Foundations and Trends® in
sample
Vol. xx, No xx (xxxx) 1–114

DOI: xxxxxx

now
the essence of knowledge

Locally Decodable Codes

Sergey Yekhanin[1]

[1] *Microsoft Research Silicon Valley, 1065 La Avenida, Mountain View, CA, 94043, USA, yekhanin@microsoft.com*

Abstract

Locally decodable codes are a class of "error-correcting codes". Error-correcting codes help ensure reliability when transmitting information over noisy channels. They allow a sender of a message to add redundancy to messages, encoding bit strings representing messages into longer bit strings called codewords, in a way that the message can still be recovered even if a certain fraction of the codeword bits are corrupted. Classical error-correcting codes however do not work well when one is working with massive messages, because their decoding time increases (at least) linearly with the length of the message. As a result in typical applications the message is first partitioned into small blocks, each of which is then encoded separately. Such encoding allows efficient random-access retrieval of the message, but yields poor noise resilience.

Locally decodable codes are codes intended to address this seeming conflict between efficient retrievability and reliability. They are codes that simultaneously provide efficient random-access retrieval and high noise resilience by allowing reliable reconstruction of an arbitrary bit of the message from looking at only a small number of randomly chosen codeword bits. This text introduces and motivates locally decodable

codes, and discusses the central results of the subject. In particular, local decodability comes at the price of certain loss in terms of code efficiency, and this text describes the currently known limits on the efficiency that is achievable.

Contents

1

Introduction

Locally Decodable Codes (LDCs) are a special kind of error-correcting codes. Error-correcting codes are used to ensure reliable transmission of information over noisy channels as well as to ensure reliable storage of information on a medium that may be partially corrupted over time (or whose reading device is subject to errors).

In both of these applications the message is typically partitioned into small blocks and then each block is encoded separately. Such encoding strategy allows efficient random-access retrieval of the information, since one needs to decode only the portion of data one is interested in. Unfortunately, this strategy yields very poor noise resilience, since in case even a single block (out of possibly tens of thousands) is completely corrupted some information is lost. In view of this limitation it would seem preferable to encode the whole message into a single codeword of an error-correcting code. Such solution clearly improves the robustness to noise, but is also hardly satisfactory, since one now needs to look at the whole codeword in order to recover any particular bit of the message (at least when classical error-correcting codes are used). Such decoding complexity is prohibitive for modern massive data-sets.

Locally decodable codes are error-correcting codes that avoid the

problem mentioned above by having extremely efficient *sublinear-time* decoding algorithms. More formally, an r-query locally decodable code C encodes k-bit messages $\mathbf{x}$ in such a way that one can probabilistically recover any bit $\mathbf{x}(i)$ of the message by querying only r bits of the (possibly corrupted) codeword $C(\mathbf{x})$, where r can be as small as 2.

Example. The classical Hadamard code encoding k-bit messages to 2^k-bit codewords provides the simplest nontrivial example of locally decodable codes. In what follows, let $[k]$ denote the set $\{1, \ldots, k\}$. Every coordinate in the Hadamard code corresponds to one (of 2^k) subsets of $[k]$ and stores the XOR of the corresponding bits of the message $\mathbf{x}$. Let $\mathbf{y}$ be an (adversarially corrupted) encoding of $\mathbf{x}$. Given an index $i \in [k]$ and $\mathbf{y}$, the Hadamard decoder picks a set S in $[k]$ uniformly at random and outputs the XOR of the two coordinates of $\mathbf{y}$ corresponding to sets S and $S \triangle \{i\}$. (Here, $\triangle$ denotes the symmetric difference of sets such as $\{1,4,5\} \triangle \{4\} = \{1,5\}$, and $\{1,4,5\} \triangle \{2\} = \{1,2,4,5\}$). It is not difficult to verify that if $\mathbf{y}$ differs from the correct encoding of $\mathbf{x}$ in at most δ fraction of coordinates than with probability $1-2\delta$ both decoder's queries go to uncorrupted locations. In such case, the decoder correctly recovers the i-th bit of $\mathbf{x}$. The Hadamard code allows for a super-fast recovery of the message bits (such as, given a codeword corrupted in 0.1 fraction of coordinates, one is able to recover any bit of the message with probability 0.8 by reading only two codeword bits).

The main parameters of interest in locally decodable codes are the codeword length and the query complexity. The length of the code measures the amount of redundancy that is introduced into the message by the encoder. The query complexity counts the number of bits that need to be read from the (corrupted) codeword in order to recover a single bit of the message. Ideally, one would like to have both of these parameters as small as possible. One however can not minimize the length and the query complexity simultaneously. There is a trade-off. On one end of the spectrum we have classical error correcting codes that have both query complexity and codeword length proportional to the message length. On the other end we have the Hadamard code

that has query complexity 2 and codeword length exponential in the message length. Establishing the optimal trade-off between the length and the query complexity is the major goal of research in the area of locally decodable codes.

Interestingly, the natural application of locally decodable codes to data transmission and storage described above is neither the historically earliest nor the most important. LDCs have a host of applications in other areas of theoretical computer science.

1.1 Families of locally decodable codes

One can informally classify the known families of locally decodable codes into three broad categories based on the relation between the message length k and the query complexity r.

1. Low query complexity. Here we look at codes where r is a constant independent of k or some very slowly growing function of k. Such codes have important applications in cryptography to constructions of private information retrieval schemes. Early examples of such codes are the Hadamard code and the Reed Muller (RM) code that is sketched below.

Reed Muller code. The code is specified by three integer parameters, an alphabet size q, a number of variables n, and a degree $d < q-1$. The code encodes $k = \binom{n+d}{d}$-long q-ary messages to q^n-long codewords. We fix a certain collection of vectors $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$ in $\mathbb{F}_q^n$. A message $\mathbf{x}$ is encoded by a complete $\mathbb{F}_q^n$-evaluation of a polynomial $F \in \mathbb{F}_q[z_1, \ldots, z_n]$ of degree up to d, such that for all $i \in [k]$, $\mathbf{x}(i) = F(\mathbf{w}_i)$. Our choice of W ensures that such a polynomial exists for any $\mathbf{x}$. Given $i \in [k]$ and a δ-corrupted evaluation of F the Reed Muller decoder needs to recover the value of F at $\mathbf{w}_i$. To do this the decoder picks a random affine line L through $\mathbf{w}_i$ and reads the (corrupted) values of F at $d+1$ points of $L \setminus \{\mathbf{w}_i\}$. Next, the decoder uses univariate polynomial interpolation to recover the restriction of F to L. Each query of the decoder samples a random location, thus with probability at least $1 - (d+1)\delta$, it never queries a corrupted coordinate and decodes correctly. Setting d and q to be constant and letting n grow one gets r-query codes of length $N = \exp\left(k^{1/(r-1)}\right)$.

Other families of codes in this category are the recursive codes of Beimel et al. and the Matching Vector (MV) codes. MV codes offer the best-known trade-off between the query complexity and the codeword length of locally decodable codes for small values of query complexity. In particular they give three-query codes of length $N(k)$ where N grows slower than any function of the form $\exp(k^\epsilon)$.[1] In this book we cover the construction of matching vector codes in full detail.

2. Medium query complexity. Here we look at codes with $r = \log^c k$, for some $c > 1$. Such codes have been used in constructions of probabilistically checkable proofs. They also have applications to worst-case to average-case reductions in computational complexity theory. Setting $d = n^c$, $q = \Theta(d)$ in the definition of Reed Muller codes, and letting the number of variables n grow to infinity yields codes of query complexity $\log^c k$ and codeword length $N = k^{1+1/(c-1)+o(1)}$. These are the best-known locally decodable codes in this regime.

3. High query complexity. Here we look at codes with $r = k^\epsilon$, for some $\epsilon > 0$. This is the only regime where we (so far) have locally decodable codes of positive rate, i.e., codeword length proportional to message length. Such codes are potentially useful for data transmission and storage applications. The early examples of such codes are the Reed Muller codes with the number of variables $n = 1/\epsilon$, growing d, and $q = \Theta(d)$. Such setting of parameters yields codes of query complexity $r = k^\epsilon$ and rate $\epsilon^{\Theta(1/\epsilon)}$. The rate is always below $1/2$. Another family of codes in the high query complexity category is the family of multiplicity codes. Multiplicity codes are based on evaluating high degree multivariate polynomials together with their partial derivatives. Multiplicity codes extend Reed Muller codes; inherit the local-decodability of these codes, and at the same time achieve better tradeoffs and flexibility in their rate and query complexity. In particular for all $\alpha, \epsilon > 0$ they yield locally decodable codes of query complexity $r = k^\epsilon$ and rate $1 - \alpha$. In this book we cover multiplicity codes in full detail.

[1] Throughout the book we use the standard notation $\exp(x) = 2^{O(x)}$.

1.2 Organization

The goal of this survey is to summarize the state of the art in locally decodable codes. Our main focus is on multiplicity codes and on matching vector codes. The book is organized into eight chapters.

In chapter 2 we formally define locally decodable codes and give a detailed treatment of Reed Muller codes. In chapter 3 we study multiplicity codes. We show how multiplicity codes generalize Reed Muller codes and obtain bounds on their rate and query complexity.

In Chapter 4 we introduce the concept of matching vectors and present a transformation that turns an arbitrary family of such vectors into a family of locally decodable (matching vector) codes. We provide a detailed comparison between the parameters of matching vector codes based on the currently largest known matching families and Reed Muller codes. Chapter 5 contains a systematic study of families of matching vectors. We cover several constructions as well as impossibility results.

In chapter 6 we deal with lower bounds for the codeword length of locally decodable codes. In chapter 7 we discuss some prominent applications of locally decodable codes, namely, applications to private information retrieval schemes, secure multi party computation, and average case complexity. Finally, in the last chapter we list (and comment on) the most exciting open questions relating to locally decodable codes and private information retrieval schemes.

1.3 Notes

We now review the history of locally decodable codes. Ideas behind the early constructions of LDCs go back to classical codes [77, chapter 10], named after their discoverers, Reed and Muller. Muller discovered the codes [70] in the 1950s, and Reed proposed the majority logic decoding [81]. Since then, local decodability of these codes has been exploited extensively. In particular, in the early 1990s a number of theoretical computer science papers developed and used local decoding algorithms for some variants of these codes [11, 67, 24, 43, 44, 5, 78]. The first formal definition of locally decodable codes was given however only in

2000 by Katz and Trevisan [60], who cited Leonid Levin for inspiration. See also [90].

Today there are three families of locally decodable codes that surpass Reed Muller codes in terms of query complexity vs. codeword length trade-off. These are the recursive codes of Beimel et al. [15] (see also [97]), the matching vector codes [99, 79, 61, 38, 59, 35, 18, 69, 19, 84], and the multiplicity codes [63]. Matching vector codes offer the best-known trade-off between the query complexity and the codeword length of locally decodable codes for small values of query complexity. Multiplicity codes are the best-known locally decodable codes for large values of query complexity.

The first lower bounds for the codeword length of locally decodable codes were obtained in [60]. Further work on lower bounds includes [48, 31, 74, 62, 94, 95, 96, 41]. It is known that 1-query LDCs do not exist [60]. The length of optimal 2-query LDCs was settled in [62] and is exponential in the message length. However for values of query complexity $r \geq 3$ we are still very far from closing the gap between lower and upper bounds. Specifically, the best lower bounds to date are of the form $\tilde{\Omega}\left(k^{1+1/(\lceil r/2 \rceil - 1)}\right)$ due to [95], while the best upper bounds are super-polynomial in k when r is a constant [38, 69].

2

Preliminaries

In this chapter we formally define locally decodable and locally correctable codes, and study parameters of Reed Muller locally decodable codes. We start by setting up the notation and terminology used in the remainder of the book.

- $[k] = \{1, \ldots, k\}$;
- $\mathbb{F}_q$ is a finite field of q elements;
- $\mathbb{F}_q^*$ is the multiplicative group of $\mathbb{F}_q$;
- $(\mathbf{x}, \mathbf{y})$ stands for the dot product of vectors $\mathbf{x}$ and $\mathbf{y}$;
- $\Delta(\mathbf{x}, \mathbf{y})$ denotes the relative Hamming distance between $\mathbf{x}$ and $\mathbf{y}$, i.e., the fraction of coordinates where $\mathbf{x}$ and $\mathbf{y}$ differ;
- For a vector $\mathbf{w} \in \mathbb{F}_q^n$ and an integer $l \in [n]$, $\mathbf{w}(l)$ denotes the l-th coordinate of $\mathbf{w}$;
- A D-evaluation of a function h defined over a domain D, is a vector of values of h at all points of D;
- With a slight abuse of terminology we often refer to a dimension n of a vector $\mathbf{x} \in \mathbb{F}_q^n$ as its *length*.

We now proceed to define locally decodable codes.

2.1 Locally decodable and locally correctable codes

A q-ary LDC encoding length k messages to length N codewords has three parameters: r, δ, and ϵ. Informally an (r, δ, ϵ)-locally decodable code encodes k-long messages $\mathbf{x}$ to N-long codewords $C(\mathbf{x})$, such that for every $i \in [k]$, the coordinate value $\mathbf{x}_i$ can be recovered with probability $1 - \epsilon$, by a randomized decoding procedure that makes only r queries, even if the codeword $C(x)$ is corrupted in up to δN locations. Formally,

---

Definition 2.1. A q-ary code $C : \mathbb{F}_q^k \to \mathbb{F}_q^N$ is said to be (r, δ, ϵ)-locally decodable if there exists a randomized decoding algorithm $\mathcal{A}$ such that

(1) For all $\mathbf{x} \in \mathbb{F}_q^k$, $i \in [k]$ and all vectors $\mathbf{y} \in \mathbb{F}_q^N$ such that $\Delta(C(\mathbf{x}), \mathbf{y}) \leq \delta$:

$$\Pr[\mathcal{A}(\mathbf{y}, i) = \mathbf{x}(i)] \geq 1 - \epsilon,$$

where the probability is taken over the random coin tosses of the algorithm $\mathcal{A}$.

(2) $\mathcal{A}$ reads at most r coordinates to $\mathbf{y}$.

We would like to have LDCs that for a given message length k and alphabet size q have small values of r, N and ϵ and a large value of δ. However typically the parameters are not regarded as equally important. In applications of locally decodable codes to data transmission and storage one wants δ to be a large constant, (ideally, close to $1/4$ for binary codes), and the codeword length N to be small. At the same time the exact number of queries r is not very important provided that it is much smaller than k. Similarly the exact value of $\epsilon < 1/2$ is not important since one can easily amplify ϵ to be close to 0, by running the decoding procedure few times and taking a majority vote. At the same time in applications of locally decodable codes in cryptography one thinks of $\delta > 0$ and $\epsilon < 1/2$ as constants whose values are of low significance and focuses on the trade-off between r and N, with emphasis on very small values of r such as $r = 3$ or $r = 4$.

A locally decodable code is called *linear* if C is a linear transformation over $\mathbb{F}_q$. All constructions of locally decodable codes considered in the book yield linear codes. While our main interest is in binary codes we deal with codes over larger alphabets as well.

A locally decodable code allows to probabilistically decode any coordinate of a message by probing only few coordinates of its corrupted encoding. A stronger property that is desirable in certain applications is that of local correctability allowing to efficiently recover not only coordinates of the message but also arbitrary coordinates of the encoding. Reed Muller codes that we discuss in the next sections are locally correctable.

Definition 2.2. A code (set) C in the space $\mathbb{F}_q^N$ is (r, δ, ϵ)-locally correctable if there exists a randomized correcting algorithm $\mathcal{A}$ such that

(1) For all $\mathbf{c} \in C$, $i \in [N]$ and all vectors $\mathbf{y} \in \mathbb{F}_q^N$ such that $\Delta(\mathbf{c}, \mathbf{y}) \leq \delta$:

$$\Pr[\mathcal{A}(\mathbf{y}, i) = \mathbf{c}(i)] \geq 1 - \epsilon,$$

where the probability is taken over the random coin tosses of the algorithm $\mathcal{A}$.

(2) $\mathcal{A}$ reads at most r coordinates to $\mathbf{y}$.

The next lemma shows how one can obtain a locally decodable code from any locally correctable code that is a linear subspace of $\mathbb{F}_q^N$.

Lemma 2.3. Let q be a prime power. Suppose $C \subseteq \mathbb{F}_q^N$ is a (r, δ, ϵ)-locally correctable code that is linear subspace; then there exists a q-ary (r, δ, ϵ)-locally decodable linear code C' encoding messages of length $\dim C$ to codewords of length N.

Proof. Let $I \subseteq [N]$ be a set of $k = \dim C$ information coordinates of C, (i.e., a set of coordinates whose values uniquely determine an element of C.) For $\mathbf{c} \in C$ let $\mathbf{c}|_I \in \mathbb{F}_q^k$ denote the restriction of $\mathbf{c}$ to coordinates in I. Given a message $\mathbf{x} \in \mathbb{F}_q^k$ we define $C'(\mathbf{x})$ to be the unique element

$\mathbf{c} \in C$ such that $c|_I = \mathbf{x}$. It is easy to see that local correctability of C yields local decodability of C'. □

In what follows we often implicitly assume the translation between locally correctable linear codes and locally decodable codes. Specifically, we sometimes talk about message length (rather than dimension) of such codes.

2.2 Reed Muller locally decodable codes

The key idea behind early locally decodable codes is that of polynomial interpolation. Messages are encoded by complete evaluations of low degree multivariate polynomials over a finite field. Local decodability is achieved through reliance on the rich structure of short local dependencies between such evaluations at multiple points.

Recall that a Reed Muller locally decodable code is specified by three integer parameters, namely, a prime power (alphabet size) q, number of variables n, and a degree $d < q - 1$. The q-ary code consists of $\mathbb{F}_q^n$-evaluations of all polynomials of total degree at most d in the ring $\mathbb{F}_q[z_1, \ldots, z_n]$. Such code encodes $k = \binom{n+d}{d}$-long messages over $\mathbb{F}_q$ to q^n-long codewords. In sections 2.2.1–2.2.3 we consider three local correctors (decoders) for RM codes of increasing level of sophistication. Finally, in section 2.2.4 we show how one can turn non-binary RM LDCs into binary.

2.2.1 Basic decoding on lines

In this section we present the simplest local corrector for Reed Muller codes. To recover the value of a degree d polynomial $F \in \mathbb{F}_q[z_1, \ldots, z_n]$ at a point $\mathbf{w} \in \mathbb{F}_q^n$ it shoots a random affine line through $\mathbf{w}$ and then relies on the local dependency between the values of F at some $d + 1$ points along the line.

Proposition 2.4. Let n and d be positive integers. Let q be a prime power, $d < q-1$; then there exists a linear code of dimension $k = \binom{n+d}{d}$ in $\mathbb{F}_q^N$, $N = q^n$, that is $(d+1, \delta, (d+1)\delta)$-locally correctable for all δ.

Proof. The code consists of $\mathbb{F}_q^n$-evaluations of all polynomials of total degree at most d in the ring $\mathbb{F}_q[z_1, \ldots, z_n]$. The local correction procedure is the following. Given an evaluation of a polynomial F corrupted in up to δ fraction of coordinates and a point $\mathbf{w} \in \mathbb{F}_q^n$ the local corrector picks a vector $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random and considers a line

$$L = \{\mathbf{w} + \lambda \mathbf{v} \mid \lambda \in \mathbb{F}_q\}$$

through $\mathbf{w}$. Let S be an arbitrary subset of $\mathbb{F}_q^*$, where $|S| = d + 1$. The corrector queries coordinates of the evaluation vector corresponding to points $\mathbf{w} + \lambda \mathbf{v}$, $\lambda \in S$ to obtain values $\{e_\lambda\}$. Next, it recovers the unique univariate polynomial h, $\deg h \leq d$, such that $h(\lambda) = e_\lambda$, for all $\lambda \in S$, and outputs $h(0)$.

Note that in case all queries of our corrector go to uncorrupted locations h is the restriction of F to L, and $h(0) = F(\mathbf{w})$. It remains to note that since each individual query of the corrector samples a uniformly random location, with probability at least $1 - (d + 1)\delta$, it never query a corrupted coordinate. □

We say that an r-query code C *tolerates* a δ fraction of errors if C is (r, δ, ϵ)-locally correctable (decodable) for some $\epsilon < 1/2$. Observe that codes given by proposition 2.4 can only tolerate $\delta < 1/2(d + 1)$. Thus the fraction tolerable noise rapidly deteriorates with an increase in the query complexity. In the following section we present a better local corrector for RM codes that tolerates δ close to $1/4$ independent of the number of queries.

2.2.2 Improved decoding on lines

In contrast to the setting of proposition 2.4 here we require that d is substantially smaller than q. To recover the value of a degree d polynomial $F \in \mathbb{F}_q[z_1, \ldots, z_n]$ at a point $\mathbf{w} \in \mathbb{F}_q^n$ the corrector shoots a random affine line through $\mathbf{w}$ and then relies on the high redundancy among the values of F along the line.

Proposition 2.5. Let $\sigma < 1$ be a positive real. Let n and d be positive integers. Let q be a prime power such that $d \leq \sigma(q - 1) - 1$; then there

exists a linear code of dimension $k = \binom{n+d}{d}$ in $\mathbb{F}_q^N$, $N = q^n$, that is $(q-1, \delta, 2\delta/(1-\sigma))$-locally correctable for all δ.

Proof. The code is exactly the same as in proposition 2.4, and the correction procedure is related to the procedure above. Given a δ-corrupted evaluation of a degree d polynomial F and a point $\mathbf{w} \in \mathbb{F}_q^n$ the corrector picks a vector $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random and considers a line

$$L = \{\mathbf{w} + \lambda \mathbf{v} \mid \lambda \in \mathbb{F}_q\}$$

through $\mathbf{w}$. The corrector queries coordinates of the evaluation vector corresponding to points $\mathbf{w} + \lambda\mathbf{v}$, $\lambda \in \mathbb{F}_q^*$ to obtain values $\{e_\lambda\}$. Next, it recovers the unique univariate polynomial h, $\deg h \leq d$, such that $h(\lambda) = e_\lambda$, for all but at most $\lfloor (1-\sigma)(q-1)/2 \rfloor$ values of $\lambda \in \mathbb{F}_q^*$, and outputs $h(0)$. If such a polynomial h does not exist the corrector outputs zero. The search for h can be done efficiently using the Berlekamp-Welch algorithm [68] for decoding Reed Solomon codes.

It remains to note that since each individual query of the corrector samples a uniformly random location, by Markov's inequality the probability that $(1-\sigma)(q-1)/2$ or more of the queries go to corrupted locations is at most $2\delta/(1-\sigma)$. Therefore with probability at least $1 - 2\delta/(1-\sigma)$, h is the restriction of F to L, and $h(0) = F(\mathbf{w})$. □

When σ is small the local corrector given by proposition 2.5 tolerates a nearly $1/4$ fraction of errors. In the following section we present an even better corrector that tolerates a nearly $1/2$ fraction of errors, which is optimal for unique decoding.

