(\frac{1-\eta}{2}\right)|\Lambda_t| \le |\Lambda_{t+1}| \le \left(\frac{1+\eta}{2}\right)|\Lambda_t| \le \left(\frac{1+\eta}{2}\right)^{t+1} L.$$

2.

$$\sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_{t+1}) \le \sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_t) + \sum_{j\in I_{t+1}} \sqrt{\frac{c_1|\Lambda_t|\log(L)\,\mathbf{pl}_j(\Lambda_t)}{|\Lambda_{t+1}|^2}}$$

3.

$$\left(\sum_{j\in I_{t+1}} (\,\mathbf{pl}_j(\Lambda_{t+1}) - \mathbf{pl}_j(\Lambda_t))^2\right)^{1/2} \le \frac{\sqrt{c_1|\Lambda_t|\log(L)Q_t}}{|\Lambda_{t+1}|}$$

for some constant c_1*.*

Proof. We show that (for an appropriate choice of c_1), each of these items occurs with probability at least 2/3, 3/4, and 3/4, respectively. Thus, all three occur with probability at least 1/6, and in particular there is a set Λ_{t+1} which satisfies all three.

First, we address Item 1. By a Chernoff bound (Theorem 2.15),

$$\mathbb{P}\left\{\left||\Lambda_{t+1}| - \frac{1}{2}|\Lambda_t|\right| > u\right\} \leq 2\exp\left(-2u^2/|\Lambda_t|\right),$$

By the inductive hypothesis (4.32),

$$|\Lambda_t| \geq \left(\frac{1-\eta}{2}\right)^t L,$$

and so by our choice of $t_{\max}$ and the fact that $t \leq t_{\max}$, we have

$$|\Lambda_t| \geq 4/\eta^2. \tag{4.48}$$

Thus,

$$\mathbb{P}\left\{\left||\Lambda_{t+1}| - \frac{|\Lambda_t|}{2}\right| \geq \frac{\eta|\Lambda_t|}{2}\right\} \leq 2e^{-2} < 1/3.$$

Again by the inductive hypothesis (4.32) applied to $|\Lambda_t|$, we conclude that

$$\left(\frac{1-\eta}{2}\right)^{t+1} L \leq \left(\frac{1-\eta}{2}\right)|\Lambda_t| \leq |\Lambda_{t+1}| \leq \left(\frac{1+\eta}{2}\right)|\Lambda_t| \leq \left(\frac{1+\eta}{2}\right)^{t+1} L.$$

For Item 2, we invoke Lemma 4.11 and linearity of expectation to obtain

$$\mathbb{E}\sum_{j\in I_{t+1}} |\Lambda_{t+1}||\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1})| \leq \sum_{j\in I_{t+1}} \sqrt{C_5\log(L)\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|}.$$

By Markov's inequality, as long as $c_1 \geq 16C_5$, with probability at least 3/4,

$$\sum_{j\in I_{t+1}} |\Lambda_{t+1}||\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1})| \leq \sum_{j\in I_{t+1}} \sqrt{c_1\log(L)\,\mathbf{pl}_j(\Lambda_t)|\Lambda_t|},$$

and in the favorable case the triangle inequality implies

$$\begin{aligned}\sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_{t+1}) &\le \sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_t) + \sum_{j\in I_{t+1}} |\,\mathbf{pl}_j(\Lambda_t) - \mathbf{pl}_j(\Lambda_{t+1})| \\ &\le \sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_t) + \sum_{j\in I_{t+1}} \frac{\sqrt{c_1 \log(L)\, \mathbf{pl}_j(\Lambda_t)|\Lambda_t|}}{|\Lambda_{t+1}|}.\end{aligned}$$

Thus, Item 2 holds with probability at least 3/4.

Similarly, for Item 3, Lemma 4.11 and linearity of expectation (as well as Jensen's inequality) implies that

$$\begin{aligned}&\mathbb{E}\left(\sum_{j\in I_{t+1}} |\Lambda_{t+1}|^2(\,\mathbf{pl}_j(\Lambda_{t+1}) - \mathbf{pl}_j(\Lambda_t))^2\right)^{1/2} \\ &\qquad\le \left(\sum_{j\in I_{t+1}} C_5|\Lambda_t|\log(L)\,\mathbf{pl}_j(\Lambda_t)\right)^{1/2} \\ &\qquad\le \left(\sum_{j\in I_t} C_5|\Lambda_t|\log(L)\,\mathbf{pl}_j(\Lambda_t)\right)^{1/2} \qquad \text{since } I_{t+1}\subseteq I_t \\ &\qquad\le \sqrt{C_5|\Lambda_t|\log(L)Q_t} \qquad \text{by the inductive hypothesis (4.31)}\ .\end{aligned}$$

Again, Markov's inequality and an appropriate restriction on c_1 implies that Item 3 occurs with probability strictly more than 3/4.

This concludes the proof of Lemma 4.12. □

Finally, we show how Lemma 4.12 implies the conclusions of Lemma 4.10 for $t+1$, notably (4.31), (4.32) and (4.33). First, we observe that (4.32) follows immediately from Lemma 4.12, Item 1. Next we consider (4.31). The definition of I_{t+1} and the choice of γ, along with the fact from Lemma 4.12, Item 1 that $|\Lambda_{t+1}| \ge \left(\frac{1-\eta}{2}\right)|\Lambda_t|$, imply that for $j \in I_{t+1}$,

$$|\Lambda_t|\,\mathbf{pl}_j(\Lambda_t) \ge \gamma \ge \left(\frac{|\Lambda_t|}{|\Lambda_{t+1}|}\right)^2 \frac{c_1\log(L)}{\eta^2},$$

and so

$$\frac{\sqrt{c_1|\Lambda_t|\log(L)\,\mathbf{pl}_j(\Lambda_t)}}{|\Lambda_{t+1}|} \le \eta\,\mathbf{pl}_j(\Lambda_t). \tag{4.49}$$

Thus,

$$\begin{aligned}
\sum_{j\in I_{t+1}} \mathbf{pl}_j(\Lambda_{t+1}) &\le \sum_{j\in I_{t+1}} (1+\eta)\,\mathbf{pl}_j(\Lambda_t) && \text{by Lemma 4.12, Item 2 and from (4.49)}\\
&\le (1+\eta)\sum_{j\in I_t}\mathbf{pl}_j(\Lambda_t) && \text{since } I_{t+1}\subseteq I_t, \text{ by (4.45)}\\
&\le (1+\eta)\,Q_t && \text{by the inductive hypothesis (4.31) for } t\\
&= (1+\eta)^{t+1}\,Q && \text{by the definition of } Q_t\\
&= Q_{t+1}.
\end{aligned}$$

This establishes (4.31).

To establish the distance criterion (4.33), we use the triangle inequality to write

$$\|\,\mathbf{pl}_{I_t}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 = \|\,\mathbf{pl}_{I_{t+1}}(\Lambda_t) + \mathbf{pl}_{I_t\setminus I_{t+1}}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 \tag{4.50}$$

$$\le \|\,\mathbf{pl}_{I_{t+1}}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 \tag{4.51}$$

$$+ \|\,\mathbf{pl}_{I_t\setminus I_{t+1}}(\Lambda_t)\|_2 \tag{4.52}$$

The first term (4.51) is bounded by Lemma 4.12, Item 3, by

$$\|\,\mathbf{pl}_{I_{t+1}}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 \le \frac{\sqrt{c_1|\Lambda_t|\log(L)Q_t}}{|\Lambda_{t+1}|}.$$

To bound (4.52), we will bound both the ℓ_∞ and ℓ_1 norms of $\mathbf{pl}_{I_t\setminus I_{t+1}}(\Lambda_t)$ and use Hölder's inequality to control the ℓ_2 norm. By the inductive hypothesis (4.31) and the fact (4.45) that $I_{t+1}\subseteq I_t$,

$$\|\,\mathbf{pl}_{I_t\setminus I_{t+1}}(\Lambda_t)\|_1 \le \|\,\mathbf{pl}_{I_t}(\Lambda_t)\|_1 \le Q_t.$$

Also, by the definition of I_{t+1},

$$\| \mathbf{pl}_{I_t \setminus I_{t+1}}(\Lambda_t)\|_\infty \le \frac{\gamma}{|\Lambda_t|}.$$

Together, Hölder's inequality implies that

$$\| \mathbf{pl}_{I_t \setminus I_{t+1}}(\Lambda_t)\|_2 \le \sqrt{\| \mathbf{pl}_{I_t \setminus I_{t+1}}(\Lambda_t)\|_1 \| \mathbf{pl}_{I_t \setminus I_{t+1}}(\Lambda_t)\|_\infty} \le \sqrt{\frac{\gamma Q_t}{|\Lambda_t|}}.$$

This bounds the second term (4.52) of (4.50), and putting it all together we have

$$\| \mathbf{pl}_{I_t}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 \le \frac{\sqrt{c_1 |\Lambda_t| \log(L) Q_t}}{|\Lambda_{t+1}|} + \sqrt{\frac{\gamma Q_t}{|\Lambda_t|}}.$$

Using the fact from Lemma 4.12, Item 1 that $|\Lambda_t|/|\Lambda_{t+1}| \le 2/(1-\eta)$, as well as the definition of γ in (4.44), we may bound the above expression by

$$\| \mathbf{pl}_{I_t}(\Lambda_t) - \mathbf{pl}_{I_{t+1}}(\Lambda_{t+1})\|_2 \le \left(1 + \frac{1}{\eta}\right)\left(\frac{2}{1-\eta}\right)\sqrt{\frac{c_1 \log(L) Q_t}{|\Lambda_t|}}.$$

This establishes (4.33), for an appropriate choice of C_4, and for sufficiently large L (and hence sufficiently small η).

Finally, we verify the condition (4.34) on the size $|\mathcal{S}_{t+1}|$. By (4.32), and the fact that our choices of η and $t_{\max}$ imply that $(1+\eta)^t \le e$, $|\Lambda_t| \le eL/2^t$. We saw earlier that I_{t+1} depends only on Λ_t, so (using the fact that $L \le N/2$), there are at most

$$\sum_{r=1}^{eL/2^t} \binom{N}{r} \lesssim \binom{N}{eL/2^t}$$

choices for I_{t+1}. Similarly, we just chose Λ_{t+1} so that $|\Lambda_{t+1}| \le eL/2^{t+1}$, so there are at most $\sum_{r=1}^{eL/2^t} \binom{N}{r} \lesssim \binom{N}{eL/2^{t+1}}$ choices for Λ_{t+1}. Altogether, there are at most

$$C_6 \binom{N}{eL/2^t}\binom{N}{eL/2^{t+1}}$$

choices for the pair (I_{t+1}, Λ_{t+1}), for an appropriate constant C_6, and this establishes (4.32).

This completes the proof of Lemma 4.10.

4.8 Conclusion and future work

We have shown that "most" Reed-Solomon codes are list decodable beyond the Johnson bound, answering an open question (Question 4.1) of [43, 56, 94, 108]. More precisely, we have shown that with high probability, a Reed-Solomon code with random evaluation points of rate

$$\Omega\left(\frac{\varepsilon}{\log(q)\log^5(1/\varepsilon)}\right)$$

is list decodable up to a $1-\varepsilon$ fraction of errors with list size $O(1/\varepsilon)$. This beats the Johnson bound whenever $\varepsilon \leq \tilde{O}\left(1/\log(q)\right)$.

Our proof actually applies more generally to randomly punctured codes, and extends the results of Chapter 3 to large alphabets. This provides a positive answer (up to polylogarithmic factors) to our second motivating question, Question 4.2, about whether randomly punctured codes with good distance are optimally list-decodable. As an added corollary, we have obtained improved bounds on the list decodability of random linear codes over large alphabets. Our bounds are nearly optimal (up to polylogarithmic factors), and are the best known whenever $q \gtrsim \log^5(1/\varepsilon)$.

The most obvious open question that remains is to remove the polylogarithmic factors from the rate bound. The factor of $\log(q)$ is especially troublesome: it bites when $q = 2^{\Omega(1/\varepsilon)}$ is very large, but this parameter regime can be reasonable for Reed-Solomon codes. Removing this logarithmic factor seems as though it may require a restructuring of the argument. A second question is to resolve the discrepancy between our upper bound on list sizes and the bound associated with general random codes of the same rate; there is a gap of a factor of ε in the parameter regime $1/\varepsilon \leq q \leq 1/\varepsilon^2$.

To avoid ending the chapter on the shortcomings of our argument, we mention a few hopeful directions for future work. Our argument applies to generally to randomly punctured codes, and in fact to any code with independent symbols. We will explore some generalizations in Chapter 5. Additionally, list decodable codes are connected to many other pseudorandom objects; it would be extremely interesting to explore the ramifications of our argument for random families of extractors or expanders, for instance.

Acknowledgments

The results in this chapter first appeared as [96], which is joint work with Atri Rudra. We are very grateful to Mahmoud Abo Khamis, Venkat Guruswami, Prahladh Harsha, Yi Li, Anindya Patthak, and Martin Strauss for careful proof-reading and helpful comments, and to Swastik Kopparty and Shubhangi Saraf for some discussions which led to the questions considered here.

CHAPTER 5

List decoding: more general applications

In Chapters 3 and 4, we built up a general machinery for proving list-decodability results for randomly punctured codes. In fact, the arguments in those chapters are even more general. In particular, the only things we used were:

- The coordinates of the random code $\mathcal{C}$ are independent, and
- The "expected average-radius-list-decodabiity" of $\mathcal{C}$ is good. In Chapter 3, we controlled this by bounding $\mathbb{E}\|\Phi x\|_1$, and in Chapter 4 we controlled this by bounding the quantity $\mathcal{E}$.

There's nothing special about puncturing codes with good (averaged) distance, and one can imagine a whole host of operations that meet the above two criterion. In this chapter, we develope a more general theory, which will form a new code $\mathcal{C}$ of length n from an old code $\mathcal{C}_0$ of length n_0 by applying a randomized function $f : \mathcal{C}_0 \to \mathcal{C}$; the only requirement will be that the coordinate functions $f_1, \ldots, f_n$ of f are independent and that f behaves decently in expectation.

This encompasses many operations; as examples, we'll consider the case where $f_i(c) = \langle a_i, c\rangle$ for a suitable random vector $a_i \in \mathbb{F}^{n_0}$ *(random inner products);* and the case where $f_i(c) = (c_{j_1^{(i)}}, c_{j_2^{(i)}}, \ldots, c_{j_t^{(i)}}) \in \mathbb{F}_{q^t}$ for a random list of t integers $(j_1^{(i)}, \ldots, j_t^{(i)}) \in [n_0]^t$ *(random folding).* Using these two operations, we'll show:

1. The existence of binary codes that are combinatorially list decodable from $1/2-\varepsilon$ fraction of errors with optimal rate $\Omega(\varepsilon^2)$ that can be encoded in *linear* time.

2. Show that any code with $\Omega(1)$ relative distance when randomly folded (enough times) lead to codes that can be list decoded from $1-\varepsilon$ fraction of errors. This formalizes the intuition for why the folding operation has been successful in obtaining codes with optimal list decoding parameters.

5.1 Introduction

In this chapter we will work in the same regime as Chapters 3 and 4. Namely, we are interested in list-decoding q-ary codes from a $\rho = 1 - 1/q - \varepsilon$ fraction of errors, for small $\varepsilon > 0$. As we have seen in Chapter 2, the best rate one could hope for here is

$$R^*(q,\varepsilon) := 1 - H_q(1-1/q-\varepsilon) \leq \min\left\{\varepsilon, \frac{q\varepsilon^2}{\log(q)}\right\}.$$

For complexity applications it is often enough to design a code with rate $\Omega(R^*(q,\varepsilon))$ with the same error correction capability. We will focus on this parameter regime in the current paper.

Perhaps the ultimate goal of list decoding research in the parameter regime above would be to solve the following:

Problem 5.1. *Construct codes with rate $\Omega(R^*(q,\varepsilon))$ that can correct $1-1/q-\varepsilon$ fraction of errors with linear time encoding and linear time decoding.*[1]

Even though much progress has been made in algorithmic list decoding, we are far from answering Problem 5.1. In particular, if we are happy with polynomial time encoding and decoding and large enough alphabet sizes, then the problem was

[1]One needs to be careful about the machine model when one wants to claim linear runtime. In this chapter we consider the RAM model—for our purposes, it is fine to consider "linear time" to mean "a linear number of $\mathbb{F}_q$ operations," and to assume that the alphabet size is small, say polynomial in $1/\varepsilon$.

solved by Guruswami and Rudra [51] and subsequent works [23, 48, 62–64, 78]. If we are happy with non-algorithmic results, then the work in Chapters 3 and 4 (or, just plain old random codes) gives combinatorial list-decoding guarantees, over any alphabet size.

This chapter generalizes the machinery of Chapters 3 and 4 to make some modest progress on algorithmic questions, and to shed some new light on some of the recent algorithmic developments in list decoding.