2.2.3 Decoding on curves

In what follows we again require that d is substantially smaller than q. To recover the value of a degree d polynomial $F \in \mathbb{F}_q[z_1, \ldots, z_n]$ at a point $\mathbf{w} \in \mathbb{F}_q^n$ the corrector shoots a random parametric degree two curve through $\mathbf{w}$ and then relies on the high redundancy among the values of F along the curve. The advantage upon the decoder of proposition 2.5 comes from the fact that points on a random curve

(in contrast to points on a random line) constitute a two-independent sample from the underlying space.

Proposition 2.6. Let $\sigma < 1$ be a positive real. Let n and d be positive integers. Let q be a prime power such that $d \leq \sigma(q-1) - 1$; then there exists a linear code of dimension $k = \binom{n+d}{d}$ in $\mathbb{F}_q^N$, $N = q^n$, that for all positive $\delta < 1/2 - \sigma$ is $(q-1, \delta, O_{\sigma,\delta}(1/q))$-locally correctable.

Proof. The code is exactly the same as in propositions 2.4 and 2.5, and the correction procedure is related to the procedures above. Given a δ-corrupted evaluation of a degree d polynomial F and a point $\mathbf{w} \in \mathbb{F}_q^n$ the corrector picks vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_q^n$ uniformly at random and considers a degree two curve

$$\chi = \{\mathbf{w} + \lambda \mathbf{v}_1 + \lambda^2 \mathbf{v}_2 \mid \lambda \in \mathbb{F}_q\}$$

through $\mathbf{w}$. The corrector tries to reconstruct a restriction of F to χ, which is a polynomial of degree up to $2d$. To this end the corrector queries coordinates of the evaluation vector corresponding to points $\chi(\lambda) = \mathbf{w} + \lambda \mathbf{v}_1 + \lambda^2 \mathbf{v}_2$, $\lambda \in \mathbb{F}_q^*$ to obtain values $\{e_\lambda\}$. Next, it recovers the unique univariate polynomial h, $\deg h \leq 2d$, such that $h(\lambda) = e_\lambda$, for all but at most $\lfloor (1 - 2\sigma)(q-1)/2 \rfloor$ values of $\lambda \in \mathbb{F}_q^*$, and outputs $h(0)$. If such a polynomial h does not exist the corrector outputs 0. It is not hard to verify that the corrector succeeds if the number of queries that go to corrupted locations is at most $\lfloor (1 - 2\sigma)(q-1)/2 \rfloor$.

Below we analyze the success probability of the corrector. For $\mathbf{a} \in \mathbb{F}_q^n$ and $\lambda \in \mathbb{F}_q^*$ consider a random variable $x_{\mathbf{a}}^\lambda$, which is the indicator variable of the event $\chi(\lambda) = \mathbf{a}$. Let $E \subseteq \mathbb{F}_q^n$, $|E| \leq \delta N$ be the set of $\mathbf{a} \in \mathbb{F}_q^n$ such that the values of F at $\mathbf{a}$ are corrupted. For every $\lambda \in \mathbb{F}_q^*$ consider a random variable $x^\lambda \in \{0, 1\}$ defined as

$$x^\lambda = \sum_{\mathbf{a} \in E} x_{\mathbf{a}}^\lambda.$$

Note that variables $\left\{x^\lambda\right\}, \lambda \in \mathbb{F}_q^*$ are pairwise independent. For every $\lambda \in \mathbb{F}_q^*$ expectation and variance satisfy

$$\mathbb{E}\left[x^\lambda\right] \leq \delta \quad \text{and} \quad \mathbb{D}\left[x^\lambda\right] \leq \delta - \delta^2.$$

Finally consider a random variable

$$x = \sum_{\lambda \in \mathbb{F}_q^*} x^\lambda,$$

that counts the number of corrector's queries that go to corrupted locations. By pairwise independence we have

$$\mathbb{E}[x] \le (q-1)\delta \quad \text{and} \quad \mathbb{D}[x] \le (q-1)(\delta - \delta^2).$$

By Chebyshev's inequality [2] we have

$$\Pr\left[x \ge \left\lfloor \frac{(1-2\sigma)(q-1)}{2} \right\rfloor\right] \le \frac{4(\delta-\delta^2)}{(q-1)(1-2(\sigma+\delta))^2} = O_{\sigma,\delta}\left(\frac{1}{q}\right).$$

This concludes the proof. □

2.2.4 Binary codes

Propositions 2.4, 2.5, and 2.6 yield non-binary codes. As we stated earlier our main interest is in binary codes. The next lemma extends proposition 2.6 to produce binary codes that tolerate a nearly $1/4$ fraction of errors, which is optimal for unique decoding over $\mathbb{F}_2$. The idea behind the proof is fairly standard and involves concatenation.

Proposition 2.7. Let $\sigma < 1$ be a positive real. Let n and d be positive integers. Let $q = 2^b$ be a power of two such that $d \le \sigma(q-1)-1$. Suppose that there exists a binary linear code C_{inner} of distance μB encoding b-bit messages to B-bit codewords; then there exists a linear code C of dimension $k = \binom{n+d}{d} \cdot b$ in $\mathbb{F}_2^N$, $N = q^n \cdot B$, that for all positive $\delta < (1/2 - \sigma)\mu$ is $((q-1)B, \delta, O_{\sigma,\mu,\delta}(1/q))$-locally correctable.

Proof. We define the code C to be the concatenation [40] of the q-ary code C_{outer} used in propositions 2.4–2.6 and the binary code C_{inner}. In order to recover a single bit, the local corrector recovers the symbol of the q-ary alphabet that the bit falls into. Given a δ-corrupted concatenated evaluation of a degree d polynomial F and a point $\mathbf{w} \in \mathbb{F}_q^n$ the corrector acts similarly to the corrector from the proposition 2.6.

Specifically, it picks vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_q^n$ uniformly at random and considers a degree two curve

$$\chi = \{\mathbf{w} + \lambda \mathbf{v}_1 + \lambda^2 \mathbf{v}_2 \mid \lambda \in \mathbb{F}_q\}$$

through $\mathbf{w}$. To recover $F(\mathbf{w})$ the corrector attempts to reconstruct a restriction of F to χ, which is a polynomial of degree up to $2d$. To this end the corrector queries all $(q-1)B$ codeword coordinates corresponding to encodings of values of F at points $\chi(\lambda) = \mathbf{w} + \lambda \mathbf{v}_1 + \lambda^2 \mathbf{v}_2$, $\lambda \in \mathbb{F}_q^*$ and then recovers the unique univariate polynomial $h \in \mathbb{F}_q[\lambda]$, $\deg h \leq 2d$, such that C_{inner}-encodings of values of h along $\mathbb{F}_q^*$ agree with all but at most $\lfloor (1-2\sigma)\mu(q-1)B/2 \rfloor$ observed binary values. If such a polynomial h does not exist the corrector outputs 0. It is not hard to verify that the corrector succeeds if the number of queries that go to corrupted locations is at most $\lfloor (1-2\sigma)\mu(q-1)B/2 \rfloor$. Decoding can be done efficiently provided that C_{inner} has an efficient decoder.

Below we analyze the success probability of the corrector. For every $\mathbf{a} \in \mathbb{F}_q^n$ let $t_{\mathbf{a}}$ denote the number of corrupted coordinates in the C_{inner}-encoding of the value of F at $\mathbf{a}$. We have

$$\sum_{\mathbf{a} \in \mathbb{F}_q^n} t_{\mathbf{a}} \leq \delta q^n B.$$

For $\mathbf{a} \in \mathbb{F}_q^n$ and $\lambda \in \mathbb{F}_q^*$ consider a random variable $x_{\mathbf{a}}^{\lambda}$, which is the indicator variable of the event $\chi(\lambda) = \mathbf{a}$. For every $\lambda \in \mathbb{F}_q^*$ consider a random variable

$$x^{\lambda} = \sum_{\mathbf{a} \in \mathbb{F}_q^n} t_{\mathbf{a}} x_{\mathbf{a}}^{\lambda}.$$

Note that variables $\left\{x^{\lambda}\right\}, \lambda \in \mathbb{F}_q^*$ are pairwise independent and identically distributed. For every $\lambda \in \mathbb{F}_q^*$ we have

$$\mathbb{E}\left[x^{\lambda}\right] \leq \delta B \quad \text{and} \quad \mathbb{D}\left[x^{\lambda}\right] \leq (\delta - \delta^2)B^2.$$

Finally consider a random variable

$$x = \sum_{\lambda \in \mathbb{F}_q^*} x^{\lambda},$$

that counts the number of corrector's queries that go to corrupted locations. By pairwise independence we have

$$\mathbb{E}[x] \leq \delta(q-1)B \quad \text{and} \quad \mathbb{D}[x] \leq (\delta - \delta^2)(q-1)B^2.$$

By Chebyshev's inequality [2] we have

$$\Pr\left[x \geq \left\lfloor \frac{(1-2\sigma)\mu(q-1)B}{2} \right\rfloor\right] \leq$$
$$\leq \frac{4(\delta-\delta^2)}{(q-1)((1-2\sigma)\mu - 2\delta)^2} = O_{\sigma,\mu,\delta}\left(\frac{1}{q}\right).$$

This concludes the proof. □

2.3 Summary of parameters

In the previous section we gave a detailed treatment of Reed Muller locally decodable codes. These codes yield the shortest known LDCs in the medium query complexity regime ($r = \log^c k$, $c > 1$).

The method behind Reed Muller codes is simple and general. It yields codes for all possible values of query complexity r, i.e., one can set r to be an arbitrary function of the message length k by specifying an appropriate relation between n and d in propositions 2.5–2.7 and letting these parameters grow to infinity. Increasing d relative to n yields shorter codes of larger query complexity.

Below we present asymptotic parameters of several families of binary locally decodable codes based on Reed Muller codes.

- $r = O(1)$. Proposition 2.4 yields r-query LDCs of length $\exp\left(k^{1/(r-1)}\right)$ over an alphabet of size $O(r)$.
- $r = O(\log k \log\log k)$. In proposition 2.7 set $d = n$, $q = cd$ for a large constant c, and let n grow while concatenating with asymptotically good binary codes of relative distance μ close to half. This yields a family of r-query binary locally correctable codes that encode k-bit messages to $k^{O(\log\log k)}$-bit codewords and tolerate a nearly $1/4$ fraction of errors (depending on c and μ).

- $r \leq (\log k)^t$, for constant $t > 1$. In proposition 2.7 set $d = n^t$, $q = cd$ and let n grow while concatenating with asymptotically good binary codes of relative distance close to half. This yields a family of r-query binary locally correctable codes that encode k-bit messages to $k^{1+1/(t-1)+o(1)}$-bit codewords and tolerate a nearly $1/4$ fraction of errors.
- $r = O(k^{1/t} \log^{1-1/t} k)$, for integer constant $t \geq 1$. In proposition 2.7 set $n = t$, $q = cd$ and let d grow while concatenating with asymptotically good binary codes of relative distance close to half. This yields a family of r-query binary locally correctable codes that encode k-bit messages to $t^{t+o(t)} \cdot k$-bit codewords and tolerate a nearly $1/4$ fraction of errors.

We summarize the parameters of binary locally correctable codes obtained above in the following table.

r	N
$O(1)$	$\exp\left(k^{1/(r-1)}\right)$
$O(\log k \log\log k)$	$k^{O(\log\log k)}$
$(\log k)^t, t > 1$	$k^{1+1/(t-1)+o(1)}$
$O(k^{1/t} \log^{1-1/t} k), t \geq 1$	$t^{t+o(t)} \cdot k$

2.4 Notes

The concept of local correctability originates in [67, 23]. Local correctors of propositions 2.4, 2.5, and 2.6 come respectively from [11, 67], [43], and [44].

Propositions 2.6 and 2.7 give locally correctable codes that tolerate the amount of error that is nearly optimal for unique (even non-local) decoding ($1/2$ fraction of errors over large alphabets, $1/4$ over $\mathbb{F}_2$). An important model of error correction that generalizes unique decoding is that of list decoding [39, 53]. In that model the decoder is allowed to output a small list of codewords rather than a single codeword. Decoding is considered successful if the transmitted codeword appears in the list. List decoding allows for error-correction beyond the "half the minimum distance barrier". One can show that Reed Muller codes are

locally list decodable from the nearly optimal amount of noise [4, 90]. However we are not going to discuss these results in this book.

3

Multiplicity codes

In this chapter we study multiplicity codes. These codes generalize Reed Muller codes and greatly improve upon them in the regime of high rate. Recall that with Reed Muller codes, for the code to have any distance, the degrees of the polynomials used to define the code need to be smaller than the field size. Multiplicity codes, however, use much higher degree polynomials and thus have significantly improved rates, and compensate for the loss in distance by evaluating polynomials *together with their partial derivatives*.

In what follows we review the construction of multiplicity codes in full detail. Our main results give codes that simultaneously have rate approaching one, and allow for local decoding with arbitrary polynomially-small number of queries.

In the next section, we exhibit the main ideas behind multiplicity codes by presenting the simplest family of such codes. In section 3.2 we state the main theorems regarding the asymptotic parameters. In section 3.3, we formally define multiplicity codes, calculate their rate and distance, state a lemma implying their local decodability, and show how it implies the main theorems. In section 3.4 we present the local correction algorithms. Finally in appendix 3.5 we analyze the running

time of local decoders, and in appendix 3.6 we review the properties Hasse derivatives that are important for our code constructions.

3.1 Introduction

We start this section by recalling an example of the Reed Muller codes (section 2.2) based on bivariate polynomials and why it is locally correctable. We then introduce the simplest example of a multiplicity code based on bivariate polynomials, which has improved rate, and see how to locally correct it with essentially the same query complexity. Our presentation here in not entirely formal. Finally, we mention how general multiplicity codes are defined and some of the ideas that go into locally correcting them.

3.1.1 Bivariate Reed Muller codes

Let q be a prime power, let $\delta > 0$ and let integer $d = (1-\delta)q$. The Reed Muller code of degree d bivariate polynomials over $\mathbb{F}_q$ is defined as follows. The coordinates of the code are indexed by elements of $\mathbb{F}_q^2$, and so $N = q^2$. The codewords correspond to bivariate polynomials of total degree at most d over $\mathbb{F}_q$. The codeword corresponding to the polynomial $F(x_1, x_2)$ is the vector

$$C(F) = \langle F(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q^2} \in \mathbb{F}_q^{q^2}.$$

Because two distinct polynomials of degree at most d can agree on at most d/q-fraction of the points in $\mathbb{F}_q^2$, this code has distance $\delta = 1 - d/q$. Any polynomial of degree at most d is specified by one coefficient for each of the $\binom{d+2}{2}$ monomials, and so the message length $k = \binom{d+2}{2}$. Thus when q grows the rate of this code is $\binom{d+2}{2}/q^2 \approx (1-\delta)^2/2$. Notice that this code cannot have rate more than $1/2$.

Local correction of Reed Muller codes: Given a received word $\mathbf{y} \in (\mathbb{F}_q)^{q^2}$ such that $\mathbf{y}$ is close in Hamming distance to the codeword corresponding to $F(x_1, x_2)$, let us recall how the correction algorithm from section 2.2.2 works. Given a coordinate $\mathbf{w} \in \mathbb{F}_q^2$, we want to recover the "corrected" symbol at coordinate $\mathbf{w}$, namely $F(\mathbf{w})$. The algorithm picks a random direction $\mathbf{v} \in \mathbb{F}_q^2$ and looks at the restriction of $\mathbf{y}$ to coordinates in the line $L = \{\mathbf{w} + \lambda \mathbf{v} \mid \lambda \in \mathbb{F}_q\}$. With high probability

over the choice of $\mathbf{v}$, $\mathbf{y}$ and $C(F)$ will agree on many positions of L. Now $C(F)|_L$ is simply the vector consisting of evaluations of the univariate polynomial $f(\lambda) = F(\mathbf{w}+\lambda\mathbf{v}) \in \mathbb{F}_q[\lambda]$, which is of degree $\leq d$. Thus $\mathbf{y}|_L$ gives us q "noisy" evaluations of a polynomial $f(\lambda)$ of degree $\leq (1-\delta)\cdot q$; this enables us to recover $f(\lambda)$. Evaluating $f(\lambda)$ at $\lambda = 0$ gives us $F(\mathbf{w})$, as desired. Notice that this decoding algorithm makes q queries, which is $O(k^{1/2})$.

3.1.2 Bivariate Multiplicity Codes

We now introduce the simplest example of multiplicity codes, which already achieves a better rate than the Reed Muller code above, while being locally correctable with only a constant factor more queries.

Let q be a prime power, let $\delta > 0$ and let integer $d = 2(1-\delta)q$, which is twice what it was in the Reed Muller example above. The multiplicity code of *order two* evaluations of degree d bivariate polynomials over $\mathbb{F}_q$ is the code defined as follows. As before, the coordinates are indexed by $\mathbb{F}_q^2$ (so $N = q^2$) and the codewords are indexed by bivariate polynomials of degree at most d over $\mathbb{F}_q$. However the alphabet will now be $\mathbb{F}_q^3$. The codeword corresponding the polynomial $F(x_1, x_2)$ is the vector

$$C(F) = \left\langle \left(F(\mathbf{w}), \frac{\partial F}{\partial x_1}(\mathbf{w}), \frac{\partial F}{\partial x_2}(\mathbf{w}) \right) \right\rangle_{\mathbf{w}\in\mathbb{F}_q^2} \in \left(\mathbb{F}_q^3\right)^{q^2}.$$

In words, the $\mathbf{w}$ coordinate consists of the evaluation of F and its formal partial derivatives $\frac{\partial F}{\partial x_1}$ and $\frac{\partial F}{\partial x_2}$ at $\mathbf{w}$. Because two distinct polynomials of degree at most d can agree *with multiplicity two* on at most $d/2q$-fraction of the points in $\mathbb{F}_q^2$ (a fact we establish in appendix 3.6), this code has distance $\delta = 1 - d/2q$. Since the alphabet size is now q^3, the message length k equals the number of q^3-ary symbols required to specify a polynomial of degree at most d; this is clearly $\binom{d+2}{2}/3$. Thus the rate of this code is $\binom{d+2}{2}/(3q^2) \approx 2(1-\delta)^2/3$.

Summarizing the differences between this multiplicity code with the Reed-Muller code described earlier:

- Instead of polynomials of degree $(1-\delta)q$, we consider polynomials of degree double of that.

- Instead of evaluating the polynomials, we take their *order two* evaluation.

This yields a code with the same distance, while the rate improved from below $1/2$ to nearly $2/3$.

Local correction of multiplicity codes: Given a received word $\mathbf{y} \in \left(\mathbb{F}_q^3\right)^{q^2}$ such that $\mathbf{y}$ is close in Hamming distance to the codeword corresponding to $F(x_1, x_2)$, we now show how to locally correct. Given a point $\mathbf{w} \in \mathbb{F}_q^2$, we want to recover the "corrected" symbol at coordinate $\mathbf{w}$, namely $\left(F(\mathbf{w}), \frac{\partial F}{\partial x_1}(\mathbf{w}), \frac{\partial F}{\partial x_2}(\mathbf{w})\right)$. Again, the algorithm picks a random direction $\mathbf{v} \in \mathbb{F}_q^2$ and looks at the restriction of $\mathbf{y}$ to coordinates in the line $L = \{\mathbf{w} + \lambda\mathbf{v} \mid \lambda \in \mathbb{F}_q\}$. With high probability over the choice of $\mathbf{v}$, we will have that $\mathbf{y}|_L$ and $C(F)|_L$ agree in many locations. Our intermediate goal will be to recover the univariate polynomial[1] $f(\lambda) = F(\mathbf{w} + \lambda\mathbf{v})$.

The important observation is that for every $\lambda_0 \in \mathbb{F}_q$, the $\mathbf{w} + \lambda_0\mathbf{v}$ coordinate of $C(F)$ completely determines both the value and the first derivative of the univariate polynomial $f(\lambda)$ at the point λ_0. Indeed,

$$\begin{aligned} f(\lambda_0) &= F(\mathbf{w} + \mathbf{v}\lambda_0), \\ \frac{\partial f}{\partial \lambda}(\lambda_0) &= \frac{\partial F}{\partial x_1}(\mathbf{w} + \mathbf{v}\lambda_0) \cdot \mathbf{v}(1) + \frac{\partial F}{\partial x_2}(\mathbf{w} + \mathbf{v}\lambda_0) \cdot \mathbf{v}(2), \end{aligned}$$

where the last identity follows by the chain rule. Thus our knowledge of $\mathbf{y}|_L$ gives us access to q "noisy" evaluations of the polynomial $f(\lambda)$ and its derivative $\frac{\partial f}{\partial \lambda}(\lambda_0)$, where $f(\lambda)$ is of degree $\leq 2(1-\delta)q$. It turns out that this is enough to recover the polynomial $f(\lambda)$. Evaluating $f(\lambda)$ at $\lambda = 0$ gives us $F(\mathbf{w})$. Evaluating the derivative $\frac{\partial f}{\partial \lambda}(\lambda)$ at $\lambda = 0$ gives us the directional derivative of F at $\mathbf{w}$ in the direction $\mathbf{v}$ (which equals $\frac{\partial F}{\partial x_1}(\mathbf{w}) \cdot \mathbf{v}(1) + \frac{\partial F}{\partial x_2}(\mathbf{w}) \cdot \mathbf{v}(2)$).

We have clearly progressed towards our goal of computing the tuple $\left(F(\mathbf{w}), \frac{\partial F}{\partial x_1}(\mathbf{w}), \frac{\partial F}{\partial x_2}(\mathbf{w})\right)$, but we are not yet there. The final observation is that if we pick another direction $\mathbf{v}'$, and repeat the above process

[1] Unlike in the Reed Muller case, here there is a distinction between recovering $f(\lambda)$ and recovering $C(F)|_L$. It turns out that recovering $C(F)|_L$ given only $\mathbf{y}|_L$ is impossible.

to recover the directional derivative of F at $\mathbf{w}$ in direction $\mathbf{v}'$, then the two directional derivatives of F at $\mathbf{w}$ in directions $\mathbf{v}, \mathbf{v}'$ together suffice to recover $\frac{\partial F}{\partial x_1}(\mathbf{w})$ and $\frac{\partial F}{\partial x_2}(\mathbf{w})$, as desired. This algorithm makes $2q$ queries, which is $O(k^{1/2})$.

3.1.3 General Multiplicity codes

The basic example of a multiplicity code above already achieves rate above 1/2. To get codes of rate approaching 1, in the following sections we modify the above example by considering evaluations of all derivatives of F up to an even higher order. In order to locally recover the higher-order derivatives of F at a point $\mathbf{w}$, the decoding algorithm will pick many random lines passing through $\mathbf{w}$, try to recover the restriction of F to those lines, and combine all these recovered univariate polynomials in a certain way.

To reduce the query complexity to $O(k^\epsilon)$ for small ϵ, we modify the above example by considering multivariate polynomials in a larger number of variables n. The local decoding algorithm for this case, in order to locally recover at a point $\mathbf{w} \in \mathbb{F}_q^n$, decodes by picking random lines passing through $\mathbf{w}$; the reduced query complexity occurs because lines (with only q points) are now much smaller relative to a higher dimensional space $\mathbb{F}_q^n$.

Increasing both the maximum order of derivative taken and the number of variables simultaneously yields multiplicity codes with rate close to one and arbitrary low polynomial query complexity.

3.2 Main results

Below we state the two main theorems regarding the asymptotic parameters of multiplicity codes. The first theorem gives non-linear codes over growing alphabets.

Theorem 3.1. For every $0 < \epsilon, \alpha < 1$, for infinitely many k, there is a code C over an alphabet Σ, with $|\Sigma| \leq k^{O(1)}$, such that C has message length k, rate at least $1 - \alpha$, relative distance $\delta \geq \epsilon\alpha/2$, and is $(O(k^\epsilon), \epsilon\alpha/20, 0.2)$-locally correctable.

The next theorem is the analogue of theorem 3.1 for small alphabets. It gives linear codes. These codes are obtained by simply concatenating multiplicity codes with suitable good linear codes over the small alphabets. In particular by lemma 2.3, it shows the existence of locally decodable codes with the same parameters.

Theorem 3.2. Let p be a prime power. For every $\epsilon, \alpha > 0$, there exists $\delta > 0$, such that for infinitely many k, there is a linear code C over the alphabet $\Sigma = \mathbb{F}_p$, such that C has message length k, rate at least $1 - \alpha$, and is $(O(k^{\epsilon}), \delta, 0.2)$-locally correctable.

The proofs of the theorems above appear in section 3.3.3.

3.3 The code construction

In this section we formally define multiplicity codes, calculate their rate and distance, and state the main lemma 3.8 implying their local correctability. We then show how multiplicity codes imply the main theorems of the previous section.

First, we review some preliminaries on derivatives and multiplicities. We will define our codes using the *Hasse* derivative, which is a variant of the usual notion of derivative of a polynomial, and is more suitable for use in fields of small characteristic.

3.3.1 Derivatives and multiplicities

We start with some notation. For a vector $\mathbf{i} = (\mathbf{i}(1), \ldots, \mathbf{i}(n))$ of non-negative integers, its *weight*, denoted $\mathrm{wt}(\mathbf{i})$, equals $\sum_{j=1}^{n} \mathbf{i}(j)$.

Let $\mathbb{F}$ be a field and $\mathbb{F}[x_1, \ldots, x_n] = \mathbb{F}[\mathbf{x}]$ be the polynomial ring. For a vector of non-negative integers $\mathbf{i} = (\mathbf{i}(1), \ldots, \mathbf{i}(n))$, let $\mathbf{x}^{\mathbf{i}}$ denote the monomial $\prod_{j=1}^{n} x_j^{\mathbf{i}(j)} \in \mathbb{F}[\mathbf{x}]$. Note that the total degree of this monomial equals $\mathrm{wt}(\mathbf{i})$.

Definition 3.3 ((Hasse) Derivative). For $F(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and non-negative vector $\mathbf{i}$, the $\mathbf{i}$-th Hasse derivative of F, denoted $F^{(\mathbf{i})}(\mathbf{x})$, is the coefficient of $\mathbf{z}^{\mathbf{i}}$ in the polynomial $\tilde{F}(\mathbf{x}, \mathbf{z}) = F(\mathbf{x} + \mathbf{z}) \in \mathbb{F}[\mathbf{x}, \mathbf{z}]$.

Thus,

$$F(\mathbf{x}+\mathbf{z}) = \sum_{\mathbf{i}} F^{(\mathbf{i})}(\mathbf{x})\mathbf{z}^{\mathbf{i}}. \tag{3.1}$$

Observe that for all $F, G \in \mathbb{F}[\mathbf{x}]$, and $\lambda \in \mathbb{F}$,

$$(\lambda F)^{(\mathbf{i})}(\mathbf{x}) = \lambda F^{(\mathbf{i})}(\mathbf{x}) \quad \text{and} \quad F^{(\mathbf{i})}(\mathbf{x}) + G^{(\mathbf{i})}(\mathbf{x}) = (F+G)^{(\mathbf{i})}(\mathbf{x}). \tag{3.2}$$

The Hasse derivative typically behaves like the standard formal derivative, but with some key differences that make it more useful / informative over finite fields. For instance, the first two formal derivatives of the univariate polynomial x^2 over $\mathbb{F}_2$ are

$$\left(x^2\right)' = 2x = 0 \quad \text{and} \quad \left(x^2\right)'' = 0,$$

while for the Hasse derivative we have

$$\left(x^2\right)^{(1)} = 2x = 0 \quad \text{and} \quad \left(x^2\right)^{(2)} = 1.$$

We review the basic properties of Hasse derivatives in appendix 3.6. We are now ready to define the notion of the (zero-)multiplicity of a polynomial at any given point.

Definition 3.4 (Multiplicity). For $F(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and $\mathbf{w} \in \mathbb{F}^n$, the *multiplicity* of F at $\mathbf{w} \in \mathbb{F}^n$, denoted $\mathrm{mult}(F, \mathbf{w})$, is the largest integer m such that for every non-negative vector $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) < m$, we have $F^{(\mathbf{i})}(\mathbf{w}) = 0$. (If m may be taken arbitrarily large, $\mathrm{mult}(F, \mathbf{w}) = \infty$).

Note that $\mathrm{mult}(F, \mathbf{w}) \geq 0$ for every $\mathbf{w}$. The main technical fact we will need about derivatives and multiplicities is a bound on the number of points that a low-degree polynomial can vanish on with high multiplicity. We state this lemma below. The proof appears in appendix 3.6.

Lemma 3.5. Let $F \in \mathbb{F}[\mathbf{x}]$ be a nonzero n-variate polynomial of total degree at most d. Then for any finite set $S \subseteq \mathbb{F}$,

$$\sum_{\mathbf{w} \in S^n} \mathrm{mult}(F, \mathbf{w}) \leq d \cdot |S|^{n-1}.$$

In particular, for any integer $m > 0$,

$$\Pr_{\mathbf{w} \in S^n}[\mathrm{mult}(F, \mathbf{w}) \geq m] \leq \frac{d}{m|S|}.$$

3.3.2 Definition of multiplicity codes

We now come to the definition of multiplicity codes.

Definition 3.6. Let m, d, n be nonnegative integers and let q be a prime power. Let

$$\Sigma = \mathbb{F}_q^{\binom{n+m-1}{n}} = \mathbb{F}_q^{\{\mathbf{i} \,:\, \mathrm{wt}(\mathbf{i}) < m\}}.$$

For $F(x_1, \ldots, x_n) \in \mathbb{F}_q[x_1, \ldots, x_n]$, we define the order m evaluation of F at $\mathbf{w}$, denoted $F^{(<m)}(\mathbf{w})$, to be the vector

$$\left(F^{(\mathbf{i})}(\mathbf{w})\right)_{\mathrm{wt}(\mathbf{i})<m} \in \Sigma.$$

We define the multiplicity code C of order m evaluations of degree d polynomials in n variables over $\mathbb{F}_q$ as follows. The code is over the alphabet Σ, and has length q^n. The coordinates are indexed by elements of $\mathbb{F}_q^n$. For each polynomial $F(\mathbf{x}) \in \mathbb{F}_q[x_1, \ldots, x_n]$ with $\deg(F) \leq d$, there is a codeword in C given by:

$$\mathrm{Enc}_{m,d,n,q}(F) = \left(F^{(<m)}(\mathbf{w})\right)_{\mathbf{w} \in \mathbb{F}_q^n} \in (\Sigma)^{q^n}.$$

In the construction above we typically think of n, m, and d as fixed, and q growing to infinity. It is easy to see that bivariate multiplicity codes detailed in section 3.1 are a special case of the above definition.

In the next lemma, we calculate the rate and relative distance of multiplicity codes. The nice feature of the behavior here is that if we keep the distance δ fixed and let the multiplicity parameter m grow, the rate of these codes improves (and approaches $(1-\delta)^n$).

Lemma 3.7. Let C be the multiplicity code of order m evaluations of degree d polynomials in n variables over the field $\mathbb{F}_q$. Then C has relative distance which is at least $\delta = 1 - \frac{d}{mq}$ and rate $\frac{\binom{d+n}{n}}{\binom{m+n-1}{n} q^n}$, which is at least

$$\left(\frac{m}{n+m}\right)^n \cdot \left(\frac{d}{mq}\right)^n \geq \left(1 - \frac{n^2}{m}\right)(1-\delta)^n.$$

Proof. The alphabet size equals $q^{\binom{n+m-1}{n}}$. The length equals q^n. To calculate the relative distance, consider two arbitrary codewords $\mathbf{c}_1 = \mathrm{Enc}_{m,d,n,q}(F_1), \mathbf{c}_2 = \mathrm{Enc}_{m,d,n,q}(F_2)$, where $F_1 \neq F_2$. For any coordinate $\mathbf{w} \in \mathbb{F}_q^n$ where the codewords $\mathbf{c}_1, \mathbf{c}_2$ agree, we have $F_1^{(<m)}(\mathbf{w}) = F_2^{(<m)}(\mathbf{w})$. Thus for any such $\mathbf{w}$, we have $(F_1 - F_2)^{(\mathbf{i})}(\mathbf{w}) = 0$ for each $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) < m$, and hence $\mathrm{mult}(F_1 - F_2, \mathbf{w}) \geq m$. From the bound on the number of high-multiplicity zeroes of multivariate polynomials, lemma 3.5, the fraction of $\mathbf{w} \in \mathbb{F}_q^n$ on which this can happen is at most $\frac{d}{mq}$. The minimum distance δ of the multiplicity code is therefore at least $1 - \frac{d}{mq}$.

A codeword is specified by giving coefficients to each of the monomials of degree at most d. Thus the number of codewords equals $q^{\binom{d+n}{n}}$. Thus the rate equals

$$\begin{aligned}
\frac{\binom{d+n}{n}}{\binom{m+n-1}{n} q^n} &= \frac{\prod_{j=0}^{n-1} (d+n-j)}{\prod_{j=1}^{n} ((m+n-j)q)} \\
&\geq \left(\frac{1}{1+\frac{n}{m}}\right)^n \left(\frac{d}{mq}\right)^n \\
&\geq \left(1 - \frac{n^2}{m}\right) (1-\delta)^n .
\end{aligned}$$

□

The next lemma, which will be the focus of the rest of this chapter, shows that multiplicity codes are locally correctable.

Lemma 3.8. Let C be the multiplicity code of order m evaluations of degree d polynomials in n variables over $\mathbb{F}_q$. Let $\delta = 1 - \frac{d}{mq}$ be the lower bound for the relative distance of C. Suppose

$$q > \max\left\{10n, \frac{d+15m}{m}, 12(m+1)\right\};$$

then C is $(O(m)^n \cdot q, \delta/10, 0.2)$-locally correctable.

The proof of this lemma appears in section 3.4.

3.3.3 Proofs of the main theorems

We now show how to instantiate multiplicity codes to prove our main asymptotic theorems 3.1 and 3.2 based on lemma 3.8.

Proof. [of theorem 3.1] Recall that we are trying to construct, for every $0 < \epsilon, \alpha < 1$, for infinitely many k, a code over an alphabet of size $k^{O(1)}$, with message length k, rate $1-\alpha$, distance $\delta \geq \epsilon\alpha/2$, and locally correctable with $O(k^\epsilon)$ queries from $\delta/10$-fraction errors.

Pick $n = \lceil 1/\epsilon \rceil$. Observe that

$$1 - \alpha < (1 - \epsilon\alpha/2)^{2/\epsilon} < (1 - \epsilon\alpha/2)^{\lceil 1/\epsilon \rceil}.$$

Pick m to be the smallest positive integer so that

$$1 - \frac{n^2}{m} > \frac{1-\alpha}{(1-\epsilon\alpha/2)^n}.$$

Observe that n and m are constants. For every large enough prime power q, we will construct a code with codeword length $N = q^n$. Let $\delta \geq \epsilon\alpha/2$ be such that $d = (1-\delta) \cdot m \cdot q$ is an integer and

$$1 - \frac{n^2}{m} > \frac{1-\alpha}{(1-\delta)^n}.$$

Let C be the multiplicity code of order m evaluations of degree d polynomials in n variables over $\mathbb{F}_q$. Observe that C has codeword length N and is over an alphabet of size $q^{\binom{n+m-1}{n}} = N^{O(1)} = k^{O(1)}$. By Lemma 3.7, C has relative distance at least $\delta \geq \epsilon\alpha/2$ and rate at least

$$\left(1 - \frac{n^2}{m}\right) \cdot (1-\delta)^n > 1 - \alpha.$$

By lemma 3.8, C can be locally corrected from $\delta/10$-fraction errors using $O(N^{1/n}) = O(N^\epsilon) = O(k^\epsilon)$ queries. This completes the proof of theorem 3.1. □

Finally, we complete the proof of theorem 3.2, by concatenating suitable multiplicity codes with good linear codes over small alphabets.

Proof. [of theorem 3.2] Set $\alpha_1 = \alpha/2$ and $\epsilon_1 = \epsilon/2$. As in the proof of theorem 3.1, there are constants n and m such that for every large

enough prime power q, there is a multiplicity code with codeword length $N_1 = q^n$, rate $1 - \alpha_1$, relative distance at least $\delta_1 = \epsilon_1\alpha_1/2$, over an alphabet Σ_1 of size $q^{\binom{n+m-1}{n}}$, and locally correctable from $\delta_1/10$ fraction of errors with $O(N_1^{\epsilon_1})$ queries. We will take such codes C_1 where $q = p^t$ for integers $t > 0$.

We now pick another code C_2 of message length $\binom{n+m-1}{n} \cdot t$ that is $\mathbb{F}_p$-linear and has rate $1 - \alpha_1$ and use it to encode the symbols of C_1. The resulting concatenated code C is $\mathbb{F}_p$-linear (this follows from the linearity of C_2 and additivity of C_1 coming from equation (3.2)), and has distance δ and rate R that are at least the products of the corresponding parameters of C_1 and C_2. In particular, if we take C_2 to be a code of constant relative distance $\delta_2 > 0$; then C has codeword length

$$N = q^n \cdot \binom{n+m-1}{n} \cdot t \cdot \frac{1}{1-\alpha_1},$$

rate at least $1 - \alpha$ and constant relative distance (as N grows) $\delta > 0$.

We now argue that the code C is locally correctable. To locally correct some coordinate of a codeword of C given access to a corrupted codeword of C, we first run the local corrector for C_1 to decode the coordinate of C_1 that contains that coordinate of C. Whenever this local corrector wants to query a certain coordinate of C_1, we recover that symbol by decoding the corresponding codeword of C_2. The query complexity of the local corrector for C is clearly $O(N^{\epsilon_1} \log N) = O(k^{\epsilon})$. It remains to note that in case the total fraction of errors is below $\delta_1\delta_2/20$, all but $\delta_1/10$ fraction of the C_2 blocks will have less than $\delta_2/2$-fraction errors, and can be correctly recovered by the decoder for C_2. Thus the local corrector for C_1 will run correctly, and this yields the desired corrected coordinate of C. □

3.4 Local correction

In this section, we prove that multiplicity codes are locally correctable. Suppose we are dealing with the multiplicity code of order m evaluations of degree d polynomials in n variables over $\mathbb{F}_q$. Let Σ be the alphabet for this code. Let $\mathbf{y} : \mathbb{F}_q^n \to \Sigma$ be a received word. Suppose F is a polynomial over $\mathbb{F}_q$ in n variables of degree at most d such that

$\Delta(\mathbf{y}, \mathrm{Enc}_{m,d,n,q}(F))$ is small. Let $\mathbf{w} \in \mathbb{F}_q^n$. Let us show how to locally recover $F^{(<m)}(\mathbf{w})$ given oracle access to $\mathbf{y}$.

The main idea behind the local corrector is to pick many random affine lines containing $\mathbf{w}$, and to consider the restriction of $\mathbf{y}$ to those lines. With high probability over a random direction $\mathbf{v} \in \mathbb{F}_q^n$, by looking at the restriction of $\mathbf{y}$ to the line $\{\mathbf{w} + \lambda\mathbf{v} \mid \lambda \in \mathbb{F}_q\}$ and "decoding" it, we will be able to recover the univariate polynomial $F(\mathbf{w} + \lambda\mathbf{v}) = f(\lambda)$. Knowing this univariate polynomial will tell us a certain linear combination of the various derivatives of F at the point $\mathbf{w}$. Combining this information for various directions $\mathbf{v}$, we will know a system of various linear combinations of the numbers $\left\{F^{(\mathbf{i})}(\mathbf{w})\right\}_{\mathrm{wt}(\mathbf{i})<m}$. Solving this linear system, we get $F^{(\mathbf{i})}(\mathbf{w})$ for each $\mathbf{i}$, as desired.

To implement this strategy we need to relate the derivatives of the restriction of a multivariate polynomial F to a line to the derivatives of the polynomial F itself. Fix $\mathbf{w}, \mathbf{v} \in \mathbb{F}_q^n$, and consider the polynomial $f(\lambda) = F(\mathbf{w} + \lambda\mathbf{v}) \in \mathbb{F}_q[\lambda]$.

- $f(\lambda)$ **and the derivatives of** F **at** $\mathbf{w}$**:**
  By the definition of Hasse derivatives,
  $$f(\lambda) = \sum_{\mathbf{i}} F^{(\mathbf{i})}(\mathbf{w})\mathbf{v}^{\mathbf{i}}\lambda^{\mathrm{wt}(\mathbf{i})}.$$
  Grouping terms, we see that:
  $$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} F^{(\mathbf{i})}(\mathbf{w})\mathbf{v}^{\mathbf{i}} = \text{coefficient of } \lambda^e \text{ in } f(\lambda). \tag{3.3}$$
- **Derivatives of** f **at** λ_0 **and derivatives of** F **at** $\mathbf{w}+\lambda_0\mathbf{v}$**:**
  Let λ_0 be a fixed point in $\mathbb{F}_q$. By the definition of Hasse derivatives, we get the following two identities:
  $$F(\mathbf{w} + (\lambda_0 + \lambda)\mathbf{v}) = f(\lambda_0 + \lambda) = \sum_j f^{(j)}(\lambda_0)\lambda^j.$$
  $$F(\mathbf{w} + (\lambda_0 + \lambda)\mathbf{v}) = \sum_{\mathbf{i}} F^{(\mathbf{i})}(\mathbf{w} + \lambda_0\mathbf{v})(\lambda\mathbf{v})^{\mathbf{i}}.$$
  Thus,
  $$f^{(j)}(\lambda_0) = \sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=j} F^{(\mathbf{i})}(\mathbf{w} + \lambda_0\mathbf{v})\mathbf{v}^{\mathbf{i}}. \tag{3.4}$$

In particular, $f^{(j)}(\lambda_0)$ is simply a linear combination of the various $F^{(\mathbf{i})}(\mathbf{w} + \lambda_0 \mathbf{v})$ (over different $\mathbf{i}$).

We are now in a position to describe our local correction algorithm. Before describing the main algorithm for correcting from $\Omega(\delta)$-fraction errors in section 3.4.2, we informally present a simpler version of the algorithm which corrects from a much smaller fraction of errors. The analysis of this algorithm will contain many of the ideas. In the description of both algorithms, the query-efficiency will be clear, and we do not focus on how to make them run time-efficiently. In appendix 3.5, we show how the various steps of the algorithms can be made to run in a time-efficient manner as well.

3.4.1 Simplified local correction from few errors

Let C be the multiplicity code with parameters $m, d, n, q, \delta = 1 - \frac{d}{mq}$. Let $t = \binom{n+m-1}{n}$. Below we present a local corrector for the code C that corrects $\frac{\delta}{100t}$-fraction of errors.

Our input is a received word $\mathbf{y} : \mathbb{F}_q^n \to \Sigma$ and a point $\mathbf{w} \in \mathbb{F}_q^n$. We are trying to recover $F^{(<m)}(\mathbf{w})$, where $F(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ is a polynomial of degree up to d such that $\mathrm{Enc}_{m,d,n,q}(F)$ is $\delta/100t$-close to $\mathbf{y}$. With a slight abuse of notation, we write $\mathbf{y}^{(\mathbf{i})}(\mathbf{w})$ to denote the $\mathbf{i}$-th coordinate of $\mathbf{y}(\mathbf{w})$. We call the points in $\mathbb{F}_q^n$ where $\mathbf{y}$ and $\mathrm{Enc}_{m,d,n,q}(F)$ differ the "errors". The algorithm has three steps.

(1) **Pick a set V of directions:** Choose $V \subseteq \mathbb{F}_q^n$, a uniformly random subset of size t.

(2) **Recover $f(\lambda) = F(\mathbf{w}+\lambda\mathbf{v})$ for directions $\mathbf{v} \in V$:** For each $\mathbf{v} \in V$, consider the map $\ell_{\mathbf{v}} : \mathbb{F}_q \to \mathbb{F}_q^{\{0,\ldots,m-1\}}$ such that for all $\lambda_0 \in \mathbb{F}_q$ and $j \in \{0, \ldots, m-1\}$,

$$(\ell_{\mathbf{v}}(\lambda_0))(j) = \sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=j} \mathbf{y}^{(\mathbf{i})}(\mathbf{w} + \lambda_0 \mathbf{v})\mathbf{v}^{\mathbf{i}}.$$

Find the polynomial $h_{\mathbf{v}}(\lambda) \in \mathbb{F}_q[\lambda]$ of degree at most d (if any), such that $\Delta(\mathrm{Enc}_{m,d,1,q}(h_{\mathbf{v}}), \ell_{\mathbf{v}}) < \delta/2$. Observe that by lemma 3.7 order-m evaluations of degree-d polynomials on a line form a code of distance δ. Thus the polynomial $h_{\mathbf{v}}$ is

unique if it exists. In appendix 3.5 we show how to find $h_{\mathbf{v}}$ efficiently.

Assuming q is large enough compared to m, n, and $1/\delta$, one can show that with probability at least 0.9 over the choice of the set V, all $\mathbf{v} \in V$ are such that the line through $\mathbf{w}$ in direction $\mathbf{v}$ has fewer than $\delta/2$ errors on it. To see this note that the expected number of errors on a random line through $\mathbf{w}$ is at most $1 + \frac{(q-1)\delta}{100t}$. Thus by Markov's inequality a random line carries at least $\delta/2$ fraction errors with probability below $\frac{1}{50t} + O\left(\frac{1}{q}\right)$. It remains to note that lines are sampled independently.

In case all lines in V carry less than $\delta/2$ fraction of errors by formula (3.4) for all $\mathbf{v} \in V$, for all j we get

$$(\ell_{\mathbf{v}}(\lambda_0))(j) = f^{(j)}(\lambda_0),$$

for all but less than $\delta/2$ values of λ_0. Thus by lemma 3.7 we get $h_{\mathbf{v}}(\lambda) = f(\lambda)$.