5.1.1 Linear time encoding with near optimal rate

We first consider the special case of Problem 5.1 that concentrates on the encoding complexity for binary codes:

Question 5.2. *Do there* exist binary *codes with rate* $\Omega(\varepsilon^2)$ *that can be encoded in linear time and can be (combinatorially) list decoded from* $1/2 - \varepsilon$ *fraction of errors?*

We remark that once we ignore the decoding time, the question above is only interesting when we talk about linear encoding time. Chapter 3 showed that random binary linear codes of rate $R^*(q, \varepsilon)$ are list-decodable from $1/2 - \varepsilon$ fraction of errors; this immediately implies quadratic encoding time. In fact, near linear time encoding with optimal rate also follows from known results: e.g. Guruswami and Rudra [53] showed that folded Reed-Solomon code concatenated with random inner codes (with at most logarithmic block length) achieve the optimal rate and fraction of correctable errors tradeoff. This code is overall near linear time encodable since Reed-Solomon (and hence folded Reed-Solomon) codes can be encoded in near linear time.

However, obtaining linear time encoding with optimal rate is still an open question. For q-ary codes (for q sufficiently large that depends only on ε), Guruswami and Indyk showed that one can get linear time encoding and decoding with near optimal rate but for *unique* decoding [47]. For list decoding, they prove a similar result for list decoding but the rate is exponentially small in $1/\varepsilon$ [46]. This result can be used with code concatenation to give a similar result for binary codes, but also suffers from an exponentially small rate.[2]

5.1.2 Folded codes

The aforementioned result of Guruswami and Rudra [51] showed that if one applied the *folding* operation on Reed-Solomon codes, then the resulting codes (called folded Reed-Solomon codes) can be list decoded in polynomial time with optimal rate. The folding operation is illustrated in Figure 5.1: given a q-ary code $\mathcal{C}_0$ of block length n_0 and a folding parameter t (that divides n_0) and a partition of $[n_0]$ into n_0/t sets of size t positions in them, the new "folded" code $\mathcal{C}$ is the same as $\mathcal{C}_0$ except it is now a q^t-ary code, where each set of t symbols in each of the partitioned sets is now a bigger symbol. For large enough t, and appropriate partitions, this results in codes that can list decode from $1-\varepsilon$ fraction of errors with optimal rate [51, 63, 65] when one starts with Reed-Solomon or more generally certain algebraic-geometric codes.

There is a natural intuition for the effectiveness of the folding operation [51, 52]. Folding effectively reduces the number of error patterns that a decoder has to handle. For example, consider the case when $q = 2$. Consider an error pattern that corrupts a $1-2\varepsilon$ fraction of the *odd* positions (the rest do not have errors). This error pattern must be handled by any decoder which can list decode from $1/2-\varepsilon$ fraction of errors. On the other hand, consider a 2-folding (with partition as above) of the code; now the alphabet size has increased, so we hope to correct $1-1/2^2-\varepsilon = 3/4-\varepsilon$ fraction of errors. However, the earlier error pattern affects a $1-2\varepsilon$ of the new, folded symbols.

[2]We thank Venkat Guruswami for pointing out this fact.

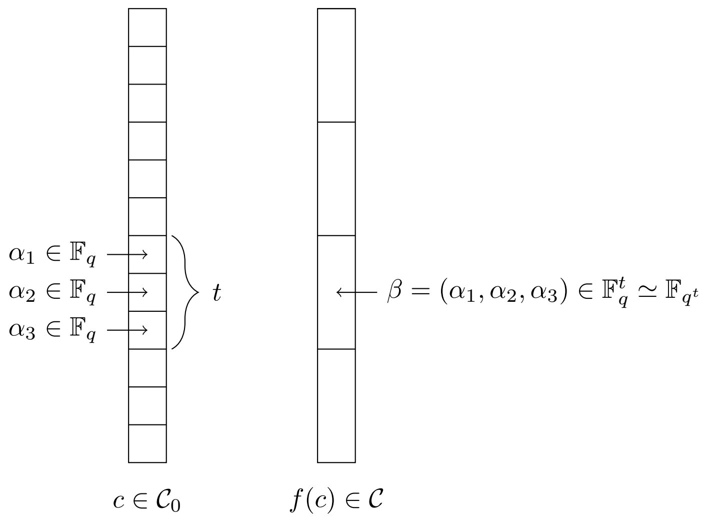

$\alpha_1 \in \mathbb{F}_q$
$\alpha_2 \in \mathbb{F}_q$
$\alpha_3 \in \mathbb{F}_q$
t
$\beta = (\alpha_1, \alpha_2, \alpha_3) \in \mathbb{F}_q^t \simeq \mathbb{F}_{q^t}$
$c \in \mathcal{C}_0$
$f(c) \in \mathcal{C}$

Figure 5.1: The folding operation. $f(c)$ is a folded version of $c \in \mathcal{C}_0$. The folded code $\mathcal{C} \in \mathbb{F}_{q^t}^{n_0/t}$ the image $f(\mathcal{C}_0)$ of $\mathcal{C}_0 \in \mathbb{F}_q^{n_0}$.

Thus, in the folded scenario, an optimal decoder need not handle this error pattern, since $1 - 2\varepsilon > 3/4 - \varepsilon$ (for small enough ε).

In some sense, this intuition is the reason that random codes over large alphabets can tolerate more error than random codes over small alphabets: because the smallest "corruptable unit" is larger when the alphabet is larger, there are fewer error patterns to worry about. Indeed, an inspection of the proof that random codes obtain optimal list-decoding parameters shows that this is the crucial difference. Since a random code over a large alphabet is in fact a folding of a random code over a small alphabet, the story we told above is at work here.

Despite this nice-sounding intuition—which doesn't use anything specific about the code—the arguments for folding of specific codes crucially exploit algebraic properties of the unfolded codes. It is natural to wonder if the intuition above can be made rigorous. In particular,

Question 5.3. *Given any code with distance $\Omega(1)$ and rate $O(\varepsilon)$ does there exist a* *folding (for sufficiently large but constant folding parameter m) such that the resulting code can be (combinatorially) list decoded from $1-\varepsilon$ fraction of errors?*

We note that we do need an $\Omega(1)$ lower bound on the distance of the original code as otherwise it is easy to come up with codes where the answer to the above question is no. The bound of $O(\varepsilon)$ on the rate of the original code is also needed, as folding preserves the rate and the list-decoding capacity theorem implies that any code that can be list decoded from $1-\varepsilon$ fraction of errors must have rate $O(\varepsilon)$.

5.1.3 Contributions of Chapter 5

We generalize the framework of Chapters 3 and 4 to address Problem 5.1. Specifically, we answer both Questions 5.2 and 5.3. This yields modest progress in both linear-time algorithms (in the case of 5.2) and in understanding why (from a philosophical point of view) existing algorithmic techniques work.

Another contribution is the generalization itself. From a technical point of view, this chapter does not contain much mathematics beyond what has been presented in earlier chapters, but it is our hope that the approach of the previous chapters can be applied fruitfully to answer many more algorithmic questions in list decoding.[3]

5.1.4 Chapter organization

In Section 5.2, we will introduce a general framework for the results of the previous two chapters. In Section 5.3, we'll address Question 5.2, and give a family of linear-time encodable binary codes. In Section 5.4, we'll address Question 5.3, and prove that randomly folded codes are optimally list-decodably with high probability. This provides some rigor behind the intuition generally invoked for algorithmic folding results.

[3]...and beyond!

5.2 Setup, and still more definitions

In this chapter, we will intrepret the results of Chapters 3 and 4 as the following intuition:

> If you take a code with alphabet Σ_0 which is list-decodable (enough) up to $\rho_0 = 1 - 1/|\Sigma_0| - \varepsilon$, and do some random (enough) stuff to the symbols, you will obtain a new code (possibly over a different alphabet Σ) which is list-decodable up to $\rho = 1 - 1/|\Sigma| - O(\varepsilon)$.

In order to make this intuition precise, we will recall (and set up) a bit of notation. So far in this dissertation, we have only ever dealt with linear codes, and so it has been convenient to take the alphabet to always be a finite field. We will deviate from this notation slightly, to emphasize that the generalizations in this chapter do not require linearity. Thus, we will consider codes $\mathcal{C} \subset \Sigma^n$ of length n over the alphabet Σ. As usual, the *rate* of $\mathcal{C}$ is defined to be

$$R := \frac{\log_{|\Sigma|}(|\mathcal{C}|)}{n}.$$

For $x, y \in \Sigma^n$, $\delta(x, y)$ is the relative Hamming distance, and $\mathrm{agr}(x, y) := n(1 - \delta(x, y))$ denotes the agreement between x and y. For $x \in \mathbb{F}^n$, $\mathrm{nnz}(x)$ will denote the number of nonzero entries in x.

As in previous chapters, we study the *average-radius list-decodability* of $\mathcal{C}$:

Definition 5.4. *A code $\mathcal{C} \subset \Sigma^n$ is (ρ, L)-average-radius list-decodable if for all sets $\Lambda \subset \mathcal{C}$ with $|\Lambda| = L$,*

$$\max_z \sum_{c \in \Lambda} \mathrm{agr}(c, z) \leq nL\rho.$$

As we have seen, average-radius list-decodability implies the standard notion of list-decodability (Definition 2.3).

In the following, we will always start with some code $\mathcal{C}_0 \in \Sigma_0^{n_0}$ and a distribution $\mathcal{D}$ on functions $f : \mathcal{C}_0 \to \Sigma^n$. We will draw a function f from $\mathcal{D}$, and define $\mathcal{C} \subset \Sigma^n$ to be the image of f. Thus, $\mathcal{C}$ will be a random code, with $|\mathcal{C}| = |\mathcal{C}_0|$.

Now we are ready to make the intuition about precise: we need to define "random enough" and "list-decodable enough." We will make the phrase "random enough" precise in the following definition.

Definition 5.5. *Let $\mathcal{D}$ be a distribution on functions $f : \mathcal{C}_0 \to \Sigma^n$, as above; write such an f as $f(x) = (f_1(x), \ldots, f_n(x))$. We say that $\mathcal{D}$ has* independent symbols *if the f_i are independent for $i = 1, \ldots, n$.*

For example, we may take $f_j(c)$ to be a random symbol from the codeword $c \in \mathcal{C}_0$, chosen independently for each j; this results (up to some abuse of notation about sampling with replacement) in a randomly punctured code. Or, if Σ_0 is a finite field $\mathbb{F}$, we could take $f_j(c) = \langle a_j, c \rangle$ for a independent random vectors $a_j \in \mathbb{F}^n$.

Now, we will quantify what it means to be "list-decodable enough." We introduce a parameter $\mathcal{E} = \mathcal{E}(\mathcal{C}_0, \mathcal{D})$, defined as follows:

$$\mathcal{E}(\mathcal{C}_0, \mathcal{D}) := \max_{\Lambda \subset \mathcal{C}_0, |\Lambda| = L} \mathbb{E}_{f \sim \mathcal{D}} \max_{z \in \Sigma^n} \sum_{c \in \mathcal{C}_0} \mathrm{agr}(f(c), z). \tag{5.1}$$

The quantity $\mathcal{E}$, which is the same as $\mathcal{E}$ from Chapter 4, captures how list-decodable $\mathcal{C}$ is in expectation. Indeed, $\max_z \sum_{c\ in \mathcal{C}_0} \mathrm{agr}(f(c), z)$ is the quantity controlled by average-radius list-decodability (Definition 5.4). To make a statement about the actual average-radius list-decodability of $\mathcal{C}$ (as opposed to in expectation), we will need to understand $\mathcal{E}$ when the expectation and the maximum are reversed:

$$\mathbb{E}_{f \sim \mathcal{D}} \max_{\Lambda \subset \mathcal{C}_0, |\Lambda| = L} \max_{z \in \Sigma^n} \sum_{c \in \mathcal{C}_0} \mathrm{agr}(f(c), z).$$

In this notation, we can combine Theorems 3.2 and 4.6 in the following statement:

Theorem 5.6. *[Follows from Theorems 3.2 and 4.6] Let $\mathcal{C}_0, \mathcal{D}$ and $\mathcal{C}$ be as above, and suppose that $\mathcal{D}$ has independent symbols. Fix $\varepsilon > 0$. Then*

$$\mathbb{E}_f \max_{z \in \Sigma^n} \max_{\Lambda \subset \mathcal{C}_0, |\Lambda| = L} \sum_{c \in \Lambda} \operatorname{agr}(f(c), z) \leq \mathcal{E} + Y + \sqrt{\mathcal{E} Y},$$

where

$$Y = CL \log(N) \log^5(L)$$

for an absolute constant C. For $|\Sigma| = 2$, we have

$$\mathbb{E}_f \max_{x \in \Sigma^n} \max_{\Lambda \subset \mathcal{C}_0, |\Lambda| = L} \sum_{c \in \Lambda} \operatorname{agr}(f(c), z) \leq \mathcal{E} + CL\sqrt{n \ln(N)}.$$

Theorem 5.6 makes the intuition above more precise: Any "random enough" operation (that is, an operation with independent symbols) of a code with good "average-radius list-decodability" (that is, good $\mathcal{E}(\mathcal{C}_0, \mathcal{D})$) will result in a code which is also list-decodable.

In this work, we answer Questions 5.2 and 5.3 by coming up with useful distributions $\mathcal{D}$ on functions f and computing the parameter $\mathcal{E}$. To control $\mathcal{E}$, we will make use of some average-radius Johnson bounds that we've already encountered: Theorems 2.8, 4.3, and 4.4. For the reader's convenience, we restate these bounds here.

Theorem 5.7 (Average-radius Johnson bounds)**.** *Let $\mathcal{C} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ be any code. Then for all $\Lambda \subset \mathbb{F}_q^k$ of size L and for all $z \in \mathbb{F}_q^n$:*

- *If $q = 2$,*

$$\sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) \leq \frac{n}{2}\left(L + \sqrt{L^2 - 2\sum_{x \neq y \in \Lambda} d(\mathcal{C}(x), \mathcal{C}(y))} \right).$$

- *For all $\varepsilon \in (0, 1)$,*

$$\sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) \leq \frac{nL}{q} + \frac{nL}{2\varepsilon}\left(1 + \varepsilon^2\right)\left(1 - \frac{1}{q}\right) - \frac{n}{2L\varepsilon} \sum_{x \neq y \in \Lambda} d(\mathcal{C}(x), \mathcal{C}(y)).$$

- $$\sum_{x \in \Lambda} \operatorname{agr}(\mathcal{C}(x), z) \leq \frac{1}{2}\left(n + \sqrt{n^2 + 4n^2 L(L-1) - 4n^2 \sum_{x \neq y \in \Lambda} d(\mathcal{C}(x), \mathcal{C}(y))} \right).$$

5.3 Efficiently encodable list-decodable codes from expander graphs

In this section, we answer Question 5.2, and give linear-time encodable binary codes with the optimal trade-off between rate and list-decoding radius.

Theorem 5.8. *There is a randomized construction of binary codes $\mathcal{C} \in \mathbb{F}_2^n$ so that the following hold with probability $1 - o(1)$, for any sufficiently small ε and any sufficiently large n.*

1. *$\mathcal{C}$ is encodable in time $O(n \ln(1/\varepsilon))$.*
2. *$\mathcal{C}$ is (ρ, L)-average-radius list-decodable with $\rho = \frac{1}{2}(1 - C\varepsilon)$ and $L = \varepsilon^{-2}$, where C is an absolute constant.*
3. *$\mathcal{C}$ has rate $\Omega(\varepsilon^2)$.*

The rest of this section is devoted to the proof of Theorem 5.8. Our codes will work as follows. We begin with a linear-time encodable code with constant rate and constant distance; we will use Spielman's variant on expander codes [100, Theorem 19]. These codes have rate 1/4, and distance $\delta_0 \geq 0$ (a small positive constant). In this case, a random puncturing of $\mathcal{C}_0$ (as in the previous chapters) will not work, as $\mathcal{C}_0$ does not have good enough distance. Instead, we will use a different operation, which can be viewed as a generalization of puncturing: we will take random inner products with vectors of weight t.

Definition 5.9 (Random t-wise XOR). *Let $\mathcal{C}_0 \in \mathbb{F}_2^{n_0}$. Choose $t \leq n_0$. For $v \in \mathbb{F}_2^{n_0}$ with $\mathrm{nnz}(v) = t$, define $f_v : \mathbb{F}_2^{n_0} \to \mathbb{F}_2$ by $f_v(c) = \langle v, c \rangle$. Define a distribution $\mathcal{D}_{ip}(t)$ on functions $f : \mathcal{C}_0 \to \mathbb{F}_2^n$ by choosing $v_1, \ldots, v_n$ independently, uniformly at random with replacement from $\{v \in \mathbb{F}_2^{n_0} : \mathrm{nnz}(v) = t\}$, and setting $f = (f_{v_1}, f_{v_2}, \ldots, f_{v_n})$. We call this distribution* random t-wise inner product.

We will choose $\mathcal{C}$ to be the ensemble of codes arising from $\mathcal{C}_0$ and $\mathcal{D}_{ip}(t)$ for $t = 4\ln(1/\varepsilon)\delta_0^{-1}$. We first verify Item 1 of Theorem 5.8, that $\mathcal{C}$ is linear-time encodable. Indeed, we have

$$\mathcal{C}(x) = A\mathcal{C}_0(x),$$

where $A \in \mathbb{F}_2^{n \times n_0}$ is a matrix whose rows are the vectors v_i, which have $\mathrm{nnz}(v_i) \leq t$. In particular, the time to multiply by A is $nt = O(n \ln(1/\varepsilon))$, as claimed.

To verify Item 2 about the list-decodability, we begin by computing the quantity $\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{ip}(t))$.

Lemma 5.10. *Let $\mathcal{C}_0 \in \mathbb{F}_2^{n_0}$ be a code with distance δ_0, and suppose $t \geq \frac{4\ln(1/\varepsilon)}{\delta_0}$. Then*

$$\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{ip}(t)) \leq \frac{n}{2}\left(L(1+\varepsilon) + \sqrt{L}\right).$$

Proof. We will use the average-radius Johnson bound, Theorem 5.7, Item 1. Thus, we start by computing the expected distance between two symbols of the code $\mathcal{C} \in \mathbb{F}_2^n$

obtained from $\mathcal{C}_0$ and $\mathcal{D}_{ip}(t)$. Let c, c' denote two distinct codewords in $\mathcal{C}_0$. Then

$$\begin{aligned}
\mathbb{E}\delta(f(c), f(c')) &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{P}\left\{f_i(c) \neq f_i(c')\right\} \\
&= \mathbb{P}\left\{\langle a_i, c\rangle \neq \langle a_i, c'\rangle\right\} \\
&= \frac{1}{2}\mathbb{P}\left\{(c - c')_{\mathrm{Supp} a_i} \neq 0\right\} \\
&= \frac{1}{2}\left(1 - (1-\delta_0)^t\right) \\
&\leq \frac{1}{2}\left(1 - e^{-\delta_0 t/2}\right).
\end{aligned}$$

In particular, if $t = \frac{4\ln(1/\varepsilon)}{\delta_0}$, then this is $\frac{1}{2}(1-\varepsilon^2)$. Then Theorem 5.7 implies that

$$\begin{aligned}
\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{ip}(t)) &= \max_{\Lambda\subset\mathcal{C}_0} \mathbb{E}_{f\sim\mathcal{D}_{ip}(t)} \max_{z\in\mathbb{F}_2^n} \sum_{c\in\Lambda} \mathrm{agr}(f(c), z) \\
&\leq \max_{\Lambda} \mathbb{E}_f \max_{z\in\mathbb{F}_2^n} \frac{n}{2}\left(L + \sqrt{L^2 - 2\sum_{c\neq c'\in\Lambda} \delta(f(c), f(c'))}\right) \\
&\leq \max_{\Lambda} \frac{n}{2}\left(L + \sqrt{L^2 - 2\sum_{c\neq c'\in\Lambda} \mathbb{E}_f\delta(f(c), f(c'))}\right) \\
&\leq \frac{n}{2}\left(L + \sqrt{L^2 - 2\sum_{c\neq c'\in\Lambda} \frac{1}{2}(1-\varepsilon^2)}\right) \\
&= \frac{n}{2}\left(L + \sqrt{L^2\varepsilon^2 + L(1-\varepsilon^2)}\right) \\
&\leq \frac{n}{2}\left(L(1+\varepsilon) + \sqrt{L}\right).
\end{aligned}$$

$\square$

Thus, Theorem 5.6 implies that with constant probability,

$$\begin{aligned}
\max_{z\in\mathbb{F}_2^n} \max_{\Lambda\subset\mathcal{C}, |\Lambda| = L} \frac{1}{L}\sum_{c\in\Lambda} \mathrm{agr}(c, z) &\leq \frac{\mathcal{E}}{L} + C\sqrt{n\ln(N)} \\
&\leq \frac{n}{2}\left(1 + \varepsilon + \frac{1}{\sqrt{L}}\right) + C\sqrt{n\ln N}.
\end{aligned}$$

In particular, if $C\sqrt{n\ln N} \leq \varepsilon n$, then in the favorable case $\mathcal{C}$ is $(\rho, L-1)$-average-radius list-decodable, for $L = \varepsilon^{-2}$ and $\rho = 1/2(1 - C'\varepsilon)$ for some constant C'.

It remains to verify Item 3, about the rate R of $\mathcal{C}$. Notice that if $|\mathcal{C}| = N$, then we are done, because then the requirement $C\sqrt{n\ln(N)} \leq \varepsilon n$ reads

$$R = \frac{\log_2(N)}{n} \leq \frac{\varepsilon^2}{C\ln(2)}.$$

Thus, to complete the proof we will argue that f is injective with high probability, and so in the favorable case $|\mathcal{C}| = N$. Fix $c \neq c' \in \mathcal{C}_0$. Then, by the same computations as above,

$$\mathbb{P}\left\{f(c) = f(c')\right\} = \left(\frac{1}{2}\left(1 + (1-\delta_0)^t\right)\right)^n \leq \left(\frac{1+\varepsilon^2}{2}\right)^n.$$

Using the fact that we will choose $n \geq C\ln(N)/\varepsilon^2$, the right hand side is

$$\left(\frac{1+\varepsilon^2}{2}\right)^{C\ln(N)/\varepsilon^2} = N^{-\ln\left(\frac{2}{1+\varepsilon^2}\right)C/\varepsilon^2} \leq N^{-3}$$

for sufficiently small ε. Thus, by the union bound on the $\binom{N}{2} \leq N^2$ choices for the pairs of distinct codewords (c, c'), we see that $\mathbb{P}\left\{|\mathcal{C}| < N\right\} \leq 1/N$, which is $o(1)$ as desired. This completes the proof of Theorem 5.8.

Remark 10 (Random inner products for $q > 2$)**.** *For this application, $q = 2$ is the interesting case. However, the argument above works just fine for $q > 2$. In this case, we define $f_v(c) = \langle v, c\rangle$ for v uniform in $\left\{v \in \mathbb{F}_q^{n_0} : \mathrm{nnz}(a) = t\right\}$, and define $\mathcal{D}_{ip}(t)$ as before. We may use the first statement of Theorem 5.6, and statements 2 or 3 of Theorem 5.7 for the average-radius Johnson bound.*

5.4 Random folding

In this section, we answer Question 5.3, and show that every code with good distance has a folding which is optimally list-decodable. We must first define the "random folding" operation.

Definition 5.11 (Random t-wise folding)**.** *Let $\mathcal{C}_0 \in \Sigma_0^{n_0}$. Choose $t \leq n_0$, and let $\Sigma = \Sigma_0^t$. For $S \subset [n_0]$ with $|S| = t$, define $f_S : \Sigma_0^{n_0} \to \Sigma^n$ by $f_S(c) = (c_i)_{i \in S}$. Define a distribution $\mathcal{D}_{fold}(t)$ on functions $f : \mathcal{C}_0 \to \Sigma^n$ by choosing $S_1, \ldots, S_n$ independently, uniformly at random with replacement from $\{S \subset [n_0] : |S| = t\}$, and setting $f = (f_{S_1}, f_{S_2}, \ldots, f_{S_n})$. We call this distribution* random t-wise folding.

Remark 11 (Definition 5.11 vs. standard folding)**.** *The definition above is slightly different from a uniformly random t-wise folding, which would correspond to a random partition of $[n_0]$ into n pieces of size t. Because the elements for each of the symbols are chosen with replacement, it's possible that the new symbols "overlap" slightly, and that other symbols from the original code are not represented at all in $\mathcal{C}_0$. However, sampling with replacement makes the computations significantly simpler. Since the goal of this section is to provide some rigor behind the intuition discussed around Question 5.3, we will go with the simpler case.*

Theorem 5.12 below analyzes folding in two parameter regimes. In the first parameter regime, we address Question 5.3, and we consider t-wise folding where $n_0 = nt$. In this case, the folded code $\mathcal{C}$ will have the same rate as the original code $\mathcal{C}_0$, and so in order for $\mathcal{C}$ to be list-decodable up to radius $1 - \varepsilon$, the rate R_0 of $\mathcal{C}_0$ must be $O(\varepsilon)$. Item 1 shows that if this necessary condition is met (with some logarithmic slack), then $\mathcal{C}$ is indeed list-decodable up to $1 - \varepsilon$. In the second parameter regime, we consider what can happen when the rate R_0 of $\mathcal{C}_0$ is significantly larger. In this case, we cannot hope to take n as small as n_0/t and hope for list-decodability up to $1 - \varepsilon$. The second part of Theorem 5.12 shows that we may take n nearly as small as the list-decoding capacity theorem allows.

Theorem 5.12. *There are constants C_i, $i = 0, \ldots, 5$, so that the following holds.*

Suppose $q > 1/\varepsilon^2$. Let $\mathcal{C}_0 \subset \mathbb{F}_q^{n_0}$ be a code with distance $\delta_0 \geq C_2 > 0$.

1. *Suppose $t \geq C_0 \log(1/\varepsilon) \geq 4\ln(1/\varepsilon)/\delta_0$. Suppose that $\mathcal{C}_0$ has rate*

$$R_0 \leq \frac{C_1\varepsilon}{\log(q) t \log^5(1/\varepsilon)}.$$

Let $\mathcal{C} \subset \mathbb{F}_{q^t}$ be a random t-wise folding of $\mathcal{C}_0$ of length $n = n_0/t$. Then with high probability, $\mathcal{C}$ is $(1 - C_3\varepsilon, 1/\varepsilon)$-average-radius list-decodable, and further the rate R of $\mathcal{C}$ satisfies $R = R_0$.

2. *Suppose that $t \geq 4\ln(1/\varepsilon)/\delta_0$, and suppose that $\mathcal{C}_0$ has rate R_0 so that*

$$R_0 \leq \left(\frac{nt}{n_0}\right)\left(\frac{\log(1/\varepsilon)}{\log(q)}\right).$$

Let $\mathcal{C}$ be a random t-wise folding of $\mathcal{C}_0$ of length

$$n \geq \frac{\log(N)\log(1/\varepsilon)}{\varepsilon}.$$

Then with high probability, $\mathcal{C}$ is $(1 - C_4\varepsilon, 1/\varepsilon)$-average-radius list-decodable, and the rate R of $\mathcal{C}$ is at least

$$R \geq \frac{C_5\varepsilon}{t\log(q)\log^5(1/\varepsilon)}.$$

The rest of this section is devoted to the proof of Theorem 5.12. As before, it suffices to control $\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{fold}(t))$, which we do via the average-radius Johnson bound (Theorem 5.7). Because we are interested in the parameter regime where $q \geq 1/\varepsilon^2$, we use the third statement in Theorem 5.7.

Suppose $t \geq 4\ln(1/\varepsilon)/\delta_0$ and set $L = 1/\varepsilon$. For $c \neq c' \in \mathcal{C}_0$, we compute

$$\begin{aligned}
\mathbb{E}_{f \sim \mathcal{D}_{fold}(t)} \delta(f(c), f(c')) &= \frac{1}{n} \sum_{i=1}^{n} \mathbb{P}\left\{ f_j(c) \neq f_j(c') \right\} \\
&= \mathbb{P}\left\{ \exists j \in S_i \,:\, c_j \neq c'_j \right\} \\
&= 1 - (1 - \delta_0)^t \\
&\leq 1 - \varepsilon^2,
\end{aligned}$$

using the choice of t in the final line. Thus, by Theorem 5.7, Item 3,

$$\begin{aligned}
\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{fold}(t)) &= \max_{\Lambda \subset \mathcal{C}_0} \mathbb{E}_{f \sim \mathcal{D}_{fold}(t)} \max_{z \in \mathbb{F}_q^n} \sum_{c \in \Lambda} \mathrm{agr}(f(c), z) \\
&\leq \max_{\Lambda \subset \mathcal{C}_0} \mathbb{E}_{f \sim \mathcal{D}_{fold}(t)} \max_{z \in \mathbb{F}_q^n} \frac{1}{2} \left(n + \sqrt{n^2 + 4n^2 L(L-1) - 4n^2 \sum_{c \neq c' \in \Lambda} \delta(f(c), f(c'))} \right) \\
&= \max_{\Lambda \subset \mathcal{C}_0} \frac{1}{2} \left(n + \sqrt{n^2 + 4n^2 L(L-1) - 4n^2 \sum_{c \neq c' \in \Lambda} \mathbb{E}_f \delta(f(c), f(c'))} \right) \\
&\leq \frac{1}{2} \left(n + \sqrt{n^2 + 4n^2 L(L-1) - 4n^2 \sum_{c \neq c' \in \Lambda} (1 - \varepsilon^2)} \right) \\
&= \frac{n}{2} \left(1 + \sqrt{1 + 4L(L-1)\varepsilon^2} \right) \\
&\leq Cn,
\end{aligned}$$

using the choice of L and defining $C = (1 + \sqrt{5})/2$. Then, by Theorem 5.6, recalling that

$$Y = CL \log(N) \log^5(L),$$

and $N = |\mathcal{C}_0|$, we have with high probability that

$$\begin{aligned}
\mathbb{E}_f \max_{z \in \Sigma^n} \max_{\Lambda \subset \mathcal{C}_0, |\Lambda| = L} \sum_{c \in \Lambda} \mathrm{agr}(f(c), z) &\leq \mathcal{E}(\mathcal{C}_0, \mathcal{D}_{fold}(t)) + Y + \sqrt{\mathcal{E}(\mathcal{C}_0, \mathcal{D}_{fold}(t) Y} \\
&\leq O\left(L \log(N) \log^5(L) + n \right).
\end{aligned}$$

In the favorable case,

(5.2)

$$\mathbb{E}_f \max_{z\in\Sigma^n} \max_{\Lambda\subset\mathcal{C},|\Lambda|=L} \frac{1}{L}\sum_{c\in\Lambda} \mathrm{agr}(c,z) \le O\left(\log(N)\log^5(L) + n/L\right) = O\left(\log(N)\log^5(1/\varepsilon) + n\varepsilon\right).$$

As before, $\mathcal{C}$ is $(1 - C\varepsilon, L-1)$ average-radius list-decodable, for some constant C, as long as the right hand side is no more than $O(n\varepsilon)$. This holds as long as

$$\log(N)\log^5(1/\varepsilon) \le n\varepsilon. \tag{5.3}$$

Equation (5.3) holds for any choice of n. First, we prove item 1 and we focus on the case that $n_0 = nt$; this mimics the parameter regime the standard definition of folding. Given $n_0 = nt$, we can translate (5.3) into a condition on R_0, the rate of $\mathcal{C}_0$. We have

$$R_0 = \frac{\log_q(N)}{n_0} = \frac{\log_q(N)}{nt},$$

and so translating (5.3) into a requirement on $R(\mathcal{C}_0)$, we see that as long as

$$R_0 \lesssim \frac{\varepsilon}{\log(q)t\log^5(1/\varepsilon)} \lesssim \frac{\varepsilon}{\log(q)\log^6(1/\varepsilon)},$$

then with high probability $\mathcal{C}$ is $(1 - C\varepsilon, L)$-list-decodable. Choose n so that this holds.

It remains to verify that the rate R of $\mathcal{C}$ is the same as the rate R_0 of $\mathcal{C}_0$. For standard folding, it is immediate that the rate of the code does not change. With our slightly randomized tweak on it (Definition 5.11), this requires some argument: it might be the case that $|\mathcal{C}| < N$, in which case the rate would decrease.

Claim 5.13. *With $\mathcal{C}_0$ as above and with $n_0 = nt$, $|\mathcal{C}| = N$ with probability at least $1 - o(1)$.*

Proof. The only way that $|\mathcal{C}| < N$ is if two codewords $c \neq c' \in \mathcal{C}_0$ collide, that is, if $f(c) = f(c')$. This is unlikely: we have

$$\mathbb{P}\left\{f(c) = f(c')\right\} = (1-\delta_0)^{nt} \leq \varepsilon^{2nt}.$$

By a union bound over $\binom{N}{2} \leq N^2$ pairs $c \neq c'$, we conclude that the probability that $|\mathcal{C}| < N$ is at most

$$\mathbb{P}\left\{|\mathcal{C}| < N\right\} \leq N^2 \varepsilon^{2nt}. \tag{5.4}$$

If $nt = n_0$, we have

$$\mathbb{P}\left\{|\mathcal{C}| < N\right\} \leq q^{2n_0 R_0} \varepsilon^{2nt} = \left(q^{R_0}\varepsilon\right)^{2n_0}.$$

In particular, when $q^{R_0} < 1/\varepsilon$, this is $o(1)$. By our assumption, $R_0 < \varepsilon$, and so this is always true for sufficiently small ε. □

By a union bound, with high probability both the favorable event (5.2) occurs, and Claim 5.13 holds. In this case, $\mathcal{C}$ is $(1 - C\varepsilon, L)$-list-decodable, and the rate R of $\mathcal{C}$ is

$$R = R_0.$$

Next, we consider a general case, where we may choose $n < n_0/t$, thus increasing the rate. It remains true that as long as (5.3) holds, then $\mathcal{C}$ is $(1 - C\varepsilon, L)$-list-decodable. Again translating the condition (5.3) into a condition on $\log_{q^t}(N)/n$, we see that as long as

$$\frac{\log_{q^t}(N)}{n} \leq \frac{\varepsilon}{t\log(q)\log^5(1/\varepsilon)}, \tag{5.5}$$

then $\mathcal{C}$ is $(1 - C\varepsilon, L)$-list-decodable. Now we must verify that the left-hand-side of (5.5) is indeed the rate R of $\mathcal{C}$, that is, that $|\mathcal{C}| = N$.