(3) **Solve a linear system to recover $F^{(<s)}(\mathbf{w})$:** For each e with $0 \leq e < m$, consider the following system of equations in the variables $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ (with one equation for each $\mathbf{v} \in V$):

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} \mathbf{v}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coefficient of } \lambda^e \text{ in } h_{\mathbf{v}}(\lambda). \tag{3.5}$$

Find all $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ which satisfy at all these equations. If for some value of e the solution is not unique output FAIL; otherwise output the vector $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})<m}$.

Formula (3.3) shows that coordinates of $F^{(<m)}(\mathbf{w})$ give a solution to (3.5). We now argue that assuming q is sufficiently large compared to m and n with probability 0.9 the solution is indeed unique.

To see this observe that for each e the left hand side of (3.5) can be interpreted as an evaluation of a homogeneous n-variate polynomial of degree e with coefficients $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ at a point $\mathbf{v}$. Thus for our purposes it is sufficient to show that a random set of t points in $\mathbb{F}_q^n$ with high probability is

an interpolating set for such polynomials. Note that the total number of homogeneous polynomial of degree e is $q^{\binom{n-1+e}{n-1}}$. Furthermore, by lemma 3.5 every such non-zero polynomial can vanish at at most $\binom{eq^{n-1}}{t}$ different t-tuples of points. It remains to note that

$$q^{\binom{n-1+e}{n-1}} \cdot \binom{eq^{n-1}}{t} = o\binom{q^n}{t},$$

when q grows, since we always have $\binom{n-1+e}{n-1} < t$. Thus with overwhelming probability a random set V yields a unique solution to (3.5) for all e.
Therefore with probability at least 0.8, our algorithm outputs $F^{(\mathbf{i})}(\mathbf{w})$, as desired.

3.4.2 Local correction from $\Omega(\delta)$-fraction errors

We now come to the main local correcting algorithm and the proof of lemma 3.8. As above, to decode at a point $\mathbf{w}$, we pick several affine lines $\{\mathbf{w} + \lambda\mathbf{v} \mid \lambda \in \mathbb{F}_q\}$ through $\mathbf{w}$, and try to recover the univariate polynomial $F(\mathbf{w} + \lambda\mathbf{v})$. However, unlike the above algorithm, we do not count on the event that *all* these lines have less than $\delta/2$-fraction errors. Instead, we pick a larger number of lines than the bare-minimum required for the next step, and hope that *most* (but not necessarily all) of these lines will have fewer than $\delta/2$-fraction errors. Counting on this weakened event allows us to correct from a significantly larger fraction of errors. To compensate for the weakening, we need to make the next step of the algorithm, that of solving a linear system, more robust; we have to solve a noisy system of linear equations.

Let us elaborate on the method by which we pick the lines in the new algorithm. In the previous algorithm, we picked exactly $\binom{n+m-1}{n}$ random lines through $\mathbf{w}$ and used them to decode from $\Omega\left(\delta/\binom{n+m-1}{n}\right)$-fraction errors. By picking a larger number of lines, we can decode all the way up to $\Omega(\delta)$-fraction errors. There are several ways of picking this larger number of lines. One way is to pick $\Theta\left(\binom{n+m-1}{n}\right)$ independent and uniformly random lines through the point $\mathbf{w}$. The algorithm we present below picks these lines differently; the directions of these lines

come from a random affinely transformed grid. This way of choosing lines admits a simpler analysis.

Let C be the multiplicity code with parameters $m, d, n, q, \delta = 1 - \frac{d}{mq}$, satisfying the condition of lemma 3.8. Below we present a local corrector for C that corrects $\frac{\delta}{10}$-fraction of errors. Our input is a received word $\mathbf{y} : \mathbb{F}_q^n \to \Sigma$ and a point $\mathbf{w} \in \mathbb{F}_q^n$. We are trying to recover $F^{(<m)}(\mathbf{w})$, where $F(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ is a polynomial of degree up to d such that $\mathrm{Enc}_{m,d,n,q}(F)$ is $\delta/10$-close to $\mathbf{y}$. As before, we write $\mathbf{y}^{(\mathbf{i})}(\mathbf{w})$ to denote the $\mathbf{i}$-th coordinate of $\mathbf{y}(\mathbf{w})$. The algorithm has three steps.

(1) **Pick a set V of directions:** Pick $\mathbf{z}, \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{F}_q^n$ independently and uniformly at random. Let $M \subset \mathbb{F}_q$ be an arbitrary set of size $12(m+1)$. Let

$$V = \left\{ \mathbf{z} + \sum_{i=1}^{n} \alpha_i \mathbf{b}_i \mid \alpha_i \in M \right\}.$$

(2) **Recover $f(\lambda) = F(\mathbf{w} + \lambda \mathbf{v})$ for directions $\mathbf{v} \in V$:** For each $\mathbf{v} \in V$, consider the map $\ell_{\mathbf{v}} : \mathbb{F}_q \to \mathbb{F}_q^{\{0,\ldots,m-1\}}$ given by

$$(\ell_{\mathbf{v}}(\lambda_0))(j) = \sum_{\mathbf{i} | \mathrm{wt}(\mathbf{i}) = j} \mathbf{y}^{(\mathbf{i})}(\mathbf{w} + \lambda_0 \mathbf{v}) \mathbf{v}^{\mathbf{i}}.$$

Find the polynomial $h_{\mathbf{v}}(\lambda) \in \mathbb{F}_q[\lambda]$ of degree at most d (if any), such that $\Delta(\mathrm{Enc}_{m,d,1,q}(h_{\mathbf{v}}), \ell_{\mathbf{v}}) < \delta/2$. As above note that by lemma 3.7, $h_{\mathbf{v}}$ is unique if it exists. In appendix 3.5 we show how to find $h_{\mathbf{v}}$ efficiently.

(3) **Solve a noisy linear system to recover $F^{(<m)}(\mathbf{w})$:** For each e with $0 \le e < m$, consider the following system of linear equations in the variables $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ (with one equation for each $\mathbf{v} \in V$):

$$\sum_{\mathbf{i} | \mathrm{wt}(\mathbf{i}) = e} \mathbf{v}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coefficient of } \lambda^e \text{ in } h_{\mathbf{v}}(\lambda). \tag{3.6}$$

Find all $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ which satisfy at least $3/5$ of these equations. If the solution is not unique, output FAIL.
In appendix 3.5 we show how in certain cases the system (3.6) can be solved efficiently.

(4) Output the vector $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})<m}$.

We now proceed to analyze the above algorithm (and thus complete the proof of lemma 3.8).

Proof. [of lemma 3.8] For $n = 1$ the lemma is trivial, so we assume $n \geq 2$. Recall that we have $q > 10n$, $q > \frac{d+15m}{m}$ (so that $q > \frac{15}{\delta}$) and that $q > 12(m+1)$. We show that the above algorithm is a local corrector from a $\frac{\delta}{10}$-fraction of errors.

Fix a received word $\mathbf{y} : \mathbb{F}_q^n \to \Sigma$ and $\mathbf{w} \in \mathbb{F}_q^n$. Let $F \in \mathbb{F}[\mathbf{x}]$ be a polynomial such that $\Delta(\mathrm{Enc}_{m,d,n,q}(F), \mathbf{y}) \leq \frac{\delta}{10}$. We call the set of points where $\mathbf{y}$ and $\mathrm{Enc}_{m,d,n,q}(F)$ differ the "errors".

Step 1: Many $\mathbf{v} \in V$ are "good". If we sample a direction $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random; then the expected number of errors on the line $\{\mathbf{w} + \lambda\mathbf{v} \mid \lambda \in \mathbb{F}_q\}$ through $\mathbf{w}$ is at most

$$1 + (q-1)\delta/10 \leq 1 + q\delta/10.$$

Therefore, by Markov's inequality the probability that the number of errors on a line is $q\delta/2$ or more is at most

$$\frac{1 + q\delta/10}{q\delta/2} = \frac{2}{q\delta} + \frac{1}{5} < \frac{1}{3}.$$

Thus at least $2/3$ fraction of all possible directions $\mathbf{v}$ yield "good" lines, i.e., lines that carry less than $\delta/2$ fraction errors. In the claim below, we show that the set V samples the set of good directions well.

Claim 3.9. Let n be a positive integer. Let $M \subseteq \mathbb{F}_q$ be an arbitrary set such that $|M|^n \geq 500$. Pick $\mathbf{z}, \mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{F}_q^n$ independently and uniformly at random. Let $V = \{\mathbf{z} + \sum_{i=1}^n \alpha_i \mathbf{b}_i \mid \alpha_i \in M\}$ be a multi-set. Then for every set $G \subseteq \mathbb{F}_q^n$ of size at least $2q^n/3$, the probability that fewer than $3/5$ of the points of V lie in G is at most 0.1.

Proof. The claim follows from a standard application of Chebyshev's inequality, using the fact that the collection of random variables $\{\mathbf{z} + \sum_{i=1}^n \alpha_i \mathbf{b}_i \mid \alpha_i \in M\}$ is pairwise independent. The argument is as follows. Let $E = \mathbb{F}_q^n \setminus G$, $|E| = \tau q^n$, $\tau < 1/3$. Let $t = |M|^n$. For $\boldsymbol{\alpha} \in M^n$

consider a random variable $x^{\boldsymbol{\alpha}}$, which is the indicator variable of the event $\mathbf{z} + \sum_{i=1}^{n} \boldsymbol{\alpha}(i)\mathbf{b}_i \in E$. Clearly $\mathbb{E}\,[x^{\boldsymbol{\alpha}}] = \tau$ and $\mathbb{D}\,[x^{\boldsymbol{\alpha}}] = \tau - \tau^2$. Now consider a random variable

$$x = \sum_{\boldsymbol{\alpha} \in M^n} x^{\boldsymbol{\alpha}} = |V \cap E|,$$

Note that x is a sum of t pairwise independent variables. Thus

$$\mathbb{E}[x] = t\tau \quad \text{and} \quad \mathbb{D}[x] = t(\tau - \tau^2).$$

By Chebyshev's inequality we have

$$\Pr\left[x \geq \frac{2}{5}t\right] \leq \frac{\tau - \tau^2}{(2/5 - \tau)^2 t}.$$

It remains to note that the RHS is less than 0.1 when $t \geq 500$. □

Now recall that in the setting of lemma 3.8 we have $n \geq 2$, and so $(12(m+1))^n > 500$. Thus with probability at least 0.9 over the choice of V, 3/5-fraction of the $\mathbf{v} \in V$ are good.

Step 2: $h_{\mathbf{v}}(\lambda) = F(\mathbf{w} + \lambda\mathbf{v})$ for each good $\mathbf{v} \in V$. By equation (3.4), for each good $\mathbf{v} \in V$, the corresponding function $\ell_{\mathbf{v}}$ is such that

$$\Delta(\mathrm{Enc}_{m,d,1,q}(F(\mathbf{w} + \lambda\mathbf{v})), \ell_{\mathbf{v}}) < \delta/2.$$

Thus for each good $\mathbf{v}$, the algorithm finds $h_{\mathbf{v}}(\lambda) = F(\mathbf{w} + \lambda\mathbf{v})$. (Note that at most one polynomial $g(\lambda)$, $\deg g \leq d$ is such that $\Delta(\mathrm{Enc}_{m,d,1,q}(g), \ell_{\mathbf{v}}) < \delta/2$, since for distinct $g_1, g_2 \in \mathbb{F}_q[\lambda]$, of degree at most d, by lemma 3.7 we have $\Delta(\mathrm{Enc}_{m,d,1,q}(g_1), \mathrm{Enc}_{m,d,1,q}(g_2)) \geq \delta$.)

Step 3: $u_{\mathbf{i}} = F^{(\mathbf{i})}(\mathbf{w})$ for each $\mathbf{i}$. Since $h_{\mathbf{v}}(\lambda) = F(\mathbf{w} + \lambda\mathbf{v})$ for each good $\mathbf{v} \in V$, equation (3.3) now implies that with probability 0.9, for each $0 \leq e < m$, the values $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ with $u_{\mathbf{i}} = F^{(\mathbf{i})}(\mathbf{w})$ satisfy at least 3/5 of the equations in the system (3.6).

Finally, we observe that this solution $\{u_{\mathbf{i}}\}$ is unique with probability at least 0.9. Indeed, with probability at least 0.9 over the choice of V, the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are linearly independent (since $q \geq 10n$). In this case, there is an $\mathbb{F}_q$-affine map which gives a bijection between M^n and V. Via this affine map, we get a degree preserving correspondence between polynomials evaluated on the set M^n and polynomials evaluated on the

set V. Now by lemma 3.5 (and recalling that $|M| > 12(m+1)$), there is no nonzero polynomial of degree below m that vanishes on more that 1/5-fraction of the points of M^n. Hence no nonzero polynomial of degree below m vanishes on more that 1/5-fraction of the points of V.

Hence for each e, the collection of values $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ that satisfies 3/5 of the equations in the system (3.6) is unique; if not, then the difference $\{u_{\mathbf{i}} - u'_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=\mathbf{e}}$ of two such collections $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}, \{u'_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ gives the coefficients of a polynomial of degree below m that vanishes on at least 1/5 fraction of vectors in V; for any $\mathbf{v} \in V$ such that both $\{u_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ and $\{u'_{\mathbf{i}}\}_{\mathrm{wt}(\mathbf{i})=e}$ satisfy the equation (3.6), we have: $\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e}(u_{\mathbf{i}} - u'_{\mathbf{i}})(\mathbf{v}^{\mathbf{i}}) = 0$, contradicting the fact that there is no nonzero polynomial of degree below m that vanishes on more that 1/5-fraction of the points of V.

Thus overall, with probability at least 0.8, the algorithm will output $F^{(\mathbf{i})}(\mathbf{w})$, as desired. □

3.5 Appendix: Decoder running time

In this section, we will see how the above local correction algorithms can be made to run efficiently, in time polynomial in the query complexity.

There are two key steps which we need to elaborate on. The first step is the search for the univariate polynomial $h_{\mathbf{v}}(\lambda)$. Here the problem boils down to the problem of decoding univariate multiplicity codes up to half their minimum distance. The second step we will elaborate on is solving the noisy linear system of equations. This will reduce to an instance of decoding Reed-Muller codes.

3.5.1 Decoding univariate multiplicity codes

We deal with the first step first, that of decoding univariate multiplicity codes. Explicitly, let $\Sigma = \mathbb{F}_q^m$, we have a function $\ell : \mathbb{F}_q \to \Sigma$, and we want to find the univariate polynomial $h(\lambda) \in \mathbb{F}_q[\lambda]$ of degree at most d such that $\Delta\left(\ell, \mathrm{Enc}_{m,d,1,q}(h)\right) < (1 - d/mq)/2 = \delta/2$. Abusing notation again, we let $\ell^{(i)} : \mathbb{F}_q \to \mathbb{F}_q$ be the function which equals the i-coordinate of ℓ, for $0 \leq i < m$.

Univariate multiplicity codes are instances of "ideal error-correcting

codes" [55, 88]. We will not formally define ideal error-correcting codes here; but rather instantiate the known algorithm for decoding these codes in our setting.

Let $h(\lambda)$ be a polynomial of degree at most d such that $\Delta(\ell, \mathrm{Enc}_{m,d,1,q}(h)) < \delta/2$. Our underlying goal is to search for polynomials $e(\lambda), g(\lambda)$ such that $g(\lambda) = e(\lambda) \cdot h(\lambda)$ (and so we obtain $h(\lambda)$ as $g(\lambda)/e(\lambda)$). By the product rule for Hasse derivatives (which states that $(P_1 \cdot P_2)^{(i)}(\lambda) = \sum_{j=0}^{i} P_1^{(j)}(\lambda) P_2^{(i-j)}(\lambda)$, see appendix 3.6), such polynomials $e(\lambda)$, $g(\lambda)$ will also satisfy the equalities

$$\begin{aligned} g^{(1)}(\lambda) &= e(\lambda)h^{(1)}(\lambda) + e^{(1)}(\lambda)h(\lambda), \\ g^{(2)}(\lambda) &= e(\lambda)h^{(2)}(\lambda) + e^{(1)}(\lambda)h^{(1)}(\lambda) + e^{(2)}(\lambda)h(\lambda), \\ &\cdots \\ g^{(m-1)}(\lambda) &= \sum_{i=0}^{s-1} e^{(i)}(\lambda)h^{(m-1-i)}(\lambda) \end{aligned} \tag{3.7}$$

This motivates the following algorithm.

- Search for nonzero polynomials $e(\lambda)$, $g(\lambda)$ of degrees at most $(sq-d)/2$, $(sq+d)/2$ respectively such that for each $x \in \mathbb{F}_q$, we have the following equations:

$$\begin{aligned} g(x) &= e(x)\ell^{(0)}(x) \\ g^{(1)}(x) &= e(x)\ell^{(1)}(x) + e^{(1)}(x)\ell^{(0)}(x) \\ &\cdots \\ g^{(m-1)}(x) &= \sum_{i=0}^{m-1} e^{(i)}(x)\ell^{(m-1-i)}(x) \end{aligned}$$

  This is a collection of mq homogeneous linear equations in $(mq-d)/2+1+(mq+d)/2+1 > sq$ unknowns (the coefficients of e and g). Thus a nonzero solution $e(\lambda), g(\lambda)$ exists. Take any such nonzero solution.
- Given e, g as above, output the polynomial g/e.

To see correctness, take any h such that $\Delta\left(\ell, \mathrm{Enc}_{m,d,1,q}(h)\right) < \delta/2$. Observe that for any x where $\ell(x) = \mathrm{Enc}_{m,d,1,q}(h)(x)$, the system of

equations (3.7) is satisfied at $\lambda = x$, and hence the polynomial $g(\lambda) - e(\lambda)h(\lambda)$ has a root with multiplicity m at x. Thus

$$\sum_{x \in \mathbb{F}_q} \text{mult}(g - eh, x) > (1 - \delta/2)mq = (mq + d)/2.$$

Since $\deg(g - eh) \leq (mq + d)/2$, we conclude that the polynomial $g - eh$ must be identically 0, and hence $h(\lambda) = g(\lambda)/e(\lambda)$, as desired (here we used the fact that e, g are not both identically 0). In particular $e \mid g$, and g/e is a polynomial.

3.5.2 Solving the noisy system

We now show how in certain cases one can solve the noisy system of linear equations (3.6) efficiently. For n and m constant, this is a system of constantly many $(O(m)^n)$ linear equations over $\mathbb{F}_q$, and hence by running over all subsystems consisting of 3/5-fraction of these equations, and solving that subsystem of linear equations exactly, we can solve the noisy system in time $\exp(m^n) \cdot \text{poly} \log q$.

This is somewhat unsatisfactory. We point out some special situations where solving these noisy linear equations can be done in time $\text{poly}(m^n, \log q)$. Observe that our task is of the form: "Given a function $y : V \to \mathbb{F}_q$, find all polynomials $G(x_1, \ldots, x_n) \in \mathbb{F}_q[x_1, \ldots, x_n]$ of degree below m such that $G(\mathbf{v}) = y(\mathbf{v})$ for at least α-fraction of the $\mathbf{v} \in V$". Thus, this is a problem of noisy polynomial interpolation.

As observed in Step 3 of the analysis of the main local correction algorithm, there is a linear map which puts M^n and V in bijection. This linear map gives rise to a degree preserving correspondence between polynomials evaluated on M^n and polynomials evaluated on V. Thus, our task of doing noisy polynomial interpolation (for polynomials of degree below m) on V reduces to noisy polynomial interpolation (for polynomials of degree below m) on M^n.

For certain fields $\mathbb{F}_q$, and certain choices of the set M, there are known efficient algorithms for doing this. In particular, if q is a prime power and M is a subfield of $\mathbb{F}_q$, given a function $y : M^n \to \mathbb{F}_q$, one can recover the unique degree below m polynomial G that agrees with y in at least $(1 + m/|M|)/2$-fraction of the points of M^n in time $\text{poly}(|S|^n, \log q)$ as this turns out to be the problem of (global) decoding

of Reed Muller codes [68]. If $|M| > 12m$, then this fraction is at most $3/5$, and this suffices for application to the local correcting algorithm of the previous section.

3.6 Appendix: Hasse derivatives and multiplicities

In this section we review the basic properties of Hasse derivatives and multiplicities.

3.6.1 Properties of Hasse Derivatives

Recall that the weight of a non-negative vector $\mathbf{i}$ is the sum of its coordinates. Also, for non-negative n-dimensional vectors $\mathbf{i}$ and $\mathbf{j}$, we use the notation

$$\binom{\mathbf{i}}{\mathbf{j}} = \prod_{k=1}^{n} \binom{\mathbf{i}(k)}{\mathbf{j}(k)}.$$

The following proposition lists basic properties of the Hasse derivatives. Parts (1)-(2) below are the same as for the analytic derivative, while parts (3) and (4) are not. Part (4) considers the derivatives of the derivatives of a polynomial and shows a different relationship than is standard for the analytic derivative. However crucial for our purposes is that it shows that the $\mathbf{j}$th derivative of the $\mathbf{i}$th derivative is zero if (though not necessarily only if) the $(\mathbf{i}+\mathbf{j})$-th derivative is zero.

Proposition 3.10. Let $P(\mathbf{x}), Q(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and let $\mathbf{i}$, $\mathbf{j}$ be vectors of nonnegative integers, then

(1) $P^{(\mathbf{i})}(\mathbf{x}) + Q^{(\mathbf{i})}(\mathbf{x}) = (P+Q)^{(\mathbf{i})}(\mathbf{x})$.
(2) If P is homogeneous of degree d, then $P^{(i)}$ is homogeneous of degree $d - \mathrm{wt}(i)$.
(3) $(P \cdot Q)^{(\mathbf{i})} = \sum_{\mathbf{k}} P^{(\mathbf{k})} \cdot Q^{(\mathbf{i}-\mathbf{k})}$.
(4) $\left(P^{(\mathbf{i})}\right)^{(\mathbf{j})}(\mathbf{x}) = \binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}} P^{(\mathbf{i}+\mathbf{j})}(\mathbf{x})$.