Claim 5.14. *With $\mathcal{C}_0$ as above and with n arbitrary, $|\mathcal{C}| = N$ with probability at least $1 - o(1)$.*

Proof. As in (5.4), we have

$$\mathbb{P}\left\{|\mathcal{C}| < N\right\} \leq N^2 \varepsilon^{2nt}.$$

We may bound the right-hand-side by

$$N^2 \varepsilon^{2nt} = \left(q^{R_0 n_0/n} \varepsilon^t\right)^{2n},$$

and for this to be $o(1)$, it is sufficient for

$$R_0 \leq \left(\frac{nt}{n_0}\right)\left(\frac{\log(1/\varepsilon)}{\log(q)}\right),$$

which was our assumption for part 2 of the theorem. □

Now, recalling our choice of n in (5.5), with high probability both (5.2) occurs and Claim 5.14 holds. In the favorable case, $\mathcal{C}$ is $(1 - C\varepsilon, L)$-list-decodable, as long as the rate R satisfies

$$R = \frac{\log_{q^t}(|\mathcal{C}|)}{n} = \frac{\log_{q^t}(N)}{n} \leq \frac{C\varepsilon}{t \log^5(1/\varepsilon)\log(q)}.$$

This completes the proof of Theorem 5.12.

5.5 Conclusion

We generalized the results of Chapters 3 and 4 to a large class of random operations, beyond just random puncturing. The purpose of these generalizations (beyond generalization for generalization's sake) was to begin to bridge the gap between the combinatorial statements of the preceding chapters and the algorithmic statements that dominate the list decoding literature. First, we used our new framework to obtain families of linear-time-encodable binary codes. Second, we used our framework to provide some insight to a successful algorithmic technique, namely, folding. Informal combinatorial arguments are often invoked as an intuition for folding, but making these rigorous has proved challenging. We made this combinatorial intuition more precise, and showed that a random folding of any code with nontrivial distance and appropriate rate is nearly optimally list-decodable with high probability.

Acknowledgments

This chapter is based on ongoing work with Atri Rudra. We thank Swastik Kopparty and Shubhangi Saraf for initial discussions on the two main questions considered in this paper (and for indeed suggesting the random XOR as an operation to consider).

CHAPTER 6

Local decoding: expander codes

In this chapter, we switch gears from list decoding to local decoding. We discussed locally decodable codes in Chapter 2. The idea is that Bob must work extremely quickly—so quickly that he doesn't have time to look at the entire codeword. We will focus on *expander codes,* which we introduced in Chapter 2.

We will present a local-decoding (actually, local-correcting) algorithm for expander codes. Our codes will have rate approaching 1. Bob will make n^ε queries (where n is the block length of the code) for an arbitrarily small constant ε, and he'll be able to handle a constant fraction of errors. In addition to providing new locally correctible codes in this regime (joining two existing constructions, multiplicity codes [79] and lifted codes [41]), this gives a sublinear-time decoding algorithm for expander codes.

Our techniques are rather different than they have been in previous chapters. Before, we had to take a union bound over several (related) events that were not sufficiently unlikely. Now, we will still have to take a union bound over not-improbable-enough events, but no amount of clever union-bounding will save us. Instead, we will see how to deal with the situation algorithmically.

6.1 Introduction

Expander codes, introduced in [98], are linear codes which are notable for their efficient decoding algorithms. In this paper, we show that when appropriately instantiated, expander codes are also *locally decodable,* and we give a sublinear time local-decoding algorithm.

We introduced locally decodable codes in Chapter 2. As in the standard model of coding theory, Alice encodes a message $x \in \mathbb{F}_q^k$ as a codeword $c \in \mathbb{F}_q^n$, and transmits it to Bob across a (malicious) noisy channel. Bob's goal is to recover x from the corrupted codeword w. Decoding algorithms typically process all of w and in turn recover all of x. The goal of local decoding is to recover only a single symbol of x, with the benefit of querying only a few bits of w. The number of symbols of w needed to recover a single bit x is known as the *query complexity*, and we will denote this by Q. The important trade-off in local decoding is between query complexity and the rate $R = k/n$ of the code. When Q is constant or even logarithmic in k, the best known codes have rates which tend to zero as n grows. The first locally decodable codes to achieve sublinear locality and rate approaching one were the multiplicity codes of Kopparty, Saraf and Yekhanin [79]. Prior to this work, only two constructions of locally decodable codes were known with sublinear locality and rate approaching one [41,79]. In this paper, we show that expander codes provide a third construction of efficiently locally decodable codes with rate approaching one.

6.1.1 Notation and preliminaries

Before we state our main results, we set notation and give a few definitions. We will construct linear codes $\mathcal{C}$ of length n and message length k, over a finite field $\mathbb{F} = \mathbb{F}_q$ That is, $\mathcal{C} \subset \mathbb{F}^n$ is a linear subspace of dimension k. As usual, the *rate* of $\mathcal{C}$ is the ratio $R = k/n$. We will also use *expander graphs;* we will give a brief introduction to expanders in Section 6.2. For $n \in \mathbb{Z}$, $[n]$ denotes the set $\{1, 2, \ldots, n\}$. For $x, y \in \mathbb{F}^N$, $\delta(x, y)$ denotes relative Hamming distance. In contrast with previous chapters, we will use $x[i]$, rather than x_i, to denote the i^{th} symbol of x. The reason for the switch is that this chapter will be somewhat more subscript-heavy than previous ones. For $x \in \mathbb{F}^n$ and $S \subset [n]$, we will use $x|_S$ to denote x restricted to symbols indexed by S.

We recall Definitions 2.9 and 2.10 of locally decodable and locally correctable codes. A code (along with an encoding algorithm) is *locally decodable* if there is an algorithm which can recover a symbol $x[i]$ of the message, making only a few queries to the received word.

Definition 6.1 (Locally Decodable Codes (LDCs)). *Let $\mathcal{C} \subset \mathbb{F}^n$ be a code of size $|\mathbb{F}|^k$, and let $E : \mathbb{F}^k \to \mathbb{F}^n$ be an encoding map. Then $(\mathcal{C}, E)$ is (Q, ρ)-locally decodable with error probability η if there is a randomized algorithm Δ, so that for any $w \in \mathbb{F}^b$ with $\Delta(w, E(x)) < \rho$, for each $i \in [k]$,*

$$\mathbb{P}\{\Delta(w, i) = x[i]\} \geq 1 - \eta,$$

and further Δ accesses at most Q symbols of w. Here, the probability is taken over the internal randomness of the decoding algorithm R.

In this work, we will actually construct *locally correctable codes,* which we will see below imply locally decodable codes.

Definition 6.2 (Locally Correctable Codes (LCCs)). *Let $\mathcal{C} \subset \mathbb{F}^n$ be a code, and let $E : \mathbb{F}^k \to \mathbb{F}^n$ be an encoding map. Then $\mathcal{C}$ is (Q, ρ)-locally correctable with error probability η if there is a randomized algorithm, Δ, so that for any $w \in \mathbb{F}^n$ with*

$\Delta(w, E(x)) < \rho$, for each $j \in [n]$,

$$\mathbb{P}\left\{\Delta(w, j) = w[j]\right\} \geq 1 - \eta,$$

and further Δ accesses at most Q symbols of w. Here, the probability is taken over the internal randomness of the decoding algorithm Δ.

The difference between locally correctable codes and locally decodable codes is that locally correctable codes can recover symbols of the *codeword* while locally decodable codes recover symbols of the *message*.

When there is a constant $\rho > 0$ and a failure probability $\eta = o(1)$ so that $\mathcal{C}$ is (Q, ρ)-locally correctable with error probability η, we will simply say that $\mathcal{C}$ is locally correctable with query complexity Q (and similarly for locally decodable).

When $\mathcal{C}$ is a linear code, writing the generator matrix in systematic form gives an encoding function $E : \mathbb{F}^k \to \mathbb{F}^n$ so that for every $x \in \mathbb{F}^k$ and for all $i \in [k]$, $E(x)[i] = x[i]$. In particular, if $\mathcal{C}$ is a (Q, ρ) linear LCC, then $(E, \mathcal{C})$ is a (Q, ρ) LDC. Because of this connection, we will focus our attention on creating locally correctable linear codes.

Many LCCs work on the following principle: suppose, for each $i \in [N]$, there is a set of Q query positions $S(i)$, which are *smooth*—that is, each query is almost uniformly distributed within the codeword—and a method to determine $c[i]$ from $\{c[j] : j \in S(i)\}$ for any uncorrupted codeword $c \in \mathcal{C}$. If Q is constant, this *smooth local reconstruction algorithm* yields a local correction algorithm: with high probability none of the locations queried are corrupted. In particular, by a union bound, the smooth local reconstruction algorithm is a local correction algorithm that fails with probability at most $\rho \cdot Q$. This argument is effective when $Q = \mathcal{O}(1)$; however, when Q is merely sublinear in n, as is the case for us, this reasoning fails. This paper demonstrates how to turn codes which only possess a local reconstruction procedure (in the noiseless setting) into LCCs with constant rate and sublinear query complexity.

Definition 6.3 (Smooth reconstruction)**.** *For a code $\mathcal{C} \subset \mathbb{F}^n$, consider a pair of algorithms (S, A), where S is a randomized query algorithm with inputs in $[n]$ and outputs in 2^n, and $A : \mathbb{F}^Q \times [n] \to \mathbb{F}$ is a deterministic reconstruction algorithm. We say that (S, A) is a s-*smooth local reconstruction algorithm *with query complexity Q if the following hold.*

1. *For each $i \in [n]$, the query set $S(i)$ has $|S(i)| \leq Q$.*
2. *For each $i \in [n]$, there is some set $B \subset [N]$ of size s, so that each query in $S(i)$ is uniformly distributed in B.*
3. *For all $i \in [n]$ and for all codewords $c \in \mathcal{C}$, $A(c|_{S(i)}, i) = c[i]$.*

If $s = n$, then we say the reconstruction is perfectly smooth, since all symbols are equally likely to be queried. Notice that the queries need not be independent. The codes we consider in this work decode a symbol indexed by $x \in \mathbb{F}^m$ by querying random subspaces through x (but not x itself), and thus will have $s = n - 1$.

6.1.2 Related work

The first local-decoding procedure for an error-correcting code was the majority-logic decoder for Reed-Muller codes proposed by Reed [89]. Local-decoding procedures have found many applications in theoretical computer science including proof-checking [5,82,88], self-testing [16,33,34] and fault-tolerant circuits [99]. While these applications implicitly used local-decoding procedures, the first explicit definition of locally decodable codes did not appear until later [75]. For an excellent survey of locally decodable codes, we refer the reader to [114]. The study of locally decodable codes focuses on the trade-off between rate (the ratio of message length to codeword length) and query complexity (the number of queries made by the decoder). Research in this area is separated into two distinct areas: the first seeks to minimize the query complexity, while the second seeks to maximize the rate. In the low-query-complexity regime, Yekhanin was the first to exhibit codes with a constant number of queries and a subexponential rate [113]. Following Yekhanin's work, there has been significant progress in constructing locally decodable codes with constant query-complexity [10,11,18,22,25,26,71,113]. On the other hand, in the high-rate regime, there has been less progress. In 2011, Kopparty, Saraf and Yekhanin introduced *multiplicity codes*, the first codes with a sublinear local-decoding algorithm [79] and rate approaching one. Like Reed-Muller codes, multiplicity codes treat the message as a multivariate polynomial, and create codewords by evaluating the polynomial at a sequence of points. Multiplicity codes are able to improve on the performance of Reed-Muller codes by also including evaluations of the partial derivatives of the message polynomial in the codeword. A separate line of work has developed high-rate locally decodable codes by "lifting" shorter codes [41]. The work of Guo, Kopparty and Sudan takes a short code $\mathcal{C}_0$ of length $|\mathbb{F}|^t$, and lifts it to a longer code $\mathcal{C}$, of length $|\mathbb{F}|^m$ for $m > t$ over $\mathbb{F}$, such that every restriction of a codeword in $\mathcal{C}$ to an affine subspace of dimension t yields a codeword in $\mathcal{C}_0$. The definition provides a natural local-correcting procedure for the outer code: to decode a symbol of the outer code, pick a random affine subspace of dimension t that contains the symbol, read the coordinates and decode the resulting codeword using the code $\mathcal{C}_0$. Guo, Kopparty and Sudan show how to lift explicit inner codes so that the outer code has constant rate and query complexity n^{ε}.

In this work, we show that *expander codes* can also give locally decodable codes with rate approaching one, and with query complexity n^ε. Expander codes, introduced by Sipser and Spielman [98], are formed by choosing a d-regular expander graph, G on n vertices, and a code $\mathcal{C}_0$ of length d (called the *inner code*), and defining the codeword to be all assignments of symbols to the edges of G so that for every vertex in G, its edges form a codeword in $\mathcal{C}_0$. We discussed this construction (for general graphs) in Chapter 2. The connection between error-correcting codes and graphs was first noticed by Gallager [32] who showed that a random bipartite graph induces a good error-correcting code. Gallager's construction was refined by Tanner [105], who suggested the use of an inner code. Sipser and Spielman [98] were the first to consider this type of code with an expander graph (which we will formally define in Section 6.2 below). Spielman [100] showed that these *expander codes* could be encoded and decoded in linear time. Spielman's work provided the first family of error-correcting codes with linear-time encoding and decoding procedures. The decoding procedure has since been improved by Barg and Zemor [7–9, 115].

6.1.3 Contributions of Chapter 6

We show that certain expander codes can be efficiently locally decoded, and we instantiate our results to obtain novel families of (n^ε, ρ)-LCCs of rate $1 - \alpha$, for any positive constants α, ε and some positive constant ρ. Our decoding algorithm runs in time linear in the number of queries, and hence sublinear in the length of the message. We provide a general method for turning codes with smooth local reconstruction algorithms into LCCs: our main result, Theorem 6.13, states that as long as the inner code $\mathcal{C}_0$ has rate at least $1/2$ and possesses a smooth local reconstruction algorithm, then the corresponding family of expander codes are constant rate LCCs. In Section 6.4, we give some examples of appropriate inner codes, leading to the parameters claimed above.

In addition to providing a sublinear time local decoding algorithm for an important family of codes, our constructions are only the third known example of LDCs with rate approaching one, after multiplicity codes [79] and lifted Reed-Solomon codes [41]. Our approach (and the resulting codes) are very different from earlier approaches. Both multiplicity codes and lifted Reed-Solomon codes use the same basic principle, also at work in Reed-Muller codes: in these schemes, for any two codewords c_1 and c_2 which differ at index i, the corresponding queries $c_1|_{S(i)}$ and $c_2|_{S(i)}$ differ in many places. Thus, if the queries are smooth, with high probability they will not have too many errors, and the correct symbol can be recovered. In contrast, our decoder works differently: while our queries are smooth, they will not have this distance property. In fact, changing a mere $\log(Q)$ out of our Q queries may change the correct answer. The trick is that these problematic error patterns must have a lot of structure, and we will show that they are unlikely to occur.

Finally, our results port a typical argument from the low-query regime to the high-rate regime. As mentioned above, when the query complexity Q is constant, a smooth local reconstruction algorithm is sufficient for local correctability. However, this reasoning fails when Q grows with n. In this paper, we show how to make this argument go through: via Theorem 6.13, any family of codes $\mathcal{C}_0$ with good rate and a smooth local decoder can be used to obtain a family of LCCs with similar parameters.

6.1.4 Chapter organization

Before getting into our local correction algorithm, we state some basic results about expander graphs. In particular, we will need a slightly nonstandard Chernoff bound for expander graphs, which we will prove in Section 6.2. Next, in Section 6.3, we will give our local correction algorithm and prove that it works, provided that the inner code $\mathcal{C}_0$ satisfies a few locality conditions. At this point, the reader will likely be asking themselves if these inner codes exist, and if so, whether or not they produce interesting results. In Section 6.4, we will give two examples of inner codes, which will produce a locally correctable outer code with the advertised parameters.

6.2 Overview of expander graphs

In this section, we give a brief overview of expander graphs and codes arising from them. We saw in Chapter 2 how to make a code $\mathcal{C} \in \mathbb{F}_q^n$ out of an *inner code* $\mathcal{C}_0 \subset \mathbb{F}_q^d$ and a d-regular bipartite graph G on $2N$ vertices. Briefly, the block length n of $\mathcal{C}$ will be $|E(G)| = Nd$, and we will identify elements of $\mathbb{F}_q^n$ with labelings of the edges of G. A labeling is in $\mathcal{C}$ if at every vertex of G, the edges leaving that vertex (in some prescribed order) form a codeword in $\mathcal{C}_0$.

In this chapter, we will consider the case when the underlying graph arises from an *expander graph.* A complete exposition of expander graphs is beyond the scope of this thesis: the reader is referred to [69] for an excellent survey. In the meantime, we will briefly recap the basic notions that we will need. Let $G = (V, E)$ be a d-regular graph on N vertices. (Not necessarily bipartite). Let A be the *normalized adjacency matrix* of G; that is, $A \in \{0, 1/d\}^{N\times N}$ and

$$A_{ij} = \begin{cases} \frac{1}{d} & (i,j) \in E \\ 0 & (i,j) \notin E \end{cases}$$

Consider the spectrum of A. It is not hard to see that the largest eigenvalue of A is 1, and that the corresponding eigenvector is the all-ones vector $\mathbf{1} \in \mathbb{R}^N$. If G is connected, it turns out that the second-largest eigenvalue is strictly less than 1.

Definition 6.4. *Let G be a connected d-regular graph with normalized adjacency matrix A. The second-largest eigenvalue of A is called the* expansion parameter *of G, and is denoted $\lambda = \lambda(G)$.*

We will see a few reasons for the name "expansion parameter" later; it turns out that the smaller λ is, the more "connected" G is. If λ is smallish, we say that G is an *expander graph.* If λ is basically as small as it can be, we say that G is a *Ramanujan graph:*

Definition 6.5. *A d-regular graph $G = (V, E)$ is a* Ramanujan graph *if $\lambda(G) \leq \frac{2\sqrt{d-1}}{d}$.*

It is known that this is basically the smallest $\lambda(G)$ can be; more precisely, $\lambda(G) \geq \frac{2\sqrt{d-1}}{d} - o(1)$. Not surprisingly (given what we've seen so far in this thesis), a random d-regular graph is Ramanujan with high probability. Much more surprisingly, there exist explicit constructions of Ramanujan graphs [83,85,86] for arbitrarily large values of d. We will use the existence (and explicitness) of these constructions as a black box.

To get a suitable bipartite graph H out of G, we will take the *double cover* of G.

Definition 6.6. *Let G be any graph on N vertices. The double cover H of G is a bipartite graph on $2N$ vertices, as follows. The vertices $V(H)$ of H are two disjoint copies V_0 and V_1 of $V(G)$. For each edge $(u, v) \in E(G)$, there are two edges (u_0, v_1) and (v_0, u_1) in $E(H)$, where u_i is the copy of u in V_i.*

The notation for double covers is illustrated in Figure 6.1.

We return to the expansion parameter λ. What does λ tell us about a graph G, or its double-cover H? Generally, as it turns out, the smaller λ is, the more like the complete graph (or the complete bipartite graph) G (or H) behaves. More

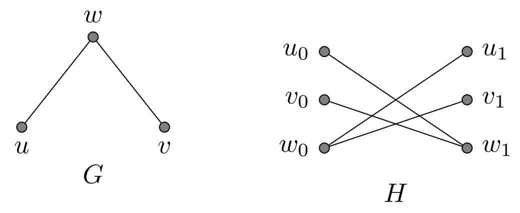


Figure 6.1: A graph G and its double-cover H.

specifically, suppose that a subset of B of vertices are "bad," and consider a random walk on G. Let X be the number of bad vertices that this walk hits. If G is a complete graph, then each step of the random walk is an independent, uniformly random vertex, and the number of bad vertices is controlled by a Chernoff bound (Theorem 2.15). We would like to mimic this behavior when G is degree d, rather than $N-1$. The well-known *expander Chernoff bound* [35, 70] says that we may do this, and the quality of the result depends on the expansion parameter λ. In this chapter, we'll need a slight variant on the expander Chernoff bound, which we state and prove below.

Lemma 6.7. *Let G be a d-regular graph on N vertices, and H be its double cover. Let $B \subset E(H)$ best a set of $\rho|E(H)|$ edges, and suppose that $\rho > 6\lambda$, where $\lambda = \lambda(G)$ is the expansion parameter. Let $v_0, \ldots, v_L$ be a random walk of length L on H, starting from the left side at a vertex chosen from a distribution*[1] *ν with $\left\|\nu - \frac{1}{n}\mathbf{1}_n\right\|_2 \le \frac{1}{\sqrt{n}}$. Let X denote the number of edges in B included in the walk, and choose γ so that $\rho + 2\lambda < \gamma < 1/2$. Then*

$$\mathbb{P}\left\{X \ge \gamma L\right\} \le \exp\left(-L\, D\left(\gamma || \rho + 2\lambda\right)\right).$$

In particular, when $\rho + 2\lambda \le \ln(1/(1-\gamma))$, we have

$$\mathbb{P}\left\{X \ge \gamma L\right\} \le \left(\frac{\rho + 2\lambda}{\gamma}\right)^{\gamma L}.$$

[1]We think of a distribution ν on V_0 as a vector $\nu \in \mathbb{R}^N_{\ge 0}$ so that $\|\nu\|_1 = 1$. Thus, $\nu[u]$ is the probability mass on vertex u.

As mentioned above, this is very much like the expander Chernoff bound [35, 70]. In this case, H is the double cover of an expander, not an expander itself, and the edges, rather than vertices, are corrupted, but the proof remains basically the same. For completeness, we include the proof of Lemma 6.7 here.

6.2.1 Proof of Lemma 6.7

The lemma follows with only a few tweaks from standard results. The only differences between this and a standard analysis of random walks on expander graphs are that (a) we are walking on the edges of the bipartite graph H, rather than on the vertices of G, and (b) our starting distribution is not uniform but instead close to uniform. Dealing with this differences is straightforward, but we document it below for completeness.

First, we need the relationship between a walk on the edges of a bipartite graph H and the corresponding walk on the vertices of G. For ease of analysis, we will treat H as directed, with one copy of each edge in each direction.

Lemma 6.8. *Let G be a degree d undirected graph on d vertices with normalized adjacency matrix A, and let H be the double cover of G. For each vertex v of G, label the edges incident to v arbitrarily, and let $v(i)$ denote the i^{th} edge of v. Let H' be the graph with vertices $V(G) \times [d] \times \{0,1\}$ and edges*

$$E(H') = \{((u,i,b),(v,j,b')) \,:\, (u,v) \in E(G), b \neq b', u(i) = v\}\,.$$

Then H' is a directed graph with $2dN$ edges, and in-degree and out-degree both equal to d. Further, the normalized adjacency matrix A' is given by

$$A' = R \otimes S$$

where $S : \mathbb{R}^2 \to \mathbb{R}^2$ is $S = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and $R : \mathbb{R}^{nd} \to \mathbb{R}^{nd}$ is an operator with the same *rank and spectrum as A.*

Proof. We will write down A' in terms of A. Index $[N]$ by vertices of V, so that $e_v \in \mathbb{R}^n$ refers to the standard basis vector with support on v. Let $\otimes$ denote the Kronecker product. We will need some linear operators. Let $B : \mathbb{R}^{N^2} \to \mathbb{R}^{N^2}$ so that

$$B(e_u \otimes e_v) = e_v \otimes e_v$$

and $P : \mathbb{R}^{N^2} \to \mathbb{R}^{Nd}$ so that

$$P(e_u \otimes e_v) = \begin{cases} e_u \otimes e_i & v = u(i) \\ 0 & (u,v) \notin E(G) \end{cases}.$$

Finally, let $S : \mathbb{R}^2 \to \mathbb{R}^2$ be the cyclic shift operator. Then a computation shows that the adjacency matrix A' of H' is given by

$$(P(I \otimes A)BP^T) \otimes S.$$

Let $R = P(I \otimes A)BP^T$. To see that the rank of R is at most N, note that for any $i \in [d]$ and any $u \in V(G)$,

$$R(e_u \otimes e_j) = e_{u(j)} \otimes \frac{1}{d}\mathbf{1}_d.$$

In particular, it does not depend on the choice of j. Since $\{e_u \otimes e_j \,:\, u \in V(G), j \in [d]\}$ is a basis for $\mathbb{R}^{Nd}$, the image of R has dimension at most n. Finally, a similar computation shows that if p is an eigenvector of A with eigenvalue λ, then $p \otimes \frac{1}{d}\mathbf{1}_d$ is a right eigenvector of R, also with eigenvalue λ. (The left eigenvectors are $P(\frac{1}{N}\mathbf{1}_N \otimes p)$). This proves the claim. □

With a characterization of A' in hand, we now wish to apply an expander Chernoff bound. Existing bounds require slight modification for this case (since the graph H'

is directed and also not itself an expander), so for completeness we sketch the changes required. The proof below follows the strategies in [2] and [70]. We begin with the following lemma, following from the analysis of [2].

Lemma 6.9. *Let G and H be as in Lemma 6.8, and let $v_0, v_1, \ldots, v_T$ be a random walk on the vertices of H, beginning at a vertex of H, chosen as follows: the side of H is chosen according to a distribution $\sigma_0 = (s, 1-s)$, and the vertex within that side is chosen independently according to a distribution ν with $\|\nu - \frac{1}{N}\mathbf{1}_N\|_2 \leq \frac{1}{\sqrt{N}}$. Let W be any set of edges in H, with $|W| \leq \rho n d$. Suppose that $\rho > 6\lambda$. Then for any set $S \subset \{0, 1, \ldots, T-1\}$,*

$$\mathbb{P}\left\{(v_t, v_{t+1}) \in W, \forall t \in S\right\} \leq (\rho + 2\lambda)^{|S|}.$$

Proof. As in Lemma 6.8, we will consider H as directed, with one edge in each direction. As before, we will index these edges by triples $(u, i, \ell) \in V(G) \times [d] \times \{0, 1\}$, so that (u, i, ℓ) refers to the i^{th} edge leaving vertex u on the ℓ^{th} side of H. Let μ be the distribution on the first step (v_0, v_1) of the walk, so

$$\mu = \nu \otimes \frac{1}{d}\mathbf{1}_d \otimes \sigma_0.$$

Let $M \in \mathbb{R}^{2Nd}$ be the projector onto the edges in W. Let $M^{(0)}$ be the restriction to edges emanating from the left side of H, and $M^{(1)}$ from the right side, so that both $M^{(0)}$ and $M^{(1)}$ are $Nd \times Nd$ binary diagonal matrices with at most $\rho N d$ nonzero entries. Let $A' = R \otimes S$ be as in the conclusion of Lemma 6.8. After running the random walk for T steps, consider the distribution on directed edges of H, conditional on the bad event that $(v_t, v_{t+1}) \in W$ for all $t \in S$. As in the analysis in [2], this distribution is given by

$$\mu_T = \frac{(M_{T_1}A')(M_{T-2}A') \cdots (M_1 A')(M_0 \mu)}{\mathbb{P}\left\{(v_t, v_{t+1}) \in W, \forall t \in S\right\}},$$

where

$$M_t = \begin{cases} M & t \in S \\ I & t \notin S \end{cases}.$$

Since the ℓ_1 norm of any distribution is 1, we have

$$\mathbb{P}\left\{(v_t, v_{t+1}) \in W, \forall t \in S\right\} = \left\|(M_{T-1}A')(M_{T-2}A') \cdots (M_1 A')(M_0 \mu)\right\|_1 \tag{6.1}$$

Let

$$\mu_0 := M_0 \mu,$$

and

$$\mu_t := M_t A' \mu_{t-1},$$

so we seek an estimate on $\left\|\mu_T\right\|_1$.

The following claim will be sufficient to prove the theorem.

Claim 6.10. *If $\rho \geq 6\lambda$, and $t \in S$,*

$$(\mu - 2\lambda)\left\|\mu_t\right\|_1 \leq \left\|\mu_{t+1}\right\|_1 \leq (\mu + 2\lambda)\left\|\mu_t\right\|_1.$$

On the other hand, if $t \notin S$,

$$\left\|\mu_t\right\|_1 = \left\|\mu_{t+1}\right\|_1.$$

The second half of the claim follows immediately from the definition of μ_t. To prove the first half, suppose that $t \in S$. We will proceed by induction. Again, we follow the analysis of [2].

Write $\mu_0 = v_0 \otimes \sigma_0$, and write $\sigma_0 = (s, 1-s)$ Part of our inductive hypothesis will be that for all t,

$$\mu_t = v_t^{(0)} \otimes s_t e_0 + v_t^{(1)} \otimes (1 - s_t) e_1,$$

where $s_t = s$ if t is even and $1-s$ if t is odd, and where $v_t^{(i)} \in \mathbb{R}^{Nd}$. For $i \in \{0,1\}$, write

$$v_t^{(i)} = x_t^{(i)} + y_t^{(i)},$$

where $x_t^{(i)} \| \mathbf{1}$ and $y_t^{(i)} \perp \mathbf{1}$. The second part of the inductive hypothesis will be

$$\|y_t^{(i)}\|_2 \le q\|x_t^{(i)}\|_2, \tag{6.2}$$

for a parameter q to be chosen later, and for $i \in \{0,1\}$.

Because

$$\begin{aligned}\|\mu_t\|_1 &= s_t\|v_t^{(0)}\|_1 + (1-s_t)\|v_t^{(1)}\|_1 \\ &= s_t\|x_t^{(0)}\|_1 + (1-s_t)\|x_t^{(1)}\|_1 \\ &= \sqrt{nd}\left(s_t\|x_t^{(0)}\|_2 + (1-s_t)\|x_t^{(1)}\|_2\right),\end{aligned}$$

it suffices to show that

$$(\mu - 2\lambda)\left\|x_t^{(0)}\right\|_2 \le \left\|x_{t+1}^{(1)}\right\|_2 \le (\mu + 2\lambda)\left\|x_t^{(0)}\right\|_2 \tag{6.3}$$

and similarly with the 0 and 1 switched. The analysis is the same for the two cases, so we just establish (6.3). Using the decomposition $A' = R \otimes S$ from Lemma 6.8,

$$\begin{aligned}\mu_{t+1} &= M_t(R\otimes S)(v_t^{(0)} \otimes s_t e_0 + v_t^{(1)} \otimes (1-s_t)e_1) \\ &= M_t\left(Rv_t^{(0)} \otimes (1-s_{t+1})e_1 + Rv_t^{(1)} \otimes s_{t+1}e_0\right) \\ &= \left(M_t^{(1)}Rv_t^{(0)}\right) \otimes (1-s_{t+1})e_1 + \left(M_t^{(0)}Rv_t^{(1)}\right) \otimes s_{t+1}e_0\end{aligned}$$

This establishes the first inductive claim about the structure of μ_{t+1}, and

$$v_{t+1}^{(0)} = M_t^{(0)}Rv_t^{(1)} \qquad \text{and} \qquad v_{t+1}^{(1)} = M_t^{(1)}Rv_t^{(0)}.$$

Consider just $v_{t+1}^{(1)}$. We have

$$v_{t+1}^{(1)} = M_t^{(1)}R(x_t^{(0)} + y_t^{(0)}).$$

Because $t \in S$, we know that $M_t^{(1)}$ is diagonal with at most $\rho n d$ nonzeros, and further we know that R has second normalized eigenvalue at most λ, by Lemma 6.8. The analysis in [2] now shows that, using the inductive hypothesis (6.2),

$$(6.4) \quad \rho\|x_t^{(0)}\|_2 - q\lambda\sqrt{\rho(1-\rho)}\|x_t^{(0)}\|_2 \le \|x_{t+1}^{(1)}\|_2 \le \rho\|x_t^{(0)}\|_2 + q\lambda\sqrt{\rho(1-\rho)}\|x_t^{(0)}\|_2,$$

and that

$$\|y_{t+1}^{(1)}\|_2 \le q\lambda\|x_t^{(0)}\|_2 + \sqrt{\rho(1-\rho)}\|x_t^{(0)}\|_2.$$

We must ensure that (6.2) is satisfied for the next round. As long as $\lambda < \rho/6$, this follows from the above when

$$q = 2\sqrt{\frac{1-\rho}{\rho}}.$$

With this choice of q, the (6.3) follows from (6.4). Further, the hypotheses on ν show that the (6.2) is satisfied in the initial step. $\square$

Finally, we invoke the following theorem, from [70].

Theorem 6.11 (Theorem 3.1 in [70]). *Let $X_1, \ldots, X_L$ be binary random variables so that for all $S \subset [L]$,*

$$\mathbb{P}\left\{\bigwedge_{i\in S} X_i = 1\right\} \le \delta^{|S|}.$$

Then for all $\gamma > \delta$,

$$\mathbb{P}\left\{\sum_{i=1}^{L} X_i \ge \gamma L\right\} \le e^{-LD(\gamma\|\delta)}.$$

Lemma 6.7 follows immediately.