Proof. Items 1 and 2 are easy to check. For item 3 write

$$\begin{aligned}
(P \cdot Q)(\mathbf{x}+\mathbf{z}) &= \sum_{\mathbf{c}} (P \cdot Q)^{(\mathbf{c})}(\mathbf{x})\mathbf{z}^{\mathbf{c}}, \\
P(\mathbf{x}+\mathbf{z}) &= \sum_{\mathbf{a}} P^{(\mathbf{a})}(\mathbf{x})\mathbf{z}^{\mathbf{a}}, \\
Q(\mathbf{x}+\mathbf{z}) &= \sum_{\mathbf{b}} Q^{(\mathbf{b})}(\mathbf{x})\mathbf{z}^{\mathbf{b}}, \\
(P \cdot Q)(\mathbf{x}+\mathbf{z}) &= \sum_{\mathbf{c}} \left(\sum_{\mathbf{d}} P^{(\mathbf{d})}(\mathbf{x}) Q^{(\mathbf{c}-\mathbf{d})}(\mathbf{x}) \right) \mathbf{z}^{\mathbf{c}}.
\end{aligned}$$

Combining the first and the last identity above we conclude the proof. For item 4, we expand $P(\mathbf{x}+\mathbf{z}+\mathbf{w})$ in two ways. First expand

$$\begin{aligned}
P(\mathbf{x}+(\mathbf{z}+\mathbf{w})) &= \sum_{\mathbf{k}} P^{(\mathbf{k})}(\mathbf{x})(\mathbf{z}+\mathbf{w})^{\mathbf{k}} \\
&= \sum_{\mathbf{k}} \sum_{\mathbf{i}+\mathbf{j}=\mathbf{k}} P^{(\mathbf{k})}(\mathbf{x}) \binom{\mathbf{k}}{\mathbf{i}} \mathbf{z}^{\mathbf{j}}\mathbf{w}^{\mathbf{i}} \\
&= \sum_{\mathbf{i},\mathbf{j}} P^{(\mathbf{i}+\mathbf{j})}(\mathbf{x}) \binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}} \mathbf{z}^{\mathbf{j}}\mathbf{w}^{\mathbf{i}}.
\end{aligned}$$

On the other hand, we may write

$$P((\mathbf{x}+\mathbf{z})+\mathbf{w}) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{x}+\mathbf{z})\mathbf{w}^{\mathbf{i}} = \sum_{\mathbf{i}} \sum_{\mathbf{j}} \left(P^{(\mathbf{i})}\right)^{(\mathbf{j})}(\mathbf{x})\mathbf{z}^{\mathbf{j}}\mathbf{w}^{\mathbf{i}}.$$

Comparing coefficients of $\mathbf{z}^{\mathbf{j}}\mathbf{w}^{\mathbf{i}}$ on both sides, we get the result. □

3.6.2 Properties of Multiplicities

We now translate some of the properties of the Hasse derivative into properties of the multiplicities.

Lemma 3.11. If $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and $\mathbf{a} \in \mathbb{F}^n$ are such that $\text{mult}(P, \mathbf{a}) = m$, then $\text{mult}\left(P^{(\mathbf{i})}, \mathbf{a}\right) \geq m - \text{wt}(\mathbf{i})$.

Proof. By assumption, for any $\mathbf{k}$ with $\mathrm{wt}(\mathbf{k}) < m$, we have $P^{(\mathbf{k})}(\mathbf{a}) = 0$. Now take any $\mathbf{j}$ such that $\mathrm{wt}(\mathbf{j}) < m - \mathrm{wt}(\mathbf{i})$. By item 4 of proposition 3.10, $\left(P^{(\mathbf{i})}\right)^{(\mathbf{j})}(\mathbf{a}) = \binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}} P^{(\mathbf{i}+\mathbf{j})}(\mathbf{a})$. Since $\mathrm{wt}(\mathbf{i}+\mathbf{j}) = \mathrm{wt}(\mathbf{i}) + \mathrm{wt}(\mathbf{j}) < m$, we deduce that $\left(P^{(\mathbf{i})}\right)^{(\mathbf{j})}(\mathbf{a}) = 0$. Thus $\mathrm{mult}\left(P^{(\mathbf{i})}, \mathbf{a}\right) \geq m - \mathrm{wt}(\mathbf{i})$. □

We now discuss the behavior of multiplicities under composition of polynomials. Let $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_\ell)$ be formal variables. Let $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ and $Q(\mathbf{y}) = (Q_1(\mathbf{y}), \ldots, Q_n(\mathbf{y})) \in \mathbb{F}[\mathbf{y}]^n$. We define the composition polynomial $P \circ Q(\mathbf{y}) \in \mathbb{F}[\mathbf{y}]$ to be the polynomial $P(Q_1(\mathbf{y}), \ldots, Q_n(\mathbf{y}))$. In this situation we have the following proposition.

Proposition 3.12. Let $P(\mathbf{x}), Q(\mathbf{y})$ be as above. Then for any $\mathbf{a} \in \mathbb{F}^\ell$,

$$\mathrm{mult}(P \circ Q, \mathbf{a}) \geq \mathrm{mult}(P, Q(\mathbf{a})) \cdot \min_{s \in [n]} \mathrm{mult}(Q_s - Q_s(\mathbf{a}), \mathbf{a}).$$

In particular, since for all s, $\mathrm{mult}(Q_s - Q_s(\mathbf{a}), \mathbf{a}) \geq 1$, we have

$$\mathrm{mult}(P \circ Q, \mathbf{a}) \geq \mathrm{mult}(P, Q(\mathbf{a})).$$

Proof. Let $m_1 = \mathrm{mult}(P, Q(\mathbf{a}))$ and $m_2 = \min_{s \in [n]} \mathrm{mult}(Q_s - Q_s(\mathbf{a}), \mathbf{a})$. Clearly $m_2 > 0$. If $m_1 = 0$ the result is obvious. Now assume $m_1 > 0$, so that $P(Q(\mathbf{a})) = 0$.

$$\begin{aligned} P(Q(\mathbf{a}+\mathbf{z})) &= P\left(Q(\mathbf{a}) + \sum_{\mathbf{i} \neq 0} Q^{(\mathbf{i})}(\mathbf{a}) \mathbf{z}^{\mathbf{i}}\right) \\ &= P\left(Q(\mathbf{a}) + \sum_{\mathrm{wt}(\mathbf{i}) \geq m_2} Q^{(\mathbf{i})}(\mathbf{a}) \mathbf{z}^{\mathbf{i}}\right), \end{aligned}$$

since for all $s \in [n]$, $\mathrm{mult}(Q_s - Q_s(\mathbf{a}), \mathbf{a}) = m_2 > 0$. Thus we can write

$$P(Q(\mathbf{a}+\mathbf{z})) = P\left(Q(\mathbf{a}) + h(\mathbf{z})\right),$$

where $h(\mathbf{z}) = \sum_{\mathrm{wt}(\mathbf{i}) \geq m_2} Q^{(\mathbf{i})}(\mathbf{a})\mathbf{z}^{\mathbf{i}}$. Therefore

$$\begin{aligned} P(Q(\mathbf{a}+\mathbf{z})) &= P(Q(\mathbf{a})) + \sum_{\mathbf{j}\neq 0} P^{(\mathbf{j})}\left(Q(\mathbf{a})\right) h(\mathbf{z})^{\mathbf{j}} \\ &= \sum_{\mathrm{wt}(\mathbf{j}) \geq m_1} P^{(\mathbf{j})}\left(Q(\mathbf{a})\right) h(\mathbf{z})^{\mathbf{j}}, \end{aligned}$$

since $\mathrm{mult}(P, Q(\mathbf{a})) = m_1 > 0$. Thus, since each monomial $\mathbf{z}^{\mathbf{i}}$ appearing in h has $\mathrm{wt}(\mathbf{i}) \geq m_2$, and each occurrence of $h(\mathbf{z})$ in $P(Q(\mathbf{a}+\mathbf{z}))$ is raised to the power $\mathbf{j}$, with $\mathrm{wt}(\mathbf{j}) \geq m_1$, we conclude that $P(Q(\mathbf{a}+\mathbf{z}))$ is of the form $\sum_{\mathrm{wt}(\mathbf{k}) \geq m_1 \cdot m_2} c_{\mathbf{k}}\mathbf{z}^{\mathbf{k}}$. This shows that $(P \circ Q)^{(\mathbf{k})}(\mathbf{a}) = 0$ for each $\mathbf{k}$ with $\mathrm{wt}(\mathbf{k}) < m_1 \cdot m_2$, and the result follows. □

Corollary 3.13. Let $P(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ where $\mathbf{x} = (x_1, \ldots, x_n)$. Let $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$. Let $P_{\mathbf{a},\mathbf{b}}(\lambda)$ be the polynomial $P(\mathbf{a} + \lambda \cdot \mathbf{b}) \in \mathbb{F}[\lambda]$. Then for any $\lambda_0 \in \mathbb{F}$,

$$\mathrm{mult}(P_{\mathbf{a},\mathbf{b}}, \lambda_0) \geq \mathrm{mult}(P, \mathbf{a} + \lambda_0 \cdot \mathbf{b}).$$

Proof. Let $Q(\lambda) = \mathbf{a} + \lambda\mathbf{b} \in \mathbb{F}[\lambda]^n$. Applying the previous proposition to $P(\mathbf{x})$ and $Q(\lambda)$, we get the desired claim. □

3.6.3 Schwartz-Zippel lemma with multiplicities

We are now ready to prove the multiplicity version of the Schwartz-Zippel lemma (lemma 3.5). In the basic form (without multiplicities) this lemma states that the probability that $P(\mathbf{a}) = 0$ when $\mathbf{a}$ is drawn uniformly at random from S^n is at most $d/|S|$, where P is a non-zero degree d polynomial and $S \subseteq \mathbb{F}$ is a finite set. Using $\min\{1, \mathrm{mult}(P, \mathbf{a})\}$ as the indicator variable that is 1 if $P(\mathbf{a}) = 0$, this lemma can be restated as saying $\sum_{\mathbf{a} \in S^n} \min\{1, \mathrm{mult}(P, \mathbf{a})\} \leq d \cdot |S|^{n-1}$. The version below strengthens the basic lemma by replacing $\min\{1, \mathrm{mult}(P, \mathbf{a})\}$ with $\mathrm{mult}(P, \mathbf{a})$.

Lemma 3.5. Let $P \in \mathbb{F}[\mathbf{x}]$ be a nonzero polynomial of total degree at most d. Then for any finite $S \subseteq \mathbb{F}$,

$$\sum_{\mathbf{a} \in S^n} \mathrm{mult}(P, \mathbf{a}) \leq d \cdot |S|^{n-1}.$$

Proof. We prove the lemma by induction on n. For the base case when $n = 1$, we first show that if $\mathrm{mult}(P, a) = m$ then $(x - a)^m$ divides $P(x)$. To see this, note that by definition of multiplicity, we have that $P(a + z) = \sum_i P^{(i)}(a)z^i$ and $P^{(i)}(a) = 0$ for all $i < m$. We conclude that z^m divides $P(a + z)$, and thus $(x - a)^m$ divides $P(x)$. It follows that $\sum_{a \in S} \mathrm{mult}(P, a)$ is at most the degree of P.

Now suppose $n > 1$. Let

$$P(x_1, \ldots, x_n) = \sum_{j=0}^{t} P_j(x_1, \ldots, x_{n-1}) x_n^j,$$

where $0 \leq t \leq d$, $P_t(x_1, \ldots, x_{n-1}) \neq 0$ and $\deg(P_j) \leq d - j$. For any $a_1, \ldots, a_{n-1} \in S$, let $m_{a_1,\ldots,a_{n-1}} = \mathrm{mult}(P_t, (a_1, \ldots, a_{n-1}))$. We will show that

$$\sum_{a_n \in S} \mathrm{mult}(P, (a_1, \ldots, a_n)) \leq m_{a_1,\ldots,a_{n-1}} \cdot |S| + t. \tag{3.8}$$

Given this, we may then bound

$$\sum_{a_1,\ldots,a_n \in S} \mathrm{mult}(P, (a_1, \ldots, a_n)) \leq \sum_{a_1,\ldots,a_{n-1} \in S} m_{a_1,\ldots,a_{n-1}} \cdot |S| + |S|^{n-1} \cdot t.$$

By the induction hypothesis applied to P_t, we know that

$$\sum_{a_1,\ldots,a_{n-1} \in S} m_{a_1,\ldots,a_{n-1}} \leq \deg(P_t) \cdot |S|^{n-2} \leq (d - t) \cdot |S|^{n-2}.$$

This implies the result.

We now prove the inequality (3.8). Fix $a_1, \ldots, a_{n-1} \in S$ and let $\mathbf{i} = (i_1, \ldots, i_{n-1})$ be such that $\mathrm{wt}(\mathbf{i}) = m_{a_1,\ldots,a_{n-1}}$ and $P_t^{(\mathbf{i})}(x_1, \ldots, x_{n-1})$ is nonzero. Letting $(\mathbf{i}, 0)$ denote the vector $(i_1, \ldots, i_{n-1}, 0)$, we note that

$$P^{(\mathbf{i},0)}(x_1, \ldots, x_n) = \sum_{j=0}^{t} P_j^{(\mathbf{i})}(x_1, \ldots, x_{n-1}) x_n^j,$$

and hence $P^{(\mathbf{i},0)}$ is a nonzero polynomial.

Now by lemma 3.11 and corollary 3.13, we know that

$$\begin{aligned} \mathrm{mult}(P(\mathbf{x}), \mathbf{a}) &\leq \mathrm{wt}(\mathbf{i}, 0) + \mathrm{mult}(P^{(\mathbf{i},0)}(x_1, \ldots, x_n), (a_1, \ldots, a_n)) \\ &\leq m_{a_1,\ldots,a_{n-1}} + \mathrm{mult}(P^{(\mathbf{i},0)}(a_1, \ldots, a_{n-1}, x_n), a_n). \end{aligned}$$

Summing this up over all $a_n \in S$, and applying the $n = 1$ case of this lemma to the nonzero univariate degree-t polynomial $P^{(\mathbf{i},0)}(a_1, \ldots, a_{n-1}, x_n)$, we get the inequality (3.8). This completes the proof of the lemma. □

3.7 Notes

Results presented in this chapter are due to Kopparty et al. [63]. Our review of Hasse derivatives and multiplicities in appendix 3.6 very closely follows the one in [36].

Multiplicity codes improve upon Reed Muller codes by evaluating multivariate polynomials together with their partial derivatives. The notions of partial derivatives and multiplicities have played an important role in several other works in coding theory and theoretical computer science. The "method of multiplicities" is a powerful combinatorial/algorithmic technique which has been developed and used in a number of contexts in recent years [52, 76, 54, 85, 36]. It is a method for analyzing subsets of $\mathbb{F}_q^n$ by interpolating a polynomial that vanishes at each point of that subset with high multiplicity; this often yields a strengthening of the "polynomial method", which would analyze such a subset by interpolating a polynomial that simply vanishes at each point of that subset. Rozenbloom and Tsfasman [83] consider extensions of Reed-Solomon codes where the polynomial is evaluated together with its derivatives (basically, univariate multiplicity codes) to obtain codes for some metrics generalizing the usual Hamming metric. Xing [98] considers the space of differentials on an algebraic curve to prove the existence of error-correcting codes above the Tsfasman-Vladut-Zink bound. Woodruff et al. [97] use evaluations of polynomials and their derivatives to construct private information retrieval schemes with improved communication complexity (section 7.1.2). Multiplicity codes add to this body of work, which follows the general theme that wherever polynomials and their zeroes are useful, also considering their derivatives and high-multiplicity zeroes can be even more useful.

4

Matching vector codes

In this chapter we give a detailed treatment of locally decodable codes that arise from families of matching vectors. Any construction of such codes naturally falls into two parts: the design of a matching vector family, and the actual code construction. Here we focus on the second part and defer an in-depth study of matching vector families to chapter 5.

The chapter is organized into seven sections. In section 4.1 we explain the intuition behind matching vector codes and setup the language that is used later. Our presentation follows the latest "polynomial-centric" view of MV codes that fleshes out some intrinsic similarity between MV codes and Reed Muller codes. In sections 4.2–4.4 we discuss three local decoders for matching vector codes of increasing level of sophistication. In section 4.5 we show how one can turn non-binary matching vector codes into binary. Finally, in sections 4.6 and 4.7 we summarize asymptotic parameters of MV codes and provide a detailed comparison between matching vector locally decodable codes and Reed Muller locally decodable codes.

4.1 The framework

MV codes inherit some structure from Reed Muller codes. A matching vector code consists of a linear subspace of polynomials in $\mathbb{F}_q[z_1, \ldots, z_n]$, evaluated at all points of $\mathbb{C}_m^n$, where $\mathbb{C}_m$ is a certain multiplicative subgroup of $\mathbb{F}_q^*$. The decoding algorithm is similar to traditional local decoders for RM codes given by propositions 2.4–2.5. The decoder shoots a line in a certain direction and decodes along it. The difference is that the monomials which are used are not of low-degree, they are chosen according to a matching family of vectors. Further, the lines for decoding are *multiplicative*, a notion that we define shortly. In what follows let $\mathbb{Z}_m$ denote the ring of integers modulo an integer m.

Definition 4.1. Let $S \subseteq \mathbb{Z}_m \setminus \{0\}$. We say that families $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ and $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ of vectors in $\mathbb{Z}_m^n$ form an S-matching family if the following two conditions are satisfied:

- For all $i \in [k]$, $(\mathbf{u}_i, \mathbf{v}_i) = 0$;
- For all $i, j \in [k]$ such that $i \neq j$, $(\mathbf{u}_j, \mathbf{v}_i) \in S$.

We now show how one can obtain a matching vector locally decodable code out of a matching family. We start with some notation.

- We assume that q is a prime power, m divides $q - 1$, and denote the unique subgroup of $\mathbb{F}_q^*$ of order m by $\mathbb{C}_m$;
- We fix some generator g of $\mathbb{C}_m$;
- For $\mathbf{w} \in \mathbb{Z}_m^n$, we define $g^{\mathbf{w}} \in \mathbb{C}_m^n$ by $\left(g^{\mathbf{w}(1)}, \ldots, g^{\mathbf{w}(n)}\right)$;
- For $\mathbf{w}, \mathbf{v} \in \mathbb{Z}_m^n$ we define the multiplicative line $M_{\mathbf{w},\mathbf{v}}$ through $\mathbf{w}$ in direction $\mathbf{v}$ to be the multi-set
$$M_{\mathbf{w},\mathbf{v}} = \left\{ g^{\mathbf{w}+\lambda\mathbf{v}} \mid \lambda \in \mathbb{Z}_m \right\}; \tag{4.1}$$
- For $\mathbf{u} \in \mathbb{Z}_m^n$, we define the monomial
$$\mathrm{mon}_{\mathbf{u}} \in \mathbb{F}_q[z_1, \ldots, z_n]/(z_1^m = 1, \ldots, z_n^m = 1)$$
by
$$\mathrm{mon}_{\mathbf{u}}(z_1, \ldots, z_n) = \prod_{\ell \in [n]} z_\ell^{\mathbf{u}(\ell)}. \tag{4.2}$$

Observe that for any $\mathbf{w}, \mathbf{u}, \mathbf{v} \in \mathbb{Z}_m^n$ and $\lambda \in \mathbb{Z}_m$ we have

$$\mathrm{mon}_{\mathbf{u}}\left(g^{\mathbf{w}+\lambda\mathbf{v}}\right) = g^{(\mathbf{u},\mathbf{w})}\left(g^{\lambda}\right)^{(\mathbf{u},\mathbf{v})}. \tag{4.3}$$

This suggests that if we set $y = g^{\lambda} \in \mathbb{F}_q^*$ in formula (4.3); then what we get is a univariate polynomial in y. Hence the $M_{\mathbf{w},\mathbf{v}}$-evaluation of a monomial $\mathrm{mon}_{\mathbf{u}}$ is a $\mathbb{C}_m$-evaluation of a univariate monomial

$$g^{(\mathbf{u},\mathbf{w})}y^{(\mathbf{u},\mathbf{v})} \in \mathbb{F}_q[y]. \tag{4.4}$$

This observation is the foundation of our decoding algorithms. We now specify the encoding procedure and outline the main steps taken by all decoding procedures described later on (propositions 4.2, 4.4, and 4.5). Let $\mathcal{U}, \mathcal{V}$ be an S-matching family in $\mathbb{Z}_m^n$, where $|\mathcal{U}| = |\mathcal{V}| = k$.

Encoding: We encode a message $(\mathbf{x}(1), \ldots, \mathbf{x}(k)) \in \mathbb{F}_q^k$ by the $\mathbb{C}_m^n$-evaluation of the polynomial

$$F(z_1, \ldots, z_n) = \sum_{j=1}^{k} \mathbf{x}(j) \cdot \mathrm{mon}_{\mathbf{u}_j}(z_1, \ldots, z_n). \tag{4.5}$$

Notice that $F = F_{\mathbf{x}}$ is a function of the message $\mathbf{x}$ (we will omit the subscript and treat $\mathbf{x}$ as fixed throughout this section).

Abstract decoding: The input to the decoder is a corrupted $\mathbb{C}_m^n$-evaluation of F and an index $i \in [k]$.

(1) The decoder picks $\mathbf{w} \in \mathbb{Z}_m^n$ uniformly at random;
(2) The decoder recovers the noiseless restriction of F to $M_{\mathbf{w},\mathbf{v}_i}$. To accomplish this the decoder may query the corrupted $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F at m or fewer locations.

To see that noiseless $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F uniquely determines $\mathbf{x}(i)$ note that by formulas (4.3), (4.4) and (4.5) the $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F is the $\mathbb{C}_m$-evaluation of a polynomial

$$f(y) = \sum_{j=1}^{k} \mathbf{x}(j) \cdot g^{(\mathbf{u}_j,\mathbf{w})}y^{(\mathbf{u}_j,\mathbf{v}_i)} \in \mathbb{F}_q[y]. \tag{4.6}$$

Properties of the S-matching family $\mathcal{U}, \mathcal{V}$ imply that $y^{(\mathbf{u}_j, \mathbf{v}_i)} = 1$, if $j = i$; and $y^{(\mathbf{u}_j, \mathbf{v}_i)} = y^s$, for some $s \in S$ otherwise. Formula (4.6) yields

$$f(y) = \mathbf{x}(i) \cdot g^{(\mathbf{u}_i, \mathbf{w})} + \sum_{s \in S} \left(\sum_{j \,:\, (\mathbf{u}_j, \mathbf{v}_i) = s} \mathbf{x}(j) \cdot g^{(\mathbf{u}_j, \mathbf{w})} \right) y^s. \tag{4.7}$$

For a polynomial $h \in \mathbb{F}_q[y]$ we denote by $\mathrm{supp}(h)$ the set of monomials with non zero coefficients in h, where a monomial y^e is identified with the integer e. It is evident from formula (4.7) that $\mathrm{supp}(f) \subseteq S \cup \{0\}$ and

$$\mathbf{x}(i) = f(0)/g^{(\mathbf{u}_i, \mathbf{w})}. \tag{4.8}$$

In sections 4.2–4.4 we describe several local decoders that follow the general paradigm outlined above.

4.2 Basic decoding on lines

The proposition below gives the simplest local decoder for MV codes.

Proposition 4.2. Let $\mathcal{U}, \mathcal{V}$ be a family of S-matching vectors in $\mathbb{Z}_m^n$, $|\mathcal{U}| = |\mathcal{V}| = k$, $|S| = s$. Suppose $m \mid q-1$, where q is a prime power; then there exists a q-ary linear code encoding k-long messages to m^n-long codewords that is $(s+1, \delta, (s+1)\delta)$-locally decodable for all δ.