6.3 Local correctability of expander codes

Preliminaries dispensed with, we are ready to present our local correction algorithm for expander codes. We use a formulation of expander codes due to [115]. Let G be a d-regular expander graph on N vertices with expansion parameter λ, as in

Definition 6.4. We will take G to be a Ramanujan graph, that is, so that $\lambda \leq \frac{2\sqrt{d-1}}{d}$; as mentioned above, explicit constructions of Ramanujan graphs are known [83,85,86] for arbitrarily large values of d. Let H be the double cover of G, as in Definition 6.6. Fix a linear inner code $\mathcal{C}_0$ over $\mathbb{F}$ of rate R_0 and relative distance δ_0. Let $N = nd$. For $v_i \in V(H)$, let $\Gamma(v_i) = (\Gamma_1(v_i), \ldots, \Gamma_d(v_i))$ denote the edges attached to v, with an arbitrary order. The expander code $\mathcal{C} \subset \mathbb{F}^n$ of length n arising from G and $\mathcal{C}_0$ is the Tanner code (as in Definition 2.2) defined by H and $\mathcal{C}_0$. That is,

$$\mathcal{C} = \mathcal{C}_n(\mathcal{C}_0, G) = \left\{ x \in \mathbb{F}^n \ : \ x|_{\Gamma(v_i)} \in \mathcal{C}_0 \text{ for all } v_i \in V(H) \right\} \tag{6.5}$$

As we saw in Chapter 2, as long as the inner code $\mathcal{C}_0$ has good rate and distance, so does the resulting code $\mathcal{C}$.

Theorem 6.12 ([98,105]). *The code $\mathcal{C}$ has rate $R \geq 2R_0 - 1$, and as long as $2\lambda \leq \delta_0$, the relative distance of $\mathcal{C}$ is at least $\delta_0^2/2$.*

Notice that when $R_0 < \frac{1}{2}$, Theorem 6.12 is meaningless. The rate in Theorem 6.12 comes from the fact that $\mathcal{C}_0$ has rate R_0, so each vertex induces $(1 - R_0)d$ linear constraints, and there are N vertices, so the outer code has $Nd(1 - R_0)$ constraints. Since the outer code has length $n = Nd/2$, its rate is at least $2R_0 - 1$. This naïve lower bound on the rate ignores the possibility that the constraints induced by the different vertices may not all be independent. It is an interesting question whether for certain inner codes, a more careful counting of constraints could yield a better lower bound on the rate. The ability to use inner codes of rate less than $\frac{1}{2}$ would permit much more flexibility in the choice of inner code in our constructions.

The difficulty of a more sophisticated lower bound on the rate was noticed by Tanner, who pointed out that simply permuting the codewords associated with a given vertex could drastically alter the parameters of the outer code [105].

6.3.1 Local Correction

If the inner code $\mathcal{C}_0$ has a smooth local reconstruction procedure, then not only does $\mathcal{C}$ have good distance, but we show it can also be efficiently locally corrected. Our main result is the following theorem.

Theorem 6.13. *Let $\mathcal{C}_0$ be a linear code over $\mathbb{F}$ of length d and rate $R_0 > 1/2$. Suppose that $\mathcal{C}_0$ has a s_0-smooth local reconstruction procedure with query complexity Q_0. Let $\mathcal{C} = \mathcal{C}_n(\mathcal{C}_0, G)$ be the expander code of length n arising from the inner code $\mathcal{C}_0$ and a Ramanujan graph G. Choose any $\gamma < 1/2$ and any $\zeta > 0$ satisfying $\gamma\left(e^{\zeta}Q_0\right)^{-1/\gamma} > 8\lambda$. Then $\mathcal{C}$ is (Q, ρ)-locally correctable, for any error rate ρ, with $\rho < \gamma\left(e^{\zeta}Q_0\right)^{-1/\gamma} - 2\lambda$. The success probability is*

$$1 - \left(\frac{n}{d}\right)^{-1/\ln(d/4)}$$

and the query complexity is

$$Q = \left(\frac{n}{d}\right)^{\varepsilon} \qquad \text{where} \qquad \varepsilon = \left(1 + \frac{\ln(Q_0') + 1}{\zeta}\right) \cdot \frac{\ln(Q_0')}{\ln(d/4)}.$$

Further, when the length of the inner code, d, is constant, the correction algorithm runs in time $O(|\mathbb{F}|^{Q_0'+1}Q)$, where $Q_0' = Q_0 + (d - s_0)$.

Remark 12. *We will choose d (and hence $Q_0' < d$) and $|\mathbb{F}|$ to be constant. Thus, the rate of $\mathcal{C}$, as well as the parameters ρ and ε, will be constants independent of the block length n. The parameter ζ trades off between the query complexity and the allowable error rate. When Q_0 is much smaller than d (for example, $Q_0 = 3$ and d is reasonably large), we will want to take $\zeta = O(1)$. On the other hand, if $Q_0 = d^{\varepsilon}$ and d is chosen to be a sufficiently large constant, we should take ζ on the order of $\ln(Q_0)$.*

Before diving into the details, we outline the correction algorithm. First, we observe that it suffices to consider the case when the local correction algorithm S_0 of the inner code is perfectly smooth: that is, the queries of the inner code are uniformly random. Otherwise, if S_0 is s_0-smooth with Q_0 queries, we may modify it so that it is d-smooth with $Q_0 + (d - s_0)$ queries, by having it query extra points and then ignore them. Thus, we set $Q_0' = Q_0$ and assume in the following that S_0 makes Q_0 perfectly smooth queries.

Suppose that $\mathcal{C}_0$ has local reconstruction algorithm (S_0, A_0), and we receive a corrupted codeword, w, which differs from a correct codeword c^* in at most a ρ fraction of the entries. Say we wish to determine $c^*[(u_0, v_1)]$, for $(u_0, v_1) \in E(H)$. The algorithm proceeds in two steps. The first step is to find a set of about $n^{\varepsilon/2}$ query positions which are nearly uniform in $[n]$, and whose correct values together determine $c^*[(u_0, v_1)]$. The second step is to correct each of *these* queries with very high probability—for each, we will make another $n^{\varepsilon/2}$ or so queries.

Step 1. By construction, $c^*[(u_0, v_1)]$ is a symbol in a codeword of the inner code, $\mathcal{C}_0$, which lies on the edges emanating from u_0. By applying S_0, we may choose Q_0 of these edges, $S = S_0(u_0) = \left\{ (u_0, s_1^{(i)}) \, : \, i \in [Q_0] \right\}$, so that

$$A_0 \left(c^*|_S \, , (u_0, v_1) \right) = c[(u_0, v_1)].$$

Now we repeat on each of these edges: each $(u_0, s_1^{(i)})$ is part of a codeword emanating from $s_1^{(i)}$, and so Q_0 more queries determine each of those, and so on. Repeating this L_1 times yields a Q_0-ary tree T of depth L_1, whose nodes are labeled by of edges of H. This tree-making procedure is given more precisely below in Algorithm 4. Because the queries are smooth, each path down this tree is a random walk in H; because G is an expander, this means that the leaves themselves, while not independent, are each close to uniform on $E(H)$. Note that at this point, we have not made any queries, merely documented a tree, T, of edges we could query.

Step 2. Our next step is to actually make queries to determine the correct values on the edges represented in the leaves of T. By construction, these values determine $c^*[(u_0, v_1)]$. Unfortunately, in expectation a ρ fraction of the leaves are corrupted, and without further constraints on $\mathcal{C}_0$, even one corrupted leaf is enough to give the wrong answer. To make sure that we get all of the leaves correct, we use the fact that each leaf corresponds to a position in the codeword that is nearly uniform (and in particular nearly independent of the location we are trying to reconstruct). For each edge, e, of H that shows up on a leaf of T, we repeat the tree-making process beginning at this edge, resulting in new Q_0-ary trees T_e of depth L_2. This time, we make all the queries along the way, resulting in an evaluated tree τ_e, whose nodes are labeled by elements of $\mathbb{F}$; the root of τ_e is the e-th position in the corrupted codeword, $w[e]$, and we hope to correct it to $c^*[e]$.

For a fixed edge, e, on a leaf of T, we will correct the root of $\tau = \tau_e$ with very high probability, large enough to tolerate a union bound over all the trees τ_e. For two labelings σ and ν of the same tree by elements of $\mathbb{F}$, we define the distance

$$D(\sigma, \nu) = \max_P \delta\left(\sigma|_P, \nu|_P\right), \tag{6.6}$$

where the maximum is over all paths P from the root to a leaf, and $\sigma|_P$ denotes the restriction of σ to P. We will show below in Section 6.3.2 that it is very unlikely that τ contains a path from the root to a leaf with more than a constant fraction $\gamma < 1/2$ of errors. Thus, in the favorable case, the distance between the correct tree τ^* arising from c^* and the observed tree τ is at most $D(\tau^*, \tau) \leq \gamma$. In contrast, we will show that if σ^* and τ^* are both trees arising from legitimate codewords with distinct roots, then σ^* and τ^* must differ on an entire path P, and so $D(\sigma^*, \tau) > 1 - \gamma$. To take advantage of this, we show in Algorithm 5 how to efficiently compute

$$\mathsf{Score}(a) = \min_{\sigma^* : \mathrm{root}(\sigma^*) = a} D(\sigma^*, \tau)$$

for all a, where $\mathrm{root}(\sigma^*)$ denotes the label on the root of σ^*. The above argument (made precise below in Section 6.3.2) shows that there will be a unique $a \in \mathbb{F}$ with score less than γ, and this will be the correct symbol $c^*[e]$.

Finally, with all of the leaves of T correctly evaluated, we may use A_0 to work our way back up T and determine the correct symbol corresponding to the edge at the root of T. The complete correction algorithm is given below in Algorithm 3.

Algorithm 3: correct: Local correcting protocol.

```
Input: An index $e_0 \in E(H)$, and a corrupted codeword $w \in \mathbb{F}^{E(H)}$.
Output: With high probability, the correct value of the $e_0$'th symbol.
Set $L_1 = \log(N)/\log(d/4)$ and fix a parameter $L_2$.
$T = \mathsf{makeTree}(e_0, L_1)$
for each edge $e$ of $H$ that showed up on a leaf of $T$ do
    $T_e = \mathsf{makeTree}(e, L_2)$.
    Let $\tau_e = T_e|_w$ be the tree of symbols from $w$.
    $w^*[e] = \mathsf{correctSubtree}(\tau_e)$.
Initialize a $Q_0$-ary tree $\tau^*$ of depth $L_1$.
Label the leaves of $\tau^*$ according to $T$ and $w^*$: if a leaf of $T$ is labeled $e$, label the
corresponding leaf of $\tau^*$ with $w^*[e]$.
Use the local reconstruction algorithm $A_0$ of $\mathcal{C}_0$ to label all the nodes in $\tau^*$.
return the label on the root of $\tau^*$.
```

The number of queries made by Algorithm 3 is

$$Q = Q_0^{L_1 + L_2} \tag{6.7}$$

and the running time is $O(t_d |\mathbb{F}|^{Q_0+1} Q)$, where t_d is the time required to run the local correction algorithm of $\mathcal{C}_0$. For us, both d and $|\mathbb{F}|$ will be constant, and so the running time is $O(Q)$.

Algorithm 4: makeTree: Uses the local correction property of $\mathcal{C}_0$ to construct a tree of indices.

```
Input: An initial edge $e_0 = (u_0, v_1) \in E(H)$, and a depth $L$.
Output: A $Q_0$-ary tree $T$ of depth $L$, whose nodes are indexed by edges of $H$, with root $e_0$
Initialize a tree $T$ with a single node labeled $e_0$
$s = 0$
for $\ell \in [L]$ do
    Let leaves be the current leaves of $T$.
    for $e = (u_s, v_{1-s}) \in$ leaves do
        Let $\left\{ v_{1-s}^{(i)} : i \in [d] \right\}$ be the neighbors of $u_s$ in $H$.
        Choose queries $Q_0(e) \subset \left\{ (u_s, v_{1-s}^{(i)}) : i \in [d] \right\}$, and add each query in $T$ as a child at $e$.
    $s = 1 - s$
return $T$
```

Algorithm 5: correctSubtree: Correct the root of a fully evaluated tree τ.

```
Input: $\tau$, a $Q_0$-ary tree of depth $L$ whose nodes are labeled with elements of $\mathbb{F}$.
Output: A guess at the root of the correct tree $\tau$.
For a node $x$ of $\tau$, let $\tau[x]$ denote the label on $x$.
for leaves $x$ of $\tau$ and $a \in \mathbb{F}$ do
    $\mathsf{best}_a(x) = \begin{cases} 1 & \tau[x] \neq a \\ 0 & \tau[x] = a \end{cases}$
for $\ell = L-1, L-2, \ldots, 0$ do
    for nodes $x$ at level $\ell$ in $\tau$ and $a \in \mathbb{F}$ do
        Let $y_1, \ldots, y_{Q_0}$ be the children of $x$.
        Let $S_a \subset \mathbb{F}^{Q_0}$ be the set of query responses for the children of $x$ so that $A_0$ returns $a$ on those responses.
        $\mathsf{best}_a(x) = \min_{(a_0, \ldots, a_{Q_0}) \in S_a} \max_{r \in [Q_0]} \left( \mathsf{best}_{a_r}(y_r) + \mathbf{1}_{\tau(y_r) \neq a_r} \right)$
Let $r$ be the root of $\tau$.
for $a \in \mathbb{F}$ do
    $\mathsf{Score}(a) = \dfrac{\mathsf{best}_a(r) + \mathbf{1}_{\tau(r) \neq a}}{L}$
return $a \in \mathbb{F}$ with the smallest $\mathsf{Score}(a)$.
```

6.3.2 Proof of Theorem 6.13

Suppose that $c^* \in \mathcal{C}$, and Algorithm 3 is run on a received word w with $\delta(c^*, w) \leq \rho$. To prove Theorem 6.13, we must show that Algorithm 3 returns $c^*[e_0]$ with high probability. As remarked above, we assume that the inner recovery algorithm S_0 is perfectly smooth.

We follow the proof outline sketched in Section 6.3.1, which rests on the following observation.

Proposition 6.14. *Let $c_1, c_2 \in \mathcal{C}$ and let $e \in E(H)$ so that $c_1[e] \neq c_2[e]$. Let the distance D between trees with labels in $\mathbb{F}$ be as in (6.6). Let $T = \mathsf{makeTree}(e)$, and let $\tau = T|_{c_1}$ and $\sigma = T|_{c_2}$ be the labeled trees corresponding to c_1 and c_2 respectively. Then $D(\tau, \sigma) = 1$. That is, there is some path from the root to the leaf of T so that τ and σ disagree on the entire path.*

Proof. Since $c_1[e] \neq c_2[e]$, τ and σ have different symbols at their root. Since the labels on the children of any node determine the label on the node itself (via the local correction algorithm), it must be that τ and σ differ on some child of the root. Repeating the argument proves the claim. □

Let τ_e be the tree arising from the received word w, starting at e, as in Algorithm 3. Let

$$\mathcal{T}_e = \{\mathsf{makeTree}(e)|_c \,:\, c \in \mathcal{C}\}$$

be the set of query trees arising from uncorrupted codewords, and let $\tau_e^* \in \mathcal{T}_e$ be the "correct" tree, corresponding to the original uncorrupted codeword c^*. Suppose that

$$D(\tau_e, \tau_e^*) \leq \gamma \tag{6.8}$$

for some $\gamma \in [0, 1/2)$. Then Proposition 6.14 implies that for any $\sigma_e^* \in \mathcal{T}_e$ with a different root from τ_e^* has

$$D(\tau_e, \sigma_e^*) \geq 1 - \gamma. \tag{6.9}$$

Indeed, there is some path along which τ_e^* and σ_e^* differ in every place, and along this path, τ_e agrees with τ_e^* in at least a $1 - \gamma$ fraction of the places. Thus, τ_e disagrees with σ_e^* in those same places, establishing (6.9). Consider the quantity

$$\mathsf{Score}(a) = \min_{\sigma_e^* \in \mathcal{T}_e : \mathrm{root}(\sigma_e^*) = a} D(\tau_e, \sigma_e^*). \tag{6.10}$$

Equations (6.8) and (6.9) imply that if a^* is the label on the root of τ_e^*, then $\mathsf{Score}(a) \leq \gamma$, and otherwise, $\mathsf{Score}(a) \geq 1 - \gamma$. Thus, to establish the correctness of Algorithm 3, it suffices to argue first that Algorithm 5 correctly computes $\mathsf{Score}(a)$ for each a, and second that (6.8) holds for all trees τ_e in Algorithm 3.

The first claim follows by inspection. Indeed, for a node $x \in \tau_e$, let $(\tau_e)_x$ denote the subtree below x. Let $\mathcal{T}_e^{(x,a)}$ denote the set of trees in $\mathcal{T}_e$ so that the node x is labeled a. Throughout Algorithm 3, the quantity $\mathsf{best}_a(x)$ gives the distance from the observed tree rooted at x to the best tree in $\mathcal{T}_e$, rooted at x, with the additional restriction that the label at x should be a. That is,

$$\mathsf{best}_a(x) = \min_{\sigma_e^* \in \mathcal{T}_e^{(x,a)}} \tilde{D}\left((\sigma_e^*)_x, (\tau_e)_x\right), \tag{6.11}$$

where $\tilde{D}$ is the same as D except it does not count the root, and it is not normalized. It is easy to see that (6.11) is satisfied for leaves x of τ_e. Then for each node, Algorithm 5 updates $\mathsf{best}_a(x)$ by considering the best labeling on the children of x consistent with $\tau(x) = a$, taking the distance of the worst of those children, and adding one if necessary.

To establish the second claim, that (6.8) holds for all trees τ_e, we will use Lemma 6.7 from Section 6.2. Applying Lemma 6.7 with B equal to the set of corrupted edges, we see that a random walk on H will not hit too many corrupted edges. The conditions on ρ and λ in the statement of Theorem 6.13 implies that $\rho > 6\lambda$, and so Lemma 6.7 applies to random walks on H.

Suppose that L_1 is even, and consider any leaf of T. This leaf has label $(u_0, v_1) \in E(H)$, where u is the result of a random walk of length L_1 on G and v is a randomly chosen neighbor of u. Because G is a Ramanujan graph, the distribution μ on u

satisfies

$$\left\| \mu - \frac{1}{N}\mathbf{1}_N \right\|_2 \leq \lambda^{L_1} \leq \frac{1}{\sqrt{N}}$$

as long as

$$L_1 \geq \frac{\log(N)}{\log(d/4)}.$$

Thus, Lemma 6.7 applies to random walks in H starting at e. Fix a leaf of τ_e; by the smoothness of the query algorithm S_0, each path from the root to the leaf of each tree τ_e is a uniform random walk, and so with high probability, the number of corrupted edges on this walk is not more than γL_2, which was the desired outcome.

Finally, we union bound over $Q_0^{L_1}$ trees τ_e and $Q_0^{L_2}$ paths in each tree. We will set $L_2 = CL_1$, for a constant C to be determined. Thus, (6.8) holds (and hence Algorithm 3 is correct) except with probability at most

$$\mathbb{P}\left\{\text{Algorithm 3 fails}\right\} \leq \exp\left((C+1)L_1 \ln(Q_0) - C\gamma L_1 \ln\left(\frac{\gamma}{\rho + 2\lambda}\right)\right). \tag{6.12}$$

Our goal is to show that $\mathbb{P}\left\{\text{Algorithm 3 fails}\right\} \leq \exp(-L_1)$, which is equivalent to showing

$$(C+1)\ln(Q_0) - C\gamma \ln\left(\frac{\gamma}{\rho+2\lambda}\right) < -1$$

Rearranging, this means our goal is to find C so that

$$C\left(\ln(Q_0) - \gamma \ln\left(\frac{\gamma}{\rho+2\lambda}\right)\right) < -1 - \ln(Q_0)$$

By hypothesis in Theorem 6.13 we have $\rho < \gamma \left(e^{\zeta}Q_0\right)^{-1/\gamma} - 2\lambda$, which means that

$$\gamma \ln\left(\frac{\gamma}{\rho+2\lambda}\right) > \gamma \ln\left(\frac{\gamma}{\gamma\left(e^{\zeta}Q_0\right)^{-1/\gamma}}\right) = \gamma \ln\left(\left(e^{\zeta}Q_0\right)^{1/\gamma}\right) = \zeta + \ln Q_0$$

Thus

$$\ln(Q_0) - \gamma \ln\left(\frac{\gamma}{\rho+2\lambda}\right) < -\zeta$$

Thus choosing $C = \frac{\ln(Q_0)+1}{\zeta}$ is sufficient to bound the failure probability by $\exp(-L_1)$. From (6.7), $Q = Q_0^{(C+1)L_1}$, which completes the proof of Theorem 6.13.

6.4 Examples

In this section, we provide two examples of choices for $\mathcal{C}_0$, both of which result in (n^ε, ρ)-LCCs of rate $1 - \alpha$ for any constants $\varepsilon, \alpha > 0$ and for some constant $\rho > 0$. Our first and main example is a generalization of Reed-Muller codes, based on finite geometries. With these codes as $\mathcal{C}_0$, we provide LCCs over $\mathbb{F}_p$—unlike multiplicity codes, these codes work naturally over small fields.

Our second example comes from the observation that if the $\mathcal{C}_0$ is itself an LCC (of a fixed length) our construction provides a new family of (n^ε, ρ)-LCCs. In particular, plugging the multiplicity codes of [79] into our construction yields a novel family of LCCs. This new family of LCCs has a very different structure than the underlying multiplicity codes, but achieves roughly the same rate and locality.

Codes from Affine Geometries. One advantage of our construction is that the inner code $\mathcal{C}_0$ need not actually be a good locally decodable or correctable code. Rather, we only need a smooth reconstruction procedure, which is easier to come by. One example comes from affine geometries; in this example, we will show how use Theorem 6.13 to make LCCs of length n, rate $1 - \alpha$ and query complexity n^ε, for any $\alpha, \varepsilon > 0$.

For a prime power $h = p^\ell$ and parameters r and m, consider the r-dimensional affine subspaces $L_1, \ldots, L_t$ of the vector space $\mathbb{F}_h^m$. let H be the $t \times h^m$ incidence matrix of the L_i and the points of $\mathbb{F}_h^m$, and let $\mathcal{A}^*(r, m, h)$ be the code over $\mathbb{F}_p$ whose parity check matrix is H. These codes, examples of *finite geometry codes,* are well-studied, and their ranks can be exactly computed—see [3, 4] for an overview.

The definition of of $\mathcal{A}^*(r, m, h)$ gives a reconstruction procedure: we may query all the points in a random r-dimensional affine subspace of $\mathbb{F}_h^m$ and use the corresponding parity check. In particular, if we index the positions of the codeword by elements of $\mathbb{F}_h^m$. Then given the position $x \in \mathbb{F}_h^m$, the query set $S(x)$ is all the points other than x in a random r-flat L that passes through x. Given a codeword $c \in \mathcal{A}^*(r,m,h)$, we may reconstruct c_x by

$$A\left(c|_{S(x)}\right) = -\sum_{y \in Q(x)} c_y.$$

By definition, (A,S) is a smooth reconstruction procedure which makes h^r queries.

The locality of $\mathcal{A}^*(r,m,h)$ has been noticed before, for example in [41], where it was observed that these codes could be viewed as lifted parity check codes. However, as they note, these codes do not themselves make good LCCs—the reconstruction procedure cannot tolerate any errors in the chosen subspace, and thus the error rate ρ must tend to zero as the block length grows. Even though these codes are not good LCCs, we can use them in Theorem 6.13 to obtain good LCCs with sublinear query complexity, which can correct a constant fraction of errors. We will use the bound on the rate of $\mathcal{A}^*(1,m,h)$ from [41]:

Lemma 6.15 (Lemma 3.7 in [41])**.** *Choose $\ell = \varepsilon m$, with $h = p^\ell$ as above. The dimension of $\mathcal{A}^*(1,m,h)$ is at least $h^m - h^{m(1-\beta)}$, for $\beta = \beta(\varepsilon') = \Omega(2^{-2/\varepsilon'})$.*

We will apply Lemma 6.15 with

$$\varepsilon' = \frac{\varepsilon}{2} \qquad \text{and} \qquad m = \sqrt{\frac{\ln(2/\alpha)}{\varepsilon' \beta(\varepsilon') \ln(p)}},$$

to obtain a p-ary code $\mathcal{C}_0$ of length $d = p^{\varepsilon' m^2}$ with rate R_0 at least $1 - \alpha/2$ and which has a $(d-1)$-smooth reconstruction algorithm with query complexity $Q_0 = d^{\varepsilon'}$. To apply Theorem 6.13, fix any $\varepsilon, \alpha > 0$, sufficiently small. We set $\zeta = 2\ln(Q_0)$, and choose $\gamma = 1/4$ in Theorem 6.13, and use $\mathcal{C}_0$: the resulting expander code $\mathcal{C}$ has rate $1-\alpha$ and query complexity

$$Q \leq \left(\frac{n}{d}\right)^{\varepsilon}$$

for sufficiently large d. Finally, using the fact that $\lambda \le 2/\sqrt{d}$, we see that $\mathcal{C}$ corrects against a ρ fraction of errors, where

$$\rho = \frac{1}{5} d^{-6\varepsilon'}$$

again for sufficiently large d, as long as $\varepsilon < 1/12$. Assuming ε and α are small enough that d is a suitably large constant, this rate ρ is a positive constant, and we achieve the advertised results.

Multiplicity codes. Multiplicity codes [79] are themselves a family of constant-rate locally decodable codes. We can, however, use a multiplicity code of constant length as the inner code $\mathcal{C}_0$ in our construction. This results in a new family of constant-rate locally decodable codes. The parameters we obtain from this construction are slightly worse than the original multiplicity codes, and the main reason we include this example is novelty—these new codes have a very different structure than the original multiplicity codes.

For constants $\alpha', \varepsilon' > 0$, the multiplicity codes of [79] have length d and rate $R_0 = 1 - \alpha'$ and a $(d-1)$-smooth local reconstruction algorithm with query complexity $Q_0 = O(d^{\varepsilon'})$. To apply Theorem 6.13, we will choose $\zeta = C \ln(Q_0)$ for a sufficiently large constant C, and so the query complexity of $\mathcal{C}$ will be

$$Q = \left(\frac{n}{d}\right)^{(1+\beta)\varepsilon'}$$

for an arbitrarily small constant β. Thus, setting $\varepsilon = \varepsilon'(1+\beta)$, and $\alpha = 2\alpha'$, we obtain codes $\mathcal{C}$ with rate $1-\varepsilon$ and query complexity $(n/d)^{\varepsilon}$. As long as ε is sufficiently small, $\mathcal{C}$ can tolerate errors up to $\rho = C' d^{-C''\varepsilon}$ for constants C' and C'' (depending on the constants in the constructions of the multiplicity code, as well as on C above).

Multiplicity codes require sufficiently large block length d, on the order of

$$d \approx \left(\frac{1}{\alpha^2\varepsilon^3}\right)^{1/\varepsilon} \log\left(\frac{1}{\alpha\varepsilon}\right).$$

Choosing this d results in a requirement $\rho \leq 1/\text{poly}(\alpha\varepsilon)$. We remark that the distance of the multiplicity codes is on the order of $\delta_0 = \Omega(\alpha^2\varepsilon)$, and so the distance of the resulting expander code $\mathcal{C}$ is $\Omega(\alpha^4\varepsilon^2)$.

6.5 Conclusion

In the constant-rate regime, all known LDCs work by using a smooth local reconstruction algorithm. When the locality is, say, three, then with very high probability none of the queried positions will be corrupted. This reasoning fails for constant rate codes, which have larger query complexity: we expect a ρ fraction of errors in our queries, and this is often difficult to deal with. In this chapter, we made the low-query argument valid in a high-rate setting—any code with large enough rate and with a good local reconstruction algorithm can be used to make a full-blown locally correctable code.

The payoff of our approach is the first sublinear time algorithm for decoding expander codes. More precisely, we have shown that as long as the inner code $\mathcal{C}_0$ admits a smooth local reconstruction algorithm with appropriate parameters, then the resulting expander code $\mathcal{C}$ is a (n^ε, ρ)-LCC with rate $1 - \alpha$, for any $\alpha, \varepsilon > 0$ and some constant ρ. Further, we presented a decoding algorithm with runtime linear in the number of queries.

There are only two other constructions known in this regime, and and our constructions are substantially different. Expander codes are a natural construction, and it is our hope that the additional structure of our codes, as well as the extremely fast decoding time, will lead to new applications of local decodability.

Acknowledgements

The work in Chapter 6 originally appeared as [67] (conference version) and [68] (journal version), and is joint work with Brett Hemenway and Rafail Ostrovsky.

CHAPTER 7

Summary and conclusions

7.1 Summary of contributions

We have investigated two variants of coding theory from a rather non-standard view. In list decoding, we worked very hard to ignore some very nice algebraic structure, and focused instead on probabilistic and geometric considerations. In local decoding, we used a combinatorial and probabilistic approach to provide new constructions of LCCs, while to date only algebraic constructions were known.

As punchlines of our work on list decoding, we showed that random linear codes are (nearly) optimally list-decodable with high probability, and that there exist Reed-Solomon codes which are list-decodable beyond the Johnson bound. These questions had each been open for over 15 years. Along the way, we developed a toolkit which complements existing algebraic approaches. Our toolkit could be described as a general theory of "random stuff you can do to codes." This theory gives us some insight about the structure of list-decodability: while it may not be the case that a simple structural property (like distance) is enough to guarantee optimal list-decodability, it is the case that a simple structural property and a little bit of randomness (like distance and some random puncturing) *is* enough.

In local-decoding, we gave examples of constant-rate locally correctible codes, using expander graphs and some probabilistic arguments. Our constructions are the third known family of codes in this regime, and they are of a very different flavor: while existing approaches are algebraic, ours are are combinatorial. In fact, "our" constructions are actually expander codes, which are neither ours nor new. Thus, our work also gives sublinear time decoding algorithms for a well-studied family of codes.

Finally, and most importantly, we have perhaps improved life for Alice and Bob (Figure 7.1.)

Figure 7.1: Concrete results of the work in this dissertation.

7.2 Future work and open questions

Fortunately (from the perspective of obtaining future employment) we did not solve all of the problems.[1] We conclude with a few open problems raised by our work.

7.2.1 List decoding

In Chapter 3, we gave a very simple argument for the optimal list-decodability of random linear codes over constant-sized alphabets. The major open question of that chapter was to extend the argument to large alphabet sizes. We nearly did this in Chapter 4, but there were some obnoxious logarithmic factors, and our proof became

[1]Clearly, this was a deliberate decision.

much more complicated. It is natural to ask if either or both of these issues could be ameliorated.

Question 7.1. *Is it true that random linear codes of rate $\Omega(\varepsilon)$ are list-decodable up to radius $\rho = (1 - \varepsilon)$ for sufficiently large alphabet sizes q? And, if so, is there a simple proof?*

In Chapter 4, our main motivation was Reed-Solomon codes, and we showed that there exist Reed-Solomon codes which are list-decodable beyond the Johnson bound. Again, there is the open question of removing the logarithmic factors. Additionally, there's the problem of actually *finding* such a code.

Question 7.2. *For any $R = \omega(\varepsilon^2)$, is there a set of explicit evaluation points $\alpha_1, \ldots, \alpha_n$ so that the Reed-Solomon code of rate R with these evaluation points is list-decodable up to radius $\rho = 1 - \varepsilon$?*

One way to try to attack Question 7.2 is to find evaluation points that are suitable "structure-free." More precisely, the work of [12] shows that certain algebraic structure in the evaluation points is bad (in that it hinders list-decodability); our work shows that a lack of structure (random evaluation points) are good. Making this rigorous is an interesting direction.

Question 7.3. *Charactize algebraic structure that hinders list-decodability. More precisely, is there some (nontrivial) algebraic property so that*

(a) any Reed-Solomon code whose evaluation points avoid this property is list-decodable beyond the Johnson bound, and

(b) any Reed-Solomon code whose evaluation points have this property get stuck at the Johnson bound?

Even finding a nontrivial property so that (a) is true would be interesting, and, depending on the property, could answer Question 7.2.

Choosing evaluation points to be subspace evasive sets seems like a good candidate.

Question 7.3 leads naturally to a more general question about the structure of list-decoding. Our work on list-decoding gave a theory of "random stuff you can do to codes," and one take-away is that "most codes (derived from) codes with good structural properties are optimally list-decodable." When the structural property was distance, this gave a sort of randomized version of the Johnson bound. This was satisfying because it went beyond the actual Johnson bound, but unsatisfying because of the randomness. A very ambitious goal is to derandomize this approach, and to characterize (deterministically) the pathological cases which prevent the actual Johnson bound from working.

Question 7.4. *Is there a simple structural property A (like distance) and another simple structural property B (a generalization of an answer to Question 7.3 to arbitrary codes) so that having A and not B is a sufficient condition for (near) optimal list-decodability?*

Whether or not we can derandomize our results, we might ask whether or not we can do anything efficient with them. In Chapter 5, we focused our machinery on closing the gap between the combinatorial and probabilistic approach of this thesis and the existing algorithmic approaches. The holy grail for list-decoding is Problem 5.1, and we did not come close. Any further progress in this direction would be exciting.[2]

[2]Especially if it rests on the work in this dissertation.

Question 7.5. *Is there any way to apply Theorem 4.6 to obtain* efficiently decodable *list-decodable codes, with any nontrivial parameters?*

Finally, list-decodable codes are related to many pseudorandom objects. (See [107] for a nice overview of many of these connections). It is natural to ask if our machinery could be used there as well.