Proof. The encoding procedure has already been specified by formula (4.5). To recover the value $\mathbf{x}(i)$

(1) The decoder picks $\mathbf{w} \in \mathbb{Z}_m^n$ at random, and queries the (corrupted) $M_{\mathbf{w}, \mathbf{v}_i}$-evaluation of F at $(s+1)$ consecutive locations $\left\{ g^{\mathbf{w} + \lambda \mathbf{v}_i} \mid \lambda \in \{0, \ldots, s\} \right\}$ to obtain values $c_0, \ldots, c_s$.
(2) The decoder recovers the unique sparse univariate polynomial $h(y) \in \mathbb{F}_q[y]$ with $\mathrm{supp}(h) \subseteq S \cup \{0\}$ such that for all $\lambda \in \{0, \ldots, s\}$, $h(g^\lambda) = c_\lambda$. (The uniqueness of $h(y)$ follows from standard properties of Vandermonde matrices. [65])
(3) Following the formula (4.8) the decoder returns $h(0)/g^{(\mathbf{u}_i, \mathbf{w})}$.

The discussion in section 4.1 implies that if all $(s+1)$ locations queried by the decoder are not corrupted then $h(y)$ is indeed the noise-

less restriction of F to $M_{\mathbf{w},\mathbf{v}_i}$, and decoder's output is correct. It remains to note that each individual query of the decoder samples a uniformly random location and apply the union bound. □

4.3 Improved decoding on lines

The local decoder for MV codes given in section 4.2 is similar to the local decoder for RM codes given in section 2.2.1 in that it can only tolerate $\delta < 1/2r$ fraction of errors. Thus the fraction of tolerable noise rapidly deteriorates with an increase in query complexity r. Below we introduce the concept of a bounded matching family of vectors and show how matching vector codes based on bounded matching families can be decoded from a nearly $1/4$ fraction of errors independent of r.

In what follows we identify $\mathbb{Z}_m$ with the subset $\{0, \ldots, m-1\}$ of real numbers. This imposes a total ordering on $\mathbb{Z}_m$, $0 < 1 < \ldots < m-1$ and allows us to compare elements of $\mathbb{Z}_m$ with reals. We say that a set $S \subseteq \mathbb{Z}_m$ is b-bounded if for all $s \in S$, $s < b$.

Definition 4.3. Let b be a positive real. An S-matching family $\mathcal{U}, \mathcal{V}$ in $\mathbb{Z}_m^n$ is b-bounded if the set S is b-bounded.

Below is the main result of this section.

Proposition 4.4. Let $0 < \sigma < 1$. Let $\mathcal{U}, \mathcal{V}$ be a σm-bounded family of S-matching vectors in $\mathbb{Z}_m^n$, $|\mathcal{U}| = |\mathcal{V}| = k$. Suppose $m \mid q-1$, where q is a prime power; then there exists a q-ary linear code encoding k-long messages to m^n-long codewords that is $(m, \delta, 2\delta/(1-\sigma))$-locally decodable for all δ.

Proof. The encoding procedure has already been specified by (4.5). To recover the value $\mathbf{x}(i)$,

(1) The decoder picks $\mathbf{w} \in \mathbb{Z}_m^n$ at random, and queries every point of the (corrupted) $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F at all m locations $\left\{g^{\mathbf{w}+\lambda\mathbf{v}_i} \mid \lambda \in \mathbb{Z}_m\right\}$ to obtain values $c_0, \ldots, c_{m-1}$.
(2) The decoder recovers the univariate polynomial $h(y) \in \mathbb{F}_q[y]$ of degree less than σm such that for all but at most $(m -$

$\sigma m)/2$ values of $\lambda \in \mathbb{Z}_m$, $h(g^\lambda) = c_\lambda$. If such an h does not exist the decoder encounters a failure, and returns 0. Note that $\deg h < \sigma m$ implies that $h(y)$ is unique, if it exists. The search for $h(y)$ can be done efficiently using the Berlekamp-Welch algorithm [68].

(3) Following the formula (4.8) the decoder returns $h(0)/g^{(\mathbf{u}_i,\mathbf{w})}$.

The discussion in section 4.1 implies that if the $M_{\mathbf{w},\mathbf{v}_i}$-evaluation of F is corrupted in at most $(m - \sigma m)/2$ locations, then $h(y)$ is indeed the noiseless restriction of F to $M_{\mathbf{w},\mathbf{v}_i}$, and the decoder's output is correct. It remains to note that each individual query of the decoder samples a uniformly random location and thus by Markov's inequality the probability that more than $(m - \sigma m)/2$ of decoder's queries go to corrupted locations is at most $2\delta/(1 - \sigma)$. □

4.4 Decoding on collections of lines

The local decoder for MV codes given in section 4.3 is similar to the local decoder for RM codes given in section 2.2.2 in that it can only tolerate $\delta < 1/4$ fraction of errors (be setting σ close to zero). Below we present an even better local decoder that tolerates a nearly $1/2$ fraction of errors, which is optimal for unique decoding.

The idea behind the improved local decoder is different from the idea behind the Reed Muller decoder of proposition 2.6. There we exploited restrictions of RM codewords to parametric degree two curves. It is however not clear how to utilize similar restrictions in the setting of matching vector codes. Instead, we randomly pick a sufficiently large (but constant) number of multiplicative lines. We assign weights to candidate values of the desired message symbol based on the number of errors along the collection of lines. We argue that the symbol with the largest weight is with high probability the correct symbol.

Proposition 4.5. Let $0 < \sigma < 1$. Let $\mathcal{U}, \mathcal{V}$ be a σm-bounded family of S-matching vectors in $\mathbb{Z}_m^n$, $|\mathcal{U}| = |\mathcal{V}| = k$. Suppose $m \mid q - 1$, where q is a prime power; then for every positive integer l there exists a q-ary linear code encoding k-long messages to m^n-long codewords that for

all $\delta < (1-\sigma)/2$ is $(lm, \delta, \exp_{\sigma,\delta}(-l))$-locally decodable.

Proof. The encoding procedure has already been specified by (4.5). We setup the notation needed to describe the decoder. Given a polynomial $h(y) \in \mathbb{F}_q[y]$, $\mathrm{supp}(h) \subseteq S \cup \{0\}$ and a multiplicative line M we denote the number of coordinates where $\mathbb{C}_m$-evaluation of h agrees with the M-evaluation of F by $\mathrm{agr}(h, M)$. For a symbol $e \in \mathbb{F}_q$ and a multiplicative line M we define

$$\mathrm{weight}(e, M) = \max_{h:h(0)=e} \mathrm{agr}(h, M),$$

where the maximum is taken over all $h(y) \in \mathbb{F}_q[y]$, $\mathrm{supp}(h) \subseteq S \cup \{0\}$. We now proceed to the local decoder. To recover the value $\mathbf{x}(i)$,

(1) The decoder picks vectors $\mathbf{w}_1, \ldots, \mathbf{w}_l \in \mathbb{Z}_m^n$ uniformly at random, and queries values of the corrupted evaluation of F along each of l multiplicative lines $\left\{M_{\mathbf{w}_j,\mathbf{v}_i}\right\}$, $j \in [l]$.
(2) For every symbol $e \in \mathbb{F}_q$ the decoder computes its weight,

$$\mathrm{weight}(e) = \sum_{j=1}^{l} \mathrm{weight}\left(e, M_{\mathbf{w}_j,\mathbf{v}_i}\right).$$

The weight measures the likelihood that $\mathbf{x}(i) = e$ given the observed values of the corrupted evaluation of F.
(3) The decoder outputs the symbol that has the largest weight. If such a symbol is not unique the decoder outputs 0.

Below we analyze the success probability of the decoder. Firstly, note that there cannot be two symbols $e_1 \neq e_2$ that both have weight above $lm(1+\sigma)/2$. Otherwise one of the multiplicative lines would give us two distinct polynomials $h_1(y), h_2(y) \in \mathbb{F}_q[y]$ of degree less than σm whose $\mathbb{C}_m$-evaluations agree in at least σm locations. Secondly, note that by Chernoff bound the probability that the total number of corrupted locations on lines $\left\{M_{\mathbf{w}_j,\mathbf{v}_i}\right\}$, $j \in [l]$ exceeds $lm(1-\sigma)/2$ is at most $\exp_{\sigma,\delta}(-l)$, provided that $\delta < (1-\sigma)/2$. □

4.5 Binary codes

In cases when query complexity r is super-constant propositions 4.2, 4.4, and 4.5 yield codes over growing alphabets. As we stated earlier our main interest is in binary codes. The next lemma extends proposition 4.5 to produce binary codes that tolerate a nearly $1/4$ fraction of errors, which is optimal for unique decoding over $\mathbb{F}_2$. The proof uses standard concatenation.

Proposition 4.6. Let $0 < \sigma < 1$. Let $\mathcal{U}, \mathcal{V}$ be a σm-bounded family of S-matching vectors in $\mathbb{Z}_m^n$, $|\mathcal{U}| = |\mathcal{V}| = k$. Suppose $m \mid q-1$, where $q = 2^b$. Further suppose that there exists a binary linear code C_{inner} of distance μB encoding b-bit messages to B-bit codewords; then for every positive integer l there exists a binary linear code C encoding kb-bit messages to $(m^n \cdot B)$-bit codewords that for all $\delta < (1-\sigma)\mu/2$ is $(lmB, \delta, \exp_{\sigma,\mu,\delta}(-l))$-locally decodable.

Proof. We define the code C to be the concatenation of the q-ary code C_{outer} used in propositions 4.2–4.5 and the binary code C_{inner}. In order to recover a single bit, the local decoder recovers the symbol of the q-ary alphabet that the bit falls into. Given a δ-corrupted concatenated evaluation of a polynomial F the decoder acts similarly to the decoder from the proposition 4.5.

We setup the notation needed to describe the decoder formally. Given a polynomial $h(y) \in \mathbb{F}_q[y]$, $\operatorname{supp}(h) \subseteq S \cup \{0\}$ and a multiplicative line M we denote the number of coordinates where C_{inner}-concatenated $\mathbb{C}_m$-evaluation of h agrees with corrupted C_{inner}-concatenated M-evaluation of F by $\operatorname{agr}(h, M)$. For a symbol $e \in \mathbb{F}_q$ and a multiplicative line M we define

$$\operatorname{weight}(e, M) = \max_{h:h(0)=e} \operatorname{agr}(h, M),$$

where the maximum is taken over all $h(y) \in \mathbb{F}_q[y]$, $\operatorname{supp}(h) \subseteq S \cup \{0\}$. To recover the i-th symbol of the outer code,

(1) The decoder picks vectors $\mathbf{w}_1, \ldots, \mathbf{w}_l \in \mathbb{Z}_m^n$ uniformly at random, and queries the coordinates corresponding to encodings of values of F along each of l lines $\left\{M_{\mathbf{w}_j, \mathbf{v}_i}\right\}$, $j \in [l]$.

(2) For every symbol $e \in \mathbb{F}_q$ the decoder computes its weight,

$$\text{weight}(e) = \sum_{j=1}^{l} \text{weight}\left(e, M_{\mathbf{w}_j,\mathbf{v}_i}\right).$$

The weight measures the likelihood that the i-the symbol of the outer code equals e given the observed values of the corrupted evaluation of F.

(3) The decoder outputs the required bit of the symbol that has the largest weight. If such a symbol is not unique the decoder outputs 0.

Below we analyze the success probability of the decoder. Firstly, note that there cannot be two symbols $e_1 \neq e_2$ that both have weight above $lmB(1-(1-\sigma)\mu/2)$. Otherwise one of the multiplicative lines would give us two distinct polynomials $h_1(y), h_2(y) \in \mathbb{F}_q[y]$ of degree less than σm whose concatenated $\mathbb{C}_m$-evaluations agree in at least $(1-(1-\sigma)\mu)mB$ locations. Secondly, note that by Chernoff bound the probability that the total number of corrupted locations on lines $\left\{M_{\mathbf{w}_j,\mathbf{v}_i}\right\}$, $j \in [l]$ exceeds $lmB(1-\sigma)\mu/2$ is at most $\exp_{\sigma,\mu,\delta}(-l)$, provided that $\delta < (1-\sigma)\mu/2$. □

4.6 Summary of parameters

Parameters of matching vector codes (propositions 4.2–4.6) are determined by parameters of the underlying family of matching vectors. In section 4.6.1 we apply proposition 4.6 to Grolmusz's family of matching vectors to obtain some explicit trade-offs between query complexity and codeword length of MV codes. In section 4.6.2 we use existing upper bounds on the size of matching vector families to establish lower bounds on the codeword length of MV codes.

4.6.1 Upper bounds

The largest currently known families of matching vectors are closely based on Grolmusz's construction of set systems with restricted intersections modulo composites (section 5.2). The following lemma captures the parameters of these families. We defer the proof to chapter 5.

Lemma 5.9. Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. Let w be a positive integer. Let $\{e_i\}$, $i \in [t]$ be integers such that for all i, we have $p_i^{e_i} > w^{1/t}$. Let $d = \max_i p_i^{e_i}$, and $h \geq w$ be arbitrary. Then there exists an $\binom{h}{w}$-sized σm-bounded family of matching vectors in $\mathbb{Z}_m^n$, where $n = \binom{h}{\leq d}$ and σ is an arbitrary real larger than $\sum_{i\in[t]} 1/p_i$.

A combination of proposition 4.6 and lemma 5.9 yields

Lemma 4.7. Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. Let w be a positive integer. Suppose integers $\{e_i\}$, $i \in [t]$ are such that for all i, we have $p_i^{e_i} > w^{1/t}$. Let $d = \max_i p_i^{e_i}$, and $h \geq w$ be arbitrary. Let σ be an arbitrary real number larger than $\sum_{i\in[t]} 1/p_i$. Suppose $m \mid q-1$, where $q = 2^b$. Further suppose that there exists a binary linear code C_{inner} of distance μB encoding b-bit messages to B-bit codewords; then for every positive integer l there exists a binary linear code C encoding $\left(\binom{h}{w} \cdot b\right)$-bit messages to $\left(m^{\binom{h}{\leq d}} \cdot B\right)$-bit codewords that for all $\delta < (1-\sigma)\mu/2$ is $(lmB, \delta, \exp_{\sigma,\mu,\delta}(-l))$-locally decodable.

In what follows we estimate asymptotic parameters of our codes.

Lemma 4.8. For all integers $t \geq 2$, $k \geq 2^t$ there exists a binary linear code encoding k-bit messages to

$$N = \exp\exp\left((\log k)^{1/t}(\log\log k)^{1-1/t} t \ln t\right)$$

-bit codewords that is $(r, \delta, \exp(-t))$-locally decodable for $r = t^{O(t)}$ and $\delta = 1/4 - O(1/\ln t)$.

Proof. The proof follows by setting parameters in lemma 4.7.

(1) By [87, theorem 5.7] there exists a universal constant c' such that the range $[(c/2)t\ln t, ct\ln t]$ contains at least t distinct odd primes $p_1, \ldots, p_t$;

(2) Note that $\sum_{i\in[t]} 1/p_i = O(1/\ln t)$;
(3) Set $m = \prod_{i\in[t]} p_i$. Clearly, $m = t^{\Theta(t)}$;
(4) Set b to be the smallest positive integer such that $m \left| 2^b - 1 \right.$ Clearly, $b \leq m - 1 = t^{O(t)}$. Set $q = 2^b$;
(5) A standard greedy argument (that is used to prove the classical Gilbert-Varshamov bound [68]) implies that there is a universal constant c' such that for all integers $s \geq 1$, there exists a binary linear code of distance $(1/2 - c'/\sqrt{s})s^2$ encoding s-bit messages to s^2-bit codewords. We set C_{inner} to be a binary linear code that encodes b-bit messages to $B = t^{\Theta(t)}$-bit codewords and has distance μB, for $\mu \geq \left(1/2 - c'/t^{\Theta(t)}\right)$;
(6) We now assume that there exists a positive integer w which is a multiple of t such that $k = w^{w/t}$. Clearly, we have $w = \Theta(t \log k/\log\log k)$;
(7) Following lemma 4.7 for every $i \in [t]$, let e_i be the smallest integer such that $p_i^{e_i} > w^{1/t}$. Let $d = \max_i p_i^{e_i}$. Clearly, $d = O(w^{1/t} t \ln t)$;
(8) Set $h = c'' \left\lceil w^{1+1/t} \right\rceil$, where c'' is an appropriately chosen (constant) positive integer ensuring that $h \geq d$;
(9) Observe that $\binom{h}{w} \cdot b \geq (h/w)^w \geq k$;
(10) Note that $\binom{h}{\leq d} \leq d(eh/d)^d$;
(11) Set $N = m^{\binom{h}{\leq d}} \cdot B \leq t^x$, where $x = O(t)(ew)^{O(w^{1/t} t \ln t)}$;
(12) Set $l = t$;
(13) We combine lemma 4.7 with inequalities that we proved above and make basic manipulations to obtain a binary linear code encoding k-bit messages to

$$\exp\exp\left((\log k)^{1/t}(\log\log k)^{1-1/t} t \ln t\right)$$

-bit codewords that is $(r, \delta, \exp(-t))$-locally decodable for $r = t^{O(t)}$ and $\delta = 1/4 - O(1/\ln t)$;
(14) Finally, we note that the assumption about $k = w^{w/t}$, for some w can be safely dropped. If k does not have the required shape, we pad k-bit messages with zeros to get messages of length k', where k' has the shape $w^{w/t}$ and then apply the

procedure above. One can easily check that such padding requires at most a quadratic blow up in the message length and therefore does not affect asymptotic parameters.

This completes the proof. □

The following theorem gives asymptotic parameters of matching vector codes in terms of the query complexity and the message length.

Theorem 4.9. For every large enough integer r and every $k \geq r$ there exists a binary linear r-query locally decodable code encoding k-bit messages to

$$\exp\exp\left((\log k)^{O(\log\log r/\log r)}(\log\log k)^{1-\Omega(\log\log r/\log r)}\log r\right) \quad (4.9)$$

bit codewords and tolerating $\delta = 1/4 - O(1/\ln\ln r)$ fraction of errors.

Proof. The proof follows by setting parameters in lemma 4.8. Set t to be the largest integer such that $t^{O(t)} \leq r$, where the constant in O-notation is the same as the one in lemma 4.8. Assuming r is sufficiently large we have $t = \Theta(\log r/\log\log r)$. One can also check that $k \geq r$ implies that the condition of lemma 4.8 is satisfied. An application of the lemma concludes the proof. □

Theorem 4.9 presents a trade-off between the query complexity and the codeword length of matching vector codes that tolerate a nearly optimal fraction of errors. In section 2.1 we mentioned that not all applications of LDCs require codes of such a high error tolerance. Specifically, applications of locally decodable codes in cryptography need short codes of constant query complexity $r = O(1)$, that tolerate some constant fraction of errors that is low significance.

It is possible to get a small saving in terms of codeword length in theorem 4.9 if one disregards the fraction of tolerable noise. Below we give two theorems that apply in that setting. The next theorem is from [59]. We omit the proof that slightly improves on what one gets by a simple combination of proposition 4.2 and lemma 5.9.

Theorem 4.10. For every integer $t \geq 2$, and for all $k \geq 2$, there exists an $r = 3 \cdot 2^{t-2}$-query linear locally decodable code over $\mathbb{F}_{2^t}$ encoding k-long messages to

$$\exp \exp_t \left((\log k)^{1/t} (\log \log k)^{1-1/t} \right)$$

-long codewords and tolerating $\delta = O(1/r)$ fraction of errors.

Theorem 4.10 yields non-binary codes. One can turn these codes into binary without an increase in the number of queries using the technique from [38, section 4]. Again we omit the proof.

Theorem 4.11. For every integer $t \geq 2$, and for all $k \geq 2$, there exists a $r = 3 \cdot 2^{t-2}$-query binary linear locally decodable code encoding k-bit messages to

$$\exp \exp_t \left((\log k)^{1/t} (\log \log k)^{1-1/t} \right)$$

-bit codewords and tolerating $\delta = O(1/r)$ fraction of errors.

Locally decodable codes given by propositions 4.2–4.6 and theorems 4.9 and 4.10 are *perfectly smooth*, i.e., on an arbitrary input each individual query of the decoder is distributed perfectly uniformly over the codeword coordinates. Binary codes given by theorem 4.11 are not perfectly smooth.

4.6.2 Lower bounds

Let $k(m, n)$ denote the size of the largest family of S-matching vectors in $\mathbb{Z}_m^n$ where we allow S to be an arbitrary subset of $\mathbb{Z}_m \setminus \{0\}$. It is easy to see that the rate of any locally decodable code obtained via propositions 4.2– 4.6 is at most $k(m, n)/m^n$.

In this section we use existing upper bounds on the size of matching vector families to establish lower bounds on the codeword length of matching vector codes. The codeword length lower bounds we get are very general. In particular they apply to *all* matching vector codes, irrespective of their query complexity.

We defer the proofs of the following upper bounds on $k(m,n)$ to chapter 5.

Theorem 5.23. Let m and n be arbitrary positive integers. Suppose p is a prime divisor of m; then

$$k(m,n) \leq \frac{5m^n}{p^{(n-1)/2}}.$$

Theorem 5.25. Let m and n be arbitrary positive integers; then

$$k(m,n) \leq m^{n-1+o_m(1)}.$$

We now translate upper bounds on matching vector families to lower bounds on the encoding length of matching vector codes. We first argue that any family of (non-binary) matching vector codes, (i.e., codes that for some m and n, encode $k(m,n)$-long messages to m^n-long codewords) has a multiplicative encoding blow-up (*stretch*) of at least $2^{\Omega(\sqrt{\log k})}$.

Theorem 4.12. Consider an infinite family of matching vector codes $C_\ell : \mathbb{F}_q^k \to \mathbb{F}_q^N$ for $\ell \in \mathbb{N}$, where $k = k(\ell)$ and $N = N(\ell)$ go to infinity with ℓ. For large enough ℓ, we have

$$N \geq k 2^{\Omega(\sqrt{\log k})}.$$

Proof. For each ℓ, we have a family of matching vectors in $\mathbb{Z}_m^n$ where m, n depend on ℓ. We have $N = m^n$ while $k \leq k(m,n)$. First assume that $n > \sqrt{\log N}$. Then by theorem 5.23 with p a prime divisor of m, we have

$$k \leq \frac{5m^n}{p^{(n-1)/2}} \leq \frac{5N}{2^{0.5\sqrt{\log N}-1/2}} \leq \frac{N}{2^{0.4\sqrt{\log N}}},$$

where the last inequality holds for large enough N, and hence for all large ℓ. Hence assume that $n \leq \sqrt{\log N}$ so that $m \geq 2^{\sqrt{\log N}}$. As ℓ

goes to infinity, N and hence m go to infinity. So for large enough ℓ, theorem 5.25 gives $k(m,n) \le m^{n-1+o_m(1)} \le m^{n-0.9}$. Hence

$$k \le \frac{m^n}{m^{0.9}} \le \frac{N}{2^{0.9\sqrt{\log N}}}.$$

Thus $k \le \frac{N}{2^{\Omega(\sqrt{\log N})}}$ for large enough ℓ. This implies that $N \ge k2^{\Omega(\sqrt{\log k})}$ for large enough ℓ. □

One can generalize theorem 4.12 to get a similar statement for binary matching vector codes, i.e., codes obtained by a concatenation of a non-binary MV code with an asymptotically good binary code.

Theorem 4.13. Let $\{m_\ell\}$ and $\{n_\ell\}$, $\ell \in \mathbb{N}$ be two arbitrary sequences of positive integers, such that ${m_\ell}^{n_\ell}$ monotonically grows to infinity. Consider an infinite family of binary codes $C_\ell : \mathbb{F}_2^{k_\ell} \to \mathbb{F}_2^{N_\ell}$ for $\ell \in \mathbb{N}$, where each code C_ℓ is obtained via a concatenation of a MV code encoding $k(m_\ell, n_\ell)$-long messages to $m_\ell^{n_\ell}$-long codewords over $\mathbb{F}_{q_\ell}$, (here $q_\ell = 2^t$ is the smallest integer such that $m_\ell \mid 2^t - 1$) with an asymptotically good binary code of some fixed rate; then for large enough ℓ the stretch of C_ℓ is at least $2^{\Omega(\sqrt{\log k_\ell})}$.

Proof. Pick a sufficiently large value of ℓ. Consider two cases

- $n_\ell \ne 1$. First observe that $k(m_\ell, n_\ell) \ge m_\ell$, e.g., take $\mathcal{U} = \{\mathbf{u}_\alpha\}_{\alpha \in \mathbb{Z}_m}$, $\mathcal{V} = \{\mathbf{v}_\beta\}_{\beta \in \mathbb{Z}_m}$, where

  $$\mathbf{u}_\alpha = (1, \alpha, 0, \ldots, 0) \quad \text{and} \quad \mathbf{v}_\beta = (\beta, -1, 0, \ldots, 0).$$

  Next note that by theorem 4.12 the stretch of the non-binary code is at least $2^{\Omega\left(\sqrt{\log k(m_\ell, n_\ell)}\right)}$, and the concatenation with a binary code can only increase the stretch. Finally note that the dimension k_ℓ of the binary code satisfies

  $$k_\ell \le k(m_\ell, n_\ell) m_\ell \le k^2(m_\ell, n_\ell).$$

  Thus $2^{\Omega\left(\sqrt{\log k(m_\ell, n_\ell)}\right)} \ge 2^{\Omega(\sqrt{\log k_\ell})}$, for an appropriately chosen constant in Ω-notation.

- $n_\ell = 1$. Set $k' = k(m_\ell, n_\ell)$. Be theorem 5.25, $k' = m_\ell^{o(1)}$. Note that $k_\ell = k't$ and $N_\ell = \Omega(m_\ell \cdot t)$, for some $t \leq m_\ell$. These conditions yield $N_\ell \geq \Omega\left(k_\ell^{3/2}\right)$.

This completes the proof. □

4.7 MV codes vs. RM codes

In this section we provide a comparison between matching vector codes and Reed Muller codes. We show that matching vector codes given by theorems 4.9, 4.11 have shorter codeword lengths than Reed Muller codes when the query complexity is low,

$$r \leq \log k/(\log\log k)^c,$$

for some constant c. We also show that all matching vector codes have longer codeword lengths than Reed Muller codes when the query complexity is high,

$$r \geq (\log k)^{c(\sqrt{\log k})},$$

for some constant c.

Recall that a Reed Muller locally decodable code (section 2.2) is specified by three integer parameters, namely, a prime power (alphabet size) q, a number of variables n, and a degree $d < q-1$. The q-ary code consists of $\mathbb{F}_q^n$-evaluations of all polynomials in $\mathbb{F}_q[z_1, \ldots, z_n]$ of total degree at most d. Such code encodes $k = \binom{n+d}{d}$-long messages to q^n-long codewords and has query complexity $r = q - 1$. If $d < \sigma(q-1)$, the code tolerates $\delta = 1/2 - \sigma$ fraction of errors. When q is a power of 2 non-binary RM LDCs can be turned into binary via concatenation. Concatenation with an asymptotically good code of relative distance μ yields an r-query binary linear code encoding k-bit messages to N-bit codewords and tolerating $\delta = (1/2 - \sigma)\mu$ fraction of errors, where

$$k = \binom{n+d}{d} \log q, \quad N = \Theta(q^n \log q), \quad r = \Theta(q \log q). \tag{4.10}$$

4.7.1 Low query complexity regime

We now argue that RM LDCs are inferior to codes of theorems 4.9, 4.11 for all $r \leq \log k/(\log\log k)^c$, where c is a universal constant. To arrive at such a conclusion we need a lower bound on the codeword length of Reed Muller locally decodable codes.

Let d, n, and q be such that formulas (4.10) yield an r-query LDC, where r belongs to the range of our interest. We necessarily have $d \leq n$ (otherwise $r > \log k$). Thus

$$k = \binom{n+d}{d} \log q \leq (en/d)^d \log q \leq n^{O(d)}, \tag{4.11}$$

and $n \geq k^{\Omega(1/d)}$. Therefore writing $\exp(x)$ to denote $2^{\Omega(x)}$, we have

$$N \geq \exp\exp\left(\log k/d\right) \geq \exp\exp\left(\log k/r\right). \tag{4.12}$$

Note that when r is a constant then already 3-query codes of theorem 4.11 improve substantially upon (4.12). To conclude the argument one needs to verify that there exists a constant c such that for every nondecreasing function $r(k)$, where $r(k)$ grows to infinity, and satisfies $r(k) \leq \log k/(\log\log k)^c$, for all sufficiently large k the right hand side of (4.12) evaluates to a larger value than (4.9).

4.7.2 High query complexity regime

Here we argue that all matching vector codes have longer codeword lengths than Reed Muller codes when $r \geq (\log k)^{c(\sqrt{\log k})}$, where c is a universal constant. Given the theorem 4.13 all we need to do is for every constant c' construct binary Reed Muller LDCs that have a stretch of less than $2^{c'\sqrt{\log k}}$ and query complexity of $(\log k)^{O\left(\sqrt{\log k}\right)}$. By formula (4.10) the rate of an RM LDC specified by parameters n and $d = q/2$ is given by

$$k/N \geq \Omega\left(\binom{n+d}{d}/q^n\right).$$

Combining this with $\binom{n+d}{d} \geq (d/n)^n$, and using $d = q/2$ we get

$$k/N \geq \Omega((1/2n)^n).$$

Thus in order to have rate above $1/2^{O(\sqrt{\log k})}$ it suffices to have

$$n = O\left(\sqrt{\log k}/\log\log k\right). \tag{4.13}$$

Given k we choose n to be the largest integer satisfying (4.13). Next we choose d to be the smallest integer satisfying $k \leq \binom{n+d}{d} \log q$. One can easily check that this yields $d = (\log k)^{O(\sqrt{\log k})}$, giving an RM LDC with desired parameters.

4.8 Notes

The study of matching vector codes has been initiated in [99] and developed further in [79, 61, 38, 59, 35, 18, 69, 19, 84]. An important progress in constructions of such codes has been accomplished in [38] where the first constructions of codes from matching vectors modulo composites (rather than primes) were considered.

Propositions 4.2, 4.4, and 4.5 are due to [38], [35, 18], and [18].

In proposition 4.2 above we interpolated the restriction polynomial $h(y)$ to recover its free coefficient. In certain cases (relying on special properties of the integer m and the set S) it may be possible to recover the free coefficient in ways that do not require complete interpolation and thus save on the number of queries. This general idea has been used in [99] under the name of "algebraic niceness", in [38] for the case of three-query codes, and in [59, 69] to obtain the shortest currently know LDCs in the regime of $r = O(1)$. Currently the quantitative improvements one gets through the use of "algebraic niceness" are relatively small. Therefore we do not go into detail on them in this book.

Recently in [84] it was shown that the query complexity of codes from proposition 4.4 can be reduced from m to $O(|S|)$.

Propositions 4.5 and 4.6 give matching vector codes that tolerate the amount of error that is nearly optimal for unique (even non-local) decoding (1/2 fraction of errors over large alphabets, 1/4 over $\mathbb{F}_2$). In [18] local decoders for MV codes are designed that correct the nearly optimal amount of noise in the list decoding model [4, 90] ($1-\epsilon$ fraction of errors over large alphabets, $1/2 - \epsilon$ over $\mathbb{F}_2$).

A slight improvement on theorems 4.10 and 4.11 has been recently obtained in [69].

The entire construction and analysis of matching vector codes (apart from the parts dealing with reduction to the binary case) work also if the underlying field, $\mathbb{F}_q$, is replaced with the complex number field $\mathbb{C}$. The only property we used in $\mathbb{F}_q$ is that it contains an element of order m, which trivially holds over $\mathbb{C}$ for every m. This implies the existence of linear locally decodable codes with essentially the same parameters as above also over the complex numbers (the definition of locally decodable codes over an arbitrary field is the same as for finite fields, we simply allow the decoder to preform field arithmetic operations on its inputs). Once one has a linear code over the complex numbers, it is straightforward to get a code over the reals by writing each complex number as a pair of real numbers. Interestingly, other than matching vector codes (and trivial 2-query codes of exponential stretch), there are no known constructions of locally decodable codes over $\mathbb{C}$ or $\mathbb{R}$. LDCs over characteristic zero have applications in arithmetic circuit complexity [37, 34, 7].

Upper bounds on $k(m, n)$ that are quoted in section 4.6.2 are from [35].

5

Matching vectors

In the previous chapter we have seen how parameters of matching vector locally decodable codes are governed by the parameters of the underlying families of matching vectors. This chapter contains a systematic study of such families.

In the first three sections after the introduction we deal with constructions. In section 5.2 we present a bounded family of matching vectors based on the Grolmusz's construction of set systems with restricted intersections modulo composites. This family underlies the main families of matching vector codes (theorems 4.9–4.11). In section 5.3 we present an elementary construction of a bounded family of matching vectors. This family improves upon the Grolmusz's family for large values of the modulus m. Finally, in section 5.4 we obtain an algebraic construction of an asymptotically optimal matching family in the narrow case of 4-dimensional vectors modulo a prime. This result has not been published previously.

In sections 5.5–5.7 we deal with upper bounds on the size of matching families. We gradually build up the necessary machinery and in section 5.7 prove theorems 5.23 and 5.25 that have been used in the previous chapter to establish lower bounds on the codeword length of

matching vector codes (theorems 4.12 and 4.13).

5.1 Introduction

Recall that two ordered sets of vectors $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ and $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ in $\mathbb{Z}_m^n$ form an S-matching family if:

- For all $i \in [k]$, $(\mathbf{u}_i, \mathbf{v}_i) = 0$;
- For all $i, j \in [k]$ such that $i \neq j$, $(\mathbf{u}_j, \mathbf{v}_i) \in S$.

In the previous chapter we argued that large S-matching families, for bounded sets S yield short locally decodable codes. Similarly, upper for $k(m, n)$, where $k(m, n)$ denotes the size of the largest family of $(\mathbb{Z}_m \setminus \{0\})$-matching vectors in $\mathbb{Z}_m^n$ yield restricted lower bounds for the codeword length of LDCs.

In the following sections we will see several constructions of matching families and bounds for $k(m, n)$. To make it easier to navigate between these bounds we suggest the reader to focus on how these bounds behave in the arguably most important regime when the modulus m is constant and the dimension n grows to infinity.

We begin with the linear algebra bound for $k(m, n)$ that applies in the case when $m = p$ is prime.

Theorem 5.1. For any positive integer n and any prime p, we have

$$k(p, n) \leq 1 + \binom{n + p - 2}{p - 1}. \tag{5.1}$$

Proof. Let $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$, $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ be a family of S-matching vectors of $\mathbb{F}_p^n$, for some $S \subseteq \mathbb{F}_p^*$. For each $i \in [k]$, we consider the polynomial

$$P_i(z_1, \ldots, z_n) = 1 - \left(\sum_{l=1}^{n} \mathbf{v}_i(l) \cdot z_l \right)^{p-1}$$

in the ring $\mathbb{F}_p[z_1, \ldots, z_n]$. It is easy to see that $P_i(\mathbf{u}_i) = 1$, whereas $P_i(\mathbf{u}_j) = 0$ for all $j \neq i$. This implies that the k polynomials $\{P_i\}_{i=1}^k$ are

linearly independent. But these polynomials all lie in an $\mathbb{F}_p$ vector-space of dimension $1 + \binom{n+p-2}{p-1}$, since they are spanned by the monomial 1 and all monomials of degree exactly $p-1$ in $z_1, \ldots, z_n$. □

For constant p the bound (5.1) simplifies to $k(p,n) \leq O(n^{p-1})$. The proof relies on the primarily of p. In the next section we will see this is inherent. In fact composite m allow for vastly larger matching families. In particular, $k(6,n)$ is *not* polynomially bounded in n.

5.2 The Grolmusz family

The construction of the matching family presented below is modeled along the lines of Grolmusz's construction of set systems with restricted intersections modulo composites.

We first show how to get a family of matching vectors that is not bounded, and then in section 5.2.1 show how to turn this family into a bounded one.

Definition 5.2. Let $S \subseteq \mathbb{Z}_m \setminus \{0\}$. We say that a set of polynomials $\mathcal{F} = \{f_1, \ldots, f_k\} \subseteq \mathbb{Z}_m[z_1, \ldots, z_h]$ and a set of points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\} \subseteq \mathbb{Z}_m^h$ form a polynomial S-matching family of size k if

- For all $i \in [k]$, $f_i(\mathbf{x}_i) = 0$;
- For all $i, j \in [k]$ such that $i \neq j$, $f_j(\mathbf{x}_i) \in S$.

Let $\mathcal{F}, \mathcal{X}$ be a k-sized polynomial matching family. For $i \in [k]$, let $\mathrm{supp}(f_i)$ denote the set of monomials in the support of the polynomial f_i. We define $\mathrm{supp}(\mathcal{F}) = \bigcup_{i=1}^{k} \mathrm{supp}(f_i)$ and $\dim(\mathcal{F}) = |\mathrm{supp}(\mathcal{F})|$. The following lemma was observed by Sudan [89].

Lemma 5.3. An k-sized polynomial S-matching family $\mathcal{F}, \mathcal{X}$ over $\mathbb{Z}_m$ yields a k-sized S-matching family $\mathcal{U}, \mathcal{V}$ in $\mathbb{Z}_m^n$, where $n = \dim(\mathcal{F})$.

Proof. Let $\mathrm{mon}_1, \ldots, \mathrm{mon}_n$ be the set of monomials in $\mathrm{supp}(\mathcal{F})$. For every $j \in [k]$ we have

$$f_j(z_1 \ldots, z_h) = \sum_{l=1}^{n} c_{jl} \mathrm{mon}_l.$$

We define the vector $\mathbf{u}_j$ to be the n-dimensional vector of coefficients of the polynomial f_j. Similarly, for $i \in [k]$, we define the vector $\mathbf{v}_i$ to be the vector of evaluations of monomials $\mathrm{mon}_1, \ldots, \mathrm{mon}_n$ at the point $\mathbf{x}_i$. It is easy to check that for all $i, j \in [k]$, $(\mathbf{u}_j, \mathbf{v}_i) = f_j(\mathbf{x}_i)$ and hence the sets $\mathcal{U}, \mathcal{V}$ indeed form an S-matching family. □

Definition 5.4 (Canonical set). Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. The *canonical set* in $\mathbb{Z}_m$ is the set of all non-zero s such that for every $i \in [t]$, $s \in \{0, 1\} \bmod p_i$.

Our goal now is to prove the following

Lemma 5.5. Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. Let w be a positive integer. Let $\{e_i\}$, $i \in [t]$ be integers such that for all i, we have $p_i^{e_i} > w^{1/t}$. Let $d = \max_i p_i^{e_i}$ and $h \geq w$ be arbitrary. Let S be the canonical set; then there exists an $\binom{h}{w}$-sized family of S-matching vectors in $\mathbb{Z}_m^n$, where $n = \binom{h}{\leq d}$.

We defer the proof of lemma 5.5 till the end of the section and now establish two preliminary facts (lemma 5.6 and corollary 5.7). Here we assume that parameters $m, t, \{p_i\}_{i\in[t]}, \{e_i\}_{i\in[t]}, w, h$, and the set S all satisfy the condition of lemma 5.5.

Lemma 5.6. For every $i \in [t]$, there is an explicit multilinear polynomial $f_i(z_1, \ldots, z_h) \in \mathbb{Z}_{p_i}[z_1, \ldots, z_h]$ where $\deg(f_i) \leq p_i^{e_i} - 1$ such that for $\mathbf{x} \in \{0, 1\}^h$, we have

$$f_i(\mathbf{x}) \equiv \begin{cases} 0 \bmod p_i, & \text{if } \sum_{l=1}^{h} \mathbf{x}(l) \equiv w \bmod p_i^{e_i}, \\ 1 \bmod p_i, & \text{otherwise.} \end{cases}$$

Proof. Our proof relies on the classical Lucas theorem [26, p. 28], stating that for all primes p and all integers

$$b = \sum_{j \geq 0} b_j \cdot p^j, \ 0 \leq b_j < p$$

$$s = \sum_{j \geq 0} s_j \cdot p^j,\ 0 \leq s_j < p,$$

we have

$$\binom{b}{s} \equiv \prod_j \binom{b_j}{s_j} \bmod p.$$

Let $\mathbf{x} \in \{0,1\}^h$ be an arbitrary vector of Hamming weight b. Let $(b_0, b_1, \ldots)$ be a p-ary expansion of b. Further, let ℓ be an arbitrary positive integer, and let S_{p^ℓ} be the h-variate multilinear symmetric polynomial of degree p^ℓ. By the Lucas theorem we have

$$S_{p^\ell}(\mathbf{x}) \ = \ \binom{b}{p^\ell} \ \equiv \ \binom{b_\ell}{1} \prod_{j \neq \ell} \binom{b_j}{0} \ \equiv \ b_\ell \bmod p.$$

To prove the lemma we need to write the function f_i as a polynomial of degree less than $p_i^{e_i}$. Observe that $f_i : \{0,1\}^h \to \{0,1\}$ is a symmetric function, i.e., its value stays the same under an arbitrary permutation of coordinates of an input vector $\mathbf{x}$. Moreover note that the value of $f_i(\mathbf{x})$ depends only on e_i least significant digits $b_0, \ldots, b_{e_i - 1}$ of the p_i-ary expansion of the Hamming weight b of $\mathbf{x}$.

Using the fact that every function from $\mathbb{Z}_{p_i}^{e_i} \to \mathbb{Z}_{p_i}$ is computed by some polynomial, f_i can be written as a polynomial $g(b_0, \ldots, b_{e_i - 1})$ over $\mathbb{Z}_{p_i}$ with the degree of each $b_j \leq p - 1$. But $S_{p^j}(\mathbf{x}) \equiv b_j \bmod p_i$. Hence the polynomial

$$g\left(S_1(\mathbf{x}), \ldots, S_{p_i^{e_i - 1}}(\mathbf{x})\right) \in \mathbb{Z}_{p_i}[z_1, \ldots, z_h]$$

computes the function f on binary inputs. It is a symmetric polynomial whose degree is bounded by $\sum_{j=0}^{e_i - 1} p_i^j (p_i - 1) = p_i^{e_i} - 1$. □

Corollary 5.7. There is an explicit multilinear polynomial $f(z_1, \ldots, z_h) \in \mathbb{Z}_m[z_1, \ldots, z_h]$ such that for all $\mathbf{x} \in \{0,1\}^h$, we have

$$f(\mathbf{x}) = \begin{cases} 0 \bmod m, & \text{if } \sum_{l=1}^h \mathbf{x}(l) = w, \\ s \bmod m, \text{ for } s \in S, & \text{if } \sum_{l=1}^h \mathbf{x}(l) < w, \end{cases}$$

where coordinates of $\mathbf{x}$ are summed as integers.

Proof. Define the polynomial f so that for all $i \in [t]$,

$$f(z_1, \ldots, z_h) \equiv f_i(z_1, \ldots, z_h) \bmod p_i.$$

Clearly, $\deg(f) \leq \max_{i \in [t]} \deg f_i$. We claim that it satisfies the above requirement. Observe that by the Chinese remainder theorem and lemma 5.6

$$f(\mathbf{x}) = 0 \bmod m \quad \text{iff} \quad \text{for all } i \in [t],\ \sum_{l=1}^{h} \mathbf{x}(l) \equiv w \bmod p_i^{e_i}.$$

The right hand side can be equivalently written as

$$\sum_{l=1}^{h} \mathbf{x}(l) \equiv w \bmod \prod_i p_i^{e_i}.$$

Note that for all $i \in [t]$, $p_i^{e_i} > w^{1/t}$. Hence $m = \prod_i p_i^{e_i} > w$. Thus whenever the integer sum $\sum_{l=1}^{h} \mathbf{x}(l) < w$, we have $\sum_{l=1}^{h} \mathbf{x}(l) \not\equiv w \bmod m$, which proves the claim. □

We are now ready to prove lemma 5.5.

Proof. [of lemma 5.5] For every $T \subseteq [h]$ of size w, define the polynomial f_T wherein the polynomial f from corollary 5.7, we set $z_j = 0$ for $j \notin T$ (but z_j stays untouched for $j \in T$). Define $\mathbf{x}_T \in \{0,1\}^h$ to be the indicator of the set T. Viewing vectors $\mathbf{x} \in \{0,1\}^h$ as indicator vectors $\mathbf{x}_L$ for sets $L \subseteq [h]$, it is easy to check that for all $T, L \in [h]$, $f_T(\mathbf{x}_L) = f(\mathbf{x}_{L \cap T})$. Combining this with Corollary 5.7 gives

- For all $T \subseteq [h]$, where $|T| = w$, $f_T(\mathbf{x}_T) = f(\mathbf{x}_T) \equiv 0 \bmod m$,
- For all $T \neq L \subseteq [h]$, where $|T| = |L| = w$, $f_T(\mathbf{x}_L) = f(\mathbf{x}_{L \cap T}) \in S \bmod m$,

where the second bullet follows from the observation that $|L \cap T| \leq w - 1$. Thus the set of polynomials $\mathcal{F} = \{f_T\}_{T \subseteq [h], |T| = w}$ and points $\mathcal{X} = \{\mathbf{x}_T\}_{T \subseteq [h], |T| = w}$ form a polynomial S-matching family.