Question 7.6. *Can one extend the tools from this dissertation to answer open questions in pseudorandomness? As a concrete question, is a random linear extractor*[3] *an optimal strong extractor?*

The machinery of Gaussian processes, well-understood and also well-exploited in other areas, has a great deal of potential in coding theory and pseudorandomness. The problem of controlling the worst case of a random process, a.k.a., bounding $\mathbb{E}\sup[\text{stuff}]$, is ubiquitous in coding theory, pseudorandomness, and other areas of theoretical computer science. It is exciting to think how tools from (continuous) probability might be brought to bear in these (generally discrete) domains.

7.2.2 Local decoding

In Chapter 6, we gave a general framework for turning codes $\mathcal{C}_0$ with smooth local reconstruction algorithms into full-blown locally correctible codes. We gave two instantiations of such inner codes, both of which gave codes of arbitrarily high rate and query complexity n^{ε}. Many questions remain about how to choose inner codes. A major limitation on allowable inner codes is that the rate needs to be at least 1/2 in order to obtain an expander code with nontrivial rate. However, the argument for the rate of expander codes (which we sketched in Chapter 2) is known

[3]That is, use a random seed to choose a $m \times n$ matrix from some random subset of all such matrices, and use this matrix to map a low-entropy n-bit source to m bits of near-uniform randomness. This form of the question was asked to me by Swastik Kopparty and David Zuckerman. One can say something about strong *Renyi* extractors (that is, extractors for Renyi entropy) without too much trouble, but the question is for the standard definition of a strong extractor.

not to be tight [105]. If we could overcome this obstacle, it would give us access to a much larger class of codes to use as inner codes (for example, perhaps we could use Reed-Muller codes as an inner code).

Question 7.7. *What other families of inner codes result in locally correctible expander codes?*

A more specific form of Question 7.7, which was asked of me by Avi Wigderson and Shubhangi Saraf, is whether we can find suitable inner codes over $\mathbb{R}$. There are no known locally correctable codes over the reals in this regime, and currect evidence [6, 24] indicates that finding LCCs over $\mathbb{R}$ is harder than over finite fields.

Question 7.8. *Are there suitable inner codes which are linear over* $\mathbb{R}$*?*

A final question is whether or not techniques like this could be used to obtain codes with *logarithmic* query complexity.

Question 7.9. *Can the techniques of Chapter 6 be extended to produce codes with rate tending to* 1 *and query complexity (poly)logarithmic in* n*?*

In [21], it was conjectured that no such codes exist—if they did not, it would imply explicit families of rigid matrices.

Finally, we conclude with the obvious question raised by this dissertation.

Question 7.10. *May I please have a Ph.D.?*

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Erik Agrell, Alexander Vardy, and Kenneth Zeger. Upper bounds for constant-weight codes. *Information Theory, IEEE Transactions on*, 46(7):2373–2395, 2000.

[2] Noga Alon, Uriel Feige, Avi Wigderson, and David Zuckerman. Derandomized graph products. *Computational Complexity*, 5(1):60–75, 1995.

[3] Edward F. Assmus and Jennifer D. Key. *Designs and their Codes.* Cambridge University Press, 1994.

[4] Edward F. Assmus and Jennifer D. Key. Polynomial codes and finite geometries. In Vera Pless, Richard A Brualdi, and William Cary Huffman, editors, *Handbook of Coding Theory*, volume 2, pages 1269–1343. Elsevier, 1998.

[5] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–32, New York, NY, USA, 1991.

[6] Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, pages 519–528, 2011.

[7] Alexander Barg and Gilles Zemor. Error exponents of expander codes. *Information Theory, IEEE Transactions on*, 48(6):1725–1729, June 2002.

[8] Alexander Barg and Gilles Zemor. Concatenated codes: serial and parallel. *Information Theory, IEEE Transactions on*, 51(5):1625–1634, May 2005.

[9] Alexander Barg and Gilles Zemor. Distance properties of expander codes. *Information Theory, IEEE Transactions on*, 52(1):78–90, January 2006.

[10] Amos Beimel, Yuval Ishai, Eyal Kushilevitz, and Ilan Orlov. Share Conversion and Private Information Retrieval. In *Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC)*, pages 258–268, Los Alamitos, CA, USA, 2012.

[11] Avraham Ben-Aroya, Klim Efremenko, and Amon Ta-Shma. Local List Decoding with a Constant Number of Queries. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 715–722. IEEE, October 2010.

[12] Eli Ben-Sasson, Swastik Kopparty, and Jaikumar Radhakrishnan. Subspace polynomials and limits to list decoding of reed-solomon codes. *Information Theory, IEEE Transactions on*, 56(1):113–120, 2010.

[13] Vladamir .M. Blinovsky. Code bounds for multiple packings over a nonbinary finite alphabet. *Problems of Information Transmission*, 41(1):23–32, 2005.

[14] Vladamir M. Blinovsky. On the convexity of one coding-theory function. *Problems of Information Transmission*, 44(1):34–39, 2008.

[15] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

[16] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, December 1993.

[17] Jin-yi Cai, Aduri Pavan, and D. Sivakumar. On the hardness of permanent. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 90–99, 1999.

[18] Yeow M. Chee, Tao Feng, San Ling, Huaxiong Wang, and Liang F. Zhang. Query-Efficient Locally Decodable Codes of Subexponential Length. *Computational Complexity*, pages 1–31, August 2011.

[19] Qi Cheng and Daqing Wan. On the list and bounded distance decodability of reed-solomon codes. *SIAM Journal on Computing*, 37(1):195–209, 2007.

[20] Mahdi Cheraghchi, Venkatesan Guruswami, and Ameya Velingker. Restricted isometry of fourier matrices and list decodability of random linear codes. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 432–442, 2013.

[21] Zeev Dvir. On matrix rigidity and locally self-correctable codes. *Computational Complexity*, 20(2):367–388, 2011.

[22] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching Vector Codes. *SIAM Journal on Computing*, 40(4):1154–1178, January 2011.

[23] Zeev Dvir and Shachar Lovett. Subspace evasive sets. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 351–358, 2012.

[24] Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Breaking the quadratic barrier for 3-lccs over the reals. *arXiv preprint arXiv:1311.5102*, 2013.

[25] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 39–44. ACM, 2009.

[26] Klim Efremenko. From irreducible representations to locally decodable codes. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 327–338, New York, NY, USA, 2012.

[27] Peter Elias. List decoding for noisy channels. *Technical Report 335, Research Laboratory of Electronics, MIT*, 1957.

[28] Peter Elias. Error-correcting codes for list decoding. *Information Theory, IEEE Transactions on*, 37(1):5–12, 1991.

[29] Thomas Ericson and Victor Zinoviev. Spherical codes generated by binary partitions of symmetric pointsets. *IEEE transactions on information theory*, 41(1):107–129, 1995.

[30] Vitali Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. New results for learning noisy parities and halfspaces. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 563–574, 2006.

[31] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing.* Springer, 2013.

[32] Robert G. Gallager. Low Density Parity-Check Codes. Technical report, MIT, 1963.

[33] Peter Gemmell, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 33–42, New York, NY, USA, 1991.

[34] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4):169–174, September 1992.

[35] David Gillman. A chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, 1998.

[36] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first Annual ACM Symposium on Theory of Computing*, pages 25–32, 1989.

[37] Oded Goldreich, Dana Ron, and Madhu Sudan. Chinese remaindering with errors. In *Proceedings of the thirty-first Annual ACM Symposium on Theory of Computing*, pages 225–234, 1999.

[38] Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries: The highly noisy case. *SIAM Journal on Discrete Mathematics*, 13(4):535–570, 2000.

[39] Parikshit Gopalan. A fourier-analytic approach to reed-muller decoding. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 685–694, 2010.

[40] Parikshit Gopalan, Adam R. Klivans, and David Zuckerman. List-decoding reed-muller codes over small fields. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 265–274, 2008.

[41] Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science (ITCS)*, pages 529–540, 2013.

[42] Venkatesan Guruswami. Limits to list decodability of linear codes. In *Proceedings of the 34th annual ACM symposium on theory of computing (STOC)*, pages 802–811. ACM, 2002.

[43] Venkatesan Guruswami. *List Decoding of Error-Correcting Codes (Winning Thesis of the 2002 ACM Doctoral Dissertation Competition)*, volume 3282 of *Lecture Notes in Computer Science*. Springer, 2004.

[44] Venkatesan Guruswami, Johan Håstad, and Swastik Kopparty. On the list-decodability of random linear codes. *Information Theory, IEEE Transactions on*, 57(2):718–725, 2011.

[45] Venkatesan Guruswami, Johan Håstad, M. Sudan, and David Zuckerman. Combinatorial bounds for list decoding. *Information Theory, IEEE Transactions on*, 48(5):1021–1034, 2002.

[46] Venkatesan Guruswami and Piotr Indyk. Linear time encodable and list decodable codes. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, pages 126–135, 2003.

[47] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005.

[48] Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 608–617, 2013.

[49] Venkatesan Guruswami and Srivatsan Narayanan. Combinatorial limitations of average-radius list decoding. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 591–606. Springer, 2013.

[50] Venkatesan Guruswami and Atri Rudra. Limits to list decoding reed-solomon codes. *Information Theory, IEEE Transactions on*, 52(8):3642–3649, 2006.

[51] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[52] Venkatesan Guruswami and Atri Rudra. Error correction up to the information-theoretic limit. *Commun. ACM*, 52(3):87–95, 2009.

[53] Venkatesan Guruswami and Atri Rudra. The existence of concatenated codes list-decodable up to the hamming bound. *IEEE Transactions on Information Theory*, 56(10):5195–5206, 2010.

[54] Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. *Essential Coding Theory*. 2014. In progress; available from http://www.cse.buffalo.edu/~atri/courses/coding-theory/book/.

[55] Venkatesan Guruswami and Igor Shparlinski. Unconditional proof of tightness of johnson bound. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 754–755, 2003.

[56] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Proceedings of 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 28–39, 1998.

[57] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.

[58] Venkatesan Guruswami and Madhu Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the thirty-second Annual ACM Symposium on Theory of Computing*, pages 181–190, 2000.

[59] Venkatesan Guruswami and Madhu Sudan. Extensions to the johnson bound, 2001. Manuscript.

[60] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):20, 2009.

[61] Venkatesan Guruswami and Salil Vadhan. A lower bound on list size for list decoding. *Information Theory, IEEE Transactions on*, 56(11):5681–5688, 2010.

[62] Venkatesan Guruswami and Carol Wang. Linear-algebraic list decoding for variants of reed-solomon codes. *IEEE Transactions on Information Theory*, 59(6):3257–3268, 2013.

[63] Venkatesan Guruswami and Chaoping Xing. Folded codes from function field towers and improved optimal rate list decoding. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 339–350, 2012.

[64] Venkatesan Guruswami and Chaoping Xing. List decoding reed-solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. In *Proceedings of the 45th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 843–852, 2013.

[65] Venkatesan Guruswami and Chaoping Xing. Optimal rate list decoding of folded algebraic-geometric codes over constant-sized alphabets. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1858–1866, 2014.

[66] Richard W Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950.

[67] Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Locally correctability of expander codes. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 540–551, 2013.

[68] Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Locally correctability of expander codes. *Information and Computation*, 2014. To appear.

[69] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–562, 2006.

[70] Russell Impagliazzo and Valentine Kabanets. Constructive proofs of concentration bounds. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 617–631, 2010.

[71] Toshiya Itoh and Yasuhiro Suzuki. New Constructions for Query-Efficient Locally Decodable Codes of Subexponential Length. *Information and Systems, IEICE Transactions on*, E93-D(2):263–270, October 2010.

[72] Selmer Johnson. A new upper bound for error-correcting codes. *Information Theory, IEEE Transactions on*, 8(3):203–207, 1962.

[73] Selmer Johnson. Improved asymptotic bounds for error-correcting codes. *Information Theory, IEEE Transactions on*, 9(3):198–205, 1963.

[74] Jørn Justesen. Class of constructive asymptotically good algebraic codes. *Information Theory, IEEE Transactions on*, 18(5):652–656, 1972.

[75] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–86, 2000.

[76] Tali Kaufman, Shachar Lovett, and Ely Porat. Weight distribution and list-decoding size of reed–muller codes. *Information Theory, IEEE Transactions on*, 58(5):2689–2696, 2012.

[77] Iordanis Kerenidis and Ronald De Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *Proceedings of the thirty-fifth Annual ACM Symposium on Theory of Computing*, pages 106–115, 2003.

[78] Swastik Kopparty. List-decoding multiplicity codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:44, 2012.

[79] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 167–176, 2011.

[80] Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*, volume 23. Springer, 1991.

[81] Vladimir I Levenshtein. Universal bounds for codes and designs. In Vera Pless, Richard A Brualdi, and William Cary Huffman, editors, *Handbook of Coding Theory*. Elsevier Science Inc., 1998.

[82] Richard J. Lipton. Efficient checking of computations. In *Proceedings of the 7th Annual Symposium on Theoretical aspects of computer science (STACS)*, pages 207–215, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[83] Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[84] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error-correcting codes*. Elsevier, 1977.

[85] Grigory A. Margulis. Explicit group theoretical constructions of combinatorial schemes and their application to the design of expanders and concentrators. *Problems of Information Transmission*, 9(1):39–46, 1988.

[86] Moshe Morgenstern. Existence and explicit constructions of $q+1$ regular ramanujan graphs for every prime power q. *Journal of Combinatorial Theory, Series B*, 62(1):44–62, 1994.

[87] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the guruswami-sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–294, 2005.

[88] Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, pages 194–203, New York, NY, USA, 1994. ACM.

[89] Irving Reed. A class of multiple-error-correcting codes and the decoding scheme. *Information Theory, Transactions of the IRE Professional Group on*, 4(4):38–49, September 1954.

[90] Irving S Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial & Applied Mathematics*, 8(2):300–304, 1960.

[91] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 84–93, New York, NY, USA, 2005.

[92] Mark Rudelson. Contact points of convex bodies. *Israel Journal of Mathematics*, 101(1):93–124, 1997.

[93] Mark Rudelson and Roman Vershynin. On sparse reconstruction from fourier and gaussian measurements. *Communications on Pure and Applied Mathematics*, 61(8):1025–1045, 2008.

[94] Atri Rudra. *List decoding and property testing of error-correcting codes*. PhD thesis, University of Washington, 2007.

[95] Atri Rudra. Limits to list decoding of random codes. *Information Theory, IEEE Transactions on*, 57(3):1398–1408, 2011.

[96] Atri Rudra and Mary Wootters. Every list-decodable code for high noise has abundant near-optimal-rate puncturings. In *Proceedings of the 46th Annual ACM Symposium on the Theory of Computing (STOC)*, 2014. To appear.

[97] Claude Elwood Shannon. A mathematical theory of communication. *Bell System technical journal*, 27(3):379–423, 1948.

[98] Michael Sipser and Daniel A. Spielman. Expander codes. *Information Theory, IEEE Transactions on*, 42(6):1710–1722, 1996.

[99] D. A. Spielman. Highly fault-tolerant parallel computation. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 154–163. IEEE, October 1996.

[100] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *Information Theory, IEEE Transactions on*, 42(6):1723–1731, November 1996.

[101] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

[102] Madhu Sudan. List decoding: algorithms and applications. *SIGACT News*, 31(1):16–27, 2000.

[103] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the xor lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

[104] Michel Talagrand. *The generic chaining: upper and lower bounds for stochastic processes.* Springer, 2005.

[105] R. Michael Tanner. A recursive approach to low complexity codes. *Information Theory, IEEE Transactions on*, 27(5):533–547, 1981.

[106] Nicolas Thierry-Mieg. A new pooling strategy for high-throughput screening: the shifted transversal design. *BMC bioinformatics*, 7(1):28, 2006.

[107] Salil Vadhan. The unified theory of pseudorandomness: guest column. *ACM SIGACT News*, 38(3):39–54, 2007.

[108] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

[109] Edward J. Weldon. Justesen's construction–the low-rate case (corresp.). *Information Theory, IEEE Transactions on*, 19(5):711–713, 1973.

[110] Stephen B Wicker and Vijay K Bhargava. *Reed-Solomon codes and their applications.* John Wiley & Sons, 1999.

[111] Mary Wootters. On the list decodability of random linear codes with large error rates. In *Proceedings of the 45th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 853–860, 2013.

[112] John M. Wozencraft. List Decoding. *Quarterly Progress Report, Research Laboratory of Electronics, MIT*, 48:90–95, 1958.

[113] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM*, 55(1), 2008.

[114] Sergey Yekhanin. Locally Decodable Codes. *Foundations and Trends in Theoretical Computer Science*, 2010.

[115] Gilles Zemor. On expander codes. *Information Theory, IEEE Transactions on*, 47(2):835–837, 2001.

[116] Victor V. Zyablov and Mark S. Pinsker. List cascade decoding. *Problems of Information Transmission*, 17(4):29–34, 1981.