It is clear that $k = |\mathcal{F}| = \binom{h}{w}$. To bound n, we note that $\deg(f) \leq d$ and f is multilinear. Thus we can take $\mathrm{supp}(\mathcal{F})$ to be the set of all multilinear monomials in variables $z_1, \ldots, z_h$ of degree at most d. Clearly, this yields $\dim(\mathcal{F}) = \binom{h}{\leq d}$. □

Lemma 5.5 is quite general and may be hard to parse. We now show how one can use it to establish a lower bound for $k(6,n)$. Set $t = 2, p_1 = 2, p_2 = 3$, and let w grow to infinity. Choose e_1 and e_2 to be the smallest integers satisfying $2^{e_1} \geq \sqrt{w}$ and $3^{e_2} \geq \sqrt{w}$. Clearly, $d = \max(2^{e_1}, 3^{e_2}) \leq 3\sqrt{w}$. Set $h = w^2$. By lemma 5.5 there exists an $s = \binom{w^2}{w}$-sized $\{1,3,4\}$-matching family in $\mathbb{Z}_6^n$ for $n = \binom{w^2}{\leq 3\sqrt{w}}$. Observe that

$$\log s = \Theta(w \log w), \quad \log n = \Theta(\sqrt{w} \log w), \quad \log \log n = \Theta(\log w).$$

Thus $\log s = \Theta\left(\frac{\log^2 n}{\log \log n}\right)$. Therefore in sharp contrast with theorem 5.1 we have

$$k(6,n) \geq n^{\Omega\left(\frac{\log n}{\log \log n}\right)}.$$

5.2.1 A bounded family

The following lemma shows that the canonical set can be turned into a bounded one via scaling by an invertible element. Let $\mathbb{Z}_m^*$ denote the is the set of invertible elements of $\mathbb{Z}_m$.

Lemma 5.8. Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. Let S be the canonical set in $\mathbb{Z}_m$. There exists an $\alpha \in \mathbb{Z}_m^*$ such that the set αS is σm-bounded for any $\sigma > \sum_{i \in [t]} 1/p_i$.

Proof. We start with some notation.

- For every $i \in [t]$, define the integer $\hat{p_i} = m/p_i$;
- Let $\alpha \in \mathbb{Z}_m^*$ be the unique element such that for all $i \in [t], \alpha = \hat{p_i} \bmod p_i$.

Observe that for any $i, j \in [t]$,

$$\left(\alpha^{-1}\hat{p_i}\right) \bmod p_j = \begin{cases} 1, & \text{if i=j;} \\ 0, & \text{otherwise.} \end{cases}$$

Let $s \in S$ be arbitrary. Set $I = \{i \in [t] \mid p_i \text{ does not divide } s\}$. Observe that

$$s = \alpha^{-1} \sum_{i \in I} \hat{p_i},$$

since the identity above clearly holds modulo each of the $\{p_i\}$. Therefore

$$\alpha s = \sum_{i \in I} \hat{p_i} \leq m \sum_{i \in [t]} 1/p_i.$$

This concludes the proof. □

The argument above shows that any S-matching family $\mathcal{U}, \mathcal{V}$ where S is the canonical set can be turned into a bounded one by scaling all vectors in $\mathcal{V}$ by an invertible element provided $\sum_{i \in [t]} 1/p_i < 1$. Combining lemma 5.8 with lemma 5.5 we obtain

---

Lemma 5.9. Let $m = \prod_{i=1}^{t} p_i$ be a product of distinct primes. Let w be a positive integer. Let $\{e_i\}$, $i \in [t]$ be integers such that for all i, we have $p_i^{e_i} > w^{1/t}$. Let $d = \max_i p_i^{e_i}$, and $h \geq w$ be arbitrary. Then there exists an $\binom{h}{w}$-sized σm-bounded family of matching vectors in $\mathbb{Z}_m^n$, where $n = \binom{h}{\leq d}$ and σ is an arbitrary real larger than $\sum_{i \in [t]} 1/p_i$.

The lemma above is very powerful. It underlies the constructions of locally decodable codes in chapter 4 (theorems 4.9–4.11). The lemma however yields large families of matching vectors in $\mathbb{Z}_m^n$ only when n is sufficiently large compared to m. In particular one can easily check that the lemma cannot be used to construct matching families of growing size when n is constant and m grows to infinity.

5.3 An elementary family

In this section we give an elementary construction of a bounded family of matching vectors. The construction works for both prime and composite moduli. The family improves upon the family of lemma 5.9 for large values of m. In particular it yields large families of matching vectors in $\mathbb{Z}_m^n$ when n is constant and m grows. In what follows we use $\mathbb{Z}_{\geq 0}$ to denote the set of non-negative integers.

Definition 5.10. Let $b(m', n)$ denote the number of vectors $\mathbf{w} \in \mathbb{Z}_{\geq 0}^n$ such that $\|\mathbf{w}\|_2^2 = m'$.

Thus $b(m', n)$ counts the number of integer points on the surface of the n-dimensional ball of radius $\sqrt{m'}$ in the positive orthant.

Lemma 5.11. Let $m' < m$ and $n \geq 2$ be arbitrary positive integers. There exists a $b(m', n-1)$-sized $(m'+1)$-bounded family of matching vectors in $\mathbb{Z}_m^n$.

Proof. Let $k = b(m', n-1)$ and let $\mathbf{w}_1, \ldots, \mathbf{w}_k$ be the vectors in $\mathbb{Z}_{\geq 0}^{n-1}$ such that $\|\mathbf{w}_i\|_2^2 = m'$. For each $\mathbf{w}_i$, we define vectors in $\mathbb{Z}^n$ by

$$\mathbf{u}_i = (1, -\mathbf{w}_i), \ \mathbf{v}_i = (m', \mathbf{w}_i).$$

We claim that the resulting family of vectors is a $\{1, \ldots, m'\}$-matching family. To prove this, observe that $(\mathbf{u}_i, \mathbf{v}_j) = m' - (\mathbf{w}_i, \mathbf{w}_j)$. If $i = j$, then $(\mathbf{w}_i, \mathbf{w}_j) = \|\mathbf{w}_i\|_2^2 = m'$ whereas if $i \neq j$; then by Cauchy-Schwartz

$$(\mathbf{w}_i, \mathbf{w}_j) \leq \|\mathbf{w}_i\|_2 \|\mathbf{w}_j\|_2 = m'.$$

In fact the inequality must be strict since $\mathbf{w}_i$ and $\mathbf{w}_j$ both lie on the surface of the same ball, hence they are not collinear. But since their inner product lies in $\mathbb{Z}_{\geq 0}$, we conclude that

$$(\mathbf{w}_i, \mathbf{w}_j) \in \{0, \ldots, m'-1\},$$

hence $(\mathbf{u}_i, \mathbf{v}_j) \in \{1, \ldots, m'\}$. Now note that since $m > m'$, the inner products do not change modulo m. □

The lemma below follows by combining lemma 5.11 with some crude lower bounds for $b(m', n-1)$.

Lemma 5.12. Let $m' < m$ and $n \geq 2$ be arbitrary positive integers. There exists a k-sized $(m'+1)$-bounded family of matching vectors in $\mathbb{Z}_m^n$, where

$$k = \frac{1}{m'+1}\left(\frac{m'}{n-1}\right)^{(n-1)/2} \quad \text{for } m' \geq n, \tag{5.2}$$

$$k = \binom{n-1}{m'} \quad \text{for } m' < n. \tag{5.3}$$

Proof. To prove (5.2), we set $d = \left\lfloor \sqrt{m'/(n-1)} \right\rfloor$. For every vector $\mathbf{w} \in \{0, \ldots, d\}^{n-1}$, we have $0 \leq \|\mathbf{w}\|^2 \leq (n-1)d^2 \leq m'$. By the pigeonhole principle, there exists some $m'' \in \{0, \ldots, m'\}$ such that $b(m'', n-1) \geq (d+1)^{n-1}/(m'+1)$, which by lemma 5.11 yields an $(m'+1)$-bounded matching family of size

$$k \geq \frac{1}{m'+1} \left(\left\lfloor \sqrt{\frac{m'}{n-1}} \right\rfloor + 1 \right)^{n-1} \geq \frac{1}{m'+1} \left(\frac{m'}{n-1} \right)^{(n-1)/2}.$$

Note that the condition $m' \geq n$ is only needed to ensure that the bound in meaningful.

To prove (5.3), we observe that $b(m', n-1) \geq \binom{n-1}{m'}$ by taking all vectors in $\{0,1\}^{n-1}$ of Hamming weight exactly m'. The bound follows from lemma 5.11. □

It is interesting to observe that while matching vector codes of theorem 4.9 improve upon Reed Muller locally decodable codes only when $r \leq \log k/(\log\log k)^c$, one can get MV codes that asymptotically match RM LDCs of query complexity $r = \Theta(\log k \log\log k)$ combining lemma 5.12 (where m has the shape $2^b - 1$, $n = m + 1$ and $m' = n/2$) with proposition 4.6.

5.4 An algebraic family

Below we give a previously unpublished construction of $\Omega(p^2)$ four-dimensional $\mathbb{F}_p^*$-matching vectors modulo a prime p. Later in lemma 5.20 we establish its asymptotic optimality. The technique here is different from the techniques used in sections 5.2 and 5.3. The resulting family however is not bounded, therefore it does not immediately imply locally decodable codes capable of tolerating a constant fraction of errors.

Lemma 5.13. Let p be a prime and n be a positive integer. Let π be an $\mathbb{F}_p$-hyperplane in $\mathbb{F}_{p^n}$ and let G be a multiplicative subgroup of $\mathbb{F}_{p^n}^*$. Suppose $|\pi \cap G| = d$; then there exists a $k = \lfloor |G|/d \rfloor$-sized family of $\mathbb{F}_p^*$-matching vectors in $\mathbb{F}_p^n$.

Proof. Let $\phi : \mathbb{F}_{p^n} \to \mathbb{F}_p$ be a linear map such that $\ker \phi = \pi$. Note that there exist exactly d elements $x \in G$ such that $\phi(x) = 0$. Consider a bilinear map $\Phi : \mathbb{F}_{p^n} \times \mathbb{F}_{p^n} \to \mathbb{F}_p$, such that for all y, z,

$$\Phi(y, z) = \phi(y \cdot z).$$

Note that for every $y \in G$ there exist exactly d elements $z \in G$ such that $\Phi(y, z) = 0$. Fix a basis of $\mathbb{F}_{p^n}$ over $\mathbb{F}_p$ and represent the map Φ in the coordinate form $\Phi : \mathbb{F}_p^n \times \mathbb{F}_p^n \to \mathbb{F}_p$. This yields a matrix $M \in \mathbb{F}_p^{n \times n}$ such that for every vector $\mathbf{y} \in G \subseteq \mathbb{F}_p^n$ there exist exactly d vectors $\mathbf{z} \in G$ such that

$$\mathbf{y} \cdot M \cdot \mathbf{z}^t = 0.$$

For every vector $\mathbf{g} \in G$ set $\mathbf{u_g} = \mathbf{g}$ and $\mathbf{v_g} = M \cdot \mathbf{g}^t$. Now for every vector in $\{\mathbf{u_g}\}_{\mathbf{g} \in G}$ there exist exactly d vectors in $\{\mathbf{v_h}\}_{\mathbf{h} \in G}$ such that

$$(\mathbf{u_g}, \mathbf{u_h}) = 0.$$

We apply the greedy procedure to the two families of vectors above to obtain new families $\mathcal{U}, \mathcal{V}$ where for each $\mathbf{u} \in \mathcal{U}$ there exists a unique $\mathbf{v} \in \mathcal{V}$ such that $(\mathbf{u}, \mathbf{v}) = 0$. □

Lemma 5.14. Let p be an odd prime. Then the hyperplane

$$\pi = \{x \in \mathbb{F}_{p^4} \mid x - x^p + x^{p^2} - x^{p^3} = 0\}$$

and the multiplicative group

$$G = \{x \in \mathbb{F}_{p^4}^* \mid x^{p^2+1} = 1\}.$$

share exactly two elements of $\mathbb{F}_{p^4}$, namely ± 1.

Proof. First, note that π is indeed a hyperplane. This follows from the fact that π is a kernel of a linear map, whose image is of size p, (since for every z in the image of the polynomial $x - x^p + x^{p^2} - x^{p^3}$ we have $z^p + z = 0$.) Now let $x \in \pi \cap G$. Combining $x^{p^2} = 1/x$ with the equation defining π we conclude

$$(x + 1/x) - (x + 1/x)^p = 0. \tag{5.4}$$

Thus $x + 1/x \in \mathbb{F}_p$. Therefore x satisfies a quadratic equation with coefficients in $\mathbb{F}_p$. Thus $x \in \mathbb{F}_{p^2}$, and $x^{p^2} = x$. Recall that earlier we had $x^{p^2} = 1/x$. Thus $x^2 = 1$. □

Combining lemma 5.13 and lemma 5.14 we get

Theorem 5.15. Let p be an odd prime. There exists a $(p^2+1)/2$-sized family of $\mathbb{F}_p^*$-matching vectors in $\mathbb{F}_p^4$.

5.5 Upper bounds for families modulo primes

We now turn to upper bounds on $k(m,n)$, where $k(m,n)$ denotes the size of the largest family of $(\mathbb{Z}_m \setminus \{0\})$-matching vectors in $\mathbb{Z}_m^n$. We start by bounding $k(m,n)$ in the case when $m = p$ is prime and present two bounds. The first bound has been established in section 5.1.

Theorem 5.1. For any positive integer n and any prime p, we have

$$k(p,n) \leq 1 + \binom{n+p-2}{p-1}.$$

Note that equation (5.3) shows that for constant p and growing n, the above bound is asymptotically tight.

Our second bound improves upon theorem 5.1 when p is large compared to n. We establish it by from translating the problem of constructing matching vectors into a problem about points and hyperplanes in projective space. The $n-1$ dimensional projective geometry $\mathrm{PG}(\mathbb{F}_p, n-1)$ over $\mathbb{F}_p$ consist of all points in $\mathbb{F}_p^n \setminus \{0^n\}$ under the equivalence relation $\lambda\mathbf{v} \equiv \mathbf{v}$ for $\lambda \in \mathbb{F}_p^*$. Projective hyperplanes are specified by vectors $\mathbf{u} \in \mathbb{F}_p^n \setminus \{0^n\}$ under the equivalence relation $\lambda\mathbf{u} \equiv \mathbf{u}$ for $\lambda \in \mathbb{F}_p^*$; such a hyperplane contains all points $\mathbf{v}$ where $(\mathbf{u},\mathbf{v}) = 0$.

We define a bipartite graph $H(U,V)$ where the vertices on the left correspond to all hyperplanes in $\mathrm{PG}(\mathbb{F}_p, n-1)$, vertices on the right correspond to all points in $\mathrm{PG}(\mathbb{F}_p, n-1)$ and $\mathbf{u}$ and $\mathbf{v}$ are adjacent if $(\mathbf{u},\mathbf{v}) = 0$. For $X \subseteq U$ and $Y \subseteq V$, we define $N(X)$ and $N(Y)$ to be their neighborhoods. We use $N(\mathbf{u})$ for the neighborhood of $\mathbf{u}$.

Definition 5.16. Let n be a positive integer and p be a prime. Let U be the set of hyperplanes in $\mathrm{PG}(\mathbb{F}_p, n-1)$. We say that a set $X \subseteq U$ satisfies the *unique neighbor property* if for every $\mathbf{u} \in X$, there exists $\mathbf{v} \in N(\mathbf{u})$ such that $\mathbf{v}$ is not adjacent to $\mathbf{u}'$ for any $\mathbf{u}' \in X \setminus \{\mathbf{u}\}$.

Lemma 5.17. Let n be a positive integer and p be a prime. Let U be the set of hyperplanes in $\mathrm{PG}(\mathbb{F}_p, n-1)$. There exists a set $X \subseteq U$, $|X| = k$ satisfying the unique neighbor property if and only if there exists a k-sized family of $\mathbb{Z}_p^*$-matching vectors in $\mathbb{Z}_p^n$.

Proof. Assume that $X = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ satisfies the unique neighbor property. Let $Y = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ be such that $\mathbf{v}_i$ is a unique neighbor of $\mathbf{u}_i$. This implies that $(\mathbf{u}_i, \mathbf{v}_i) = 0$ and $(\mathbf{u}_j, \mathbf{v}_i) \neq 0$ for $i \neq j$. Thus X, Y gives a $\mathbb{Z}_p^*$-matching vector family in $\mathbb{Z}_p^n$.

For the converse, let us start with a k-sized matching vector family $\mathcal{U}, \mathcal{V}$ in $\mathbb{Z}_p^n$. In case $k = 1$ the lemma holds trivially. We claim that if $k \geq 2$ then $\mathbf{u} \in \mathcal{U}$ implies that $\lambda\mathbf{u} \notin \mathcal{U}$ for any $\lambda \in \mathbb{F}_p^* \setminus \{1\}$. This is true since $(\mathbf{u}, \mathbf{v}) = 0$ implies $(\lambda\mathbf{u}, \mathbf{v}) = 0$, which would violate the definition of a matching vector family. Thus we can associate each $\mathbf{u} \in \mathcal{U}$ with a distinct hyperplane in $\mathrm{PG}(\mathbb{F}_p, n-1)$. Similarly, we can associate every $\mathbf{v} \in \mathcal{V}$ with a distinct point in $\mathrm{PG}(\mathbb{F}_p, n-1)$. It is easy to see that $\mathbf{v}_i$ is a unique neighbor of $\mathbf{u}_i$, hence the set $\mathcal{U}$ satisfies the unique neighbor property. □

Corollary 5.18. Let n be a positive integer and p be a prime. Let U be the set of hyperplanes in $\mathrm{PG}(\mathbb{F}_p, n-1)$. The size of the largest set $X \subseteq U$ that satisfies the unique neighbor property is exactly $k(p, n)$.

The expansion of the graph $H(U, V)$ was analyzed by Alon using spectral methods [1, theorem 2.3]. We use the rapid expansion of this graph to bound the size of the largest matching vector family.

Lemma 5.19. Let $n \geq 2$ be an integer and p be a prime. Let U (V) be the set of hyperplanes (points) in $\mathrm{PG}(\mathbb{F}_p, n-1)$. Let $u = \frac{p^n-1}{p-1} = |U| = |V|$. For any nonempty set $X \subseteq U$ with $|X| = x$,

$$|N(X)| \geq u - u^{\frac{n}{n-1}}/x. \tag{5.5}$$

Lemma 5.20. Let n be a positive integer and p be a prime; then

$$k(p,n) \leq 4p^{n/2} + 2. \tag{5.6}$$

Proof. If $n = 1$, inequality (5.6) holds trivially. We assume $n \geq 2$. Let $\mathcal{U} \subseteq U$, $\mathcal{V} \subseteq V$ be a matching family of size $k(p,n)$. Pick $X \subseteq \mathcal{U}$ of size $x > 0$. By formula (5.5),

$$|N(X)| \geq u - u^{\frac{n}{n-1}}/x.$$

Since every point in $\mathcal{U} \setminus X$ must contain a unique neighbor from the set $V \setminus N(X)$, we have

$$|\mathcal{U} \setminus X| \leq |V \setminus N(X)| \leq \frac{u^{\frac{n}{n-1}}}{x} \;\Rightarrow\; |\mathcal{U}| \leq \frac{u^{\frac{n}{n-1}}}{x} + x. \tag{5.7}$$

Note that the inequality in the right hand side of (5.7) holds for all positive integers x. Picking $x = \left\lceil u^{\frac{n}{2(n-1)}} \right\rceil$ gives

$$\begin{aligned} |\mathcal{U}| &\leq 2\left\lceil u^{\frac{n}{2(n-1)}} \right\rceil \\ &\leq 2\left(\frac{p^n}{p-1}\right)^{\frac{n}{2(n-1)}} + 2 \\ &= 2\left(\frac{p}{p-1}\right)^{\frac{n}{2(n-1)}} p^{n/2} + 2 \\ &\leq 4p^{n/2} + 2, \end{aligned}$$

where the last inequality is a simple calculation. □

Equation (5.2) shows that for $n = O(1)$ we have $k(p,n) = \Omega\left(p^{(n-3)/2}\right)$. Thus lemma 5.20 is nearly tight when n is a constant and p grows to infinity. Note that for this setting of parameters, the linear-algebra bound (5.1) only gives $k(p,n) \leq O(p^{n-1})$.

5.6 Upper bounds for families modulo prime powers

Bounds for matching families modulo prime powers are obtained via a reduction to the prime case.

Lemma 5.21. Let n be a positive integer, p be a prime and $e \geq 2$. We have

$$k(p^e, n) \leq p^{(e-1)n} k(p, n+1).$$

Proof. Assume for contradiction that we have a matching family $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}, \mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ of size $k > p^{(e-1)n} k(p, n+1)$ in $\mathbb{Z}_{p^e}^n$. For every $i \in [k]$, write $\mathbf{u}_i = \mathbf{u}_i' + p^{e-1}\mathbf{u}_i''$ where $\mathbf{u}_i' \in \mathbb{Z}_{p^{e-1}}^n$ and $\mathbf{u}_i'' \in \mathbb{Z}_p^n$. By the pigeonhole principle, there are $k' > k(p, n+1)$ values of i which give the same vector $\mathbf{u}_i' \in \mathbb{Z}_{p^{e-1}}^n$, assume for convenience that the corresponding vectors in $\mathcal{U}$ are $\mathbf{u}_1, \ldots, \mathbf{u}_{k'}$ with matching vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{k'}$. We will use these vectors to construct a matching vector family of size $k' > k(p, n+1)$ in $\mathbb{Z}_p^{n+1}$, which gives a contradiction.

For each $i \in [k']$, we extend $\mathbf{u}_i''$ to a vector $\bar{\mathbf{u}}_i$ by appending 1 in the last coordinate. For every $i \in [k']$, write $\mathbf{v}_i = \mathbf{v}_i' + p\mathbf{v}_i''$ where $\mathbf{v}_i' \in \mathbb{Z}_p^n$ and $\mathbf{v}_i'' \in \mathbb{Z}_{p^{e-1}}^n$. We extend $\mathbf{v}_i'$ to a vector $\bar{\mathbf{v}}_i$ by appending $(\mathbf{u}_i', \mathbf{v}_i)/p^{e-1} \in \mathbb{Z}_p$ in the last coordinate (we will show that this ratio is in fact integral).

We claim that for all $i \in [k']$, $(\bar{\mathbf{u}}_i, \bar{\mathbf{v}}_i) = 0 \bmod p$. To see this, observe that

$$(\bar{\mathbf{u}}_i, \bar{\mathbf{v}}_i) = (\mathbf{u}_i'', \mathbf{v}_i') + (\mathbf{u}_i', \mathbf{v}_i)/p^{e-1}. \tag{5.8}$$

But we have

$$\begin{aligned} (\mathbf{u}_i, \mathbf{v}_i) &= (\mathbf{u}_i', \mathbf{v}_i) + p^{e-1}(\mathbf{u}_i'', \mathbf{v}_i) \\ &\equiv (\mathbf{u}_i', \mathbf{v}_i) + p^{e-1}(\mathbf{u}_i'', \mathbf{v}_i') \\ &= 0 \bmod p^e. \end{aligned}$$

From this we conclude that $(\mathbf{u}_i', \mathbf{v}_i) \equiv 0 \bmod p^{e-1}$, and that $(\mathbf{u}_i'', \mathbf{v}_i') + (\mathbf{u}_i', \mathbf{v}_i)/p^{e-1} = 0 \bmod p$. From equation (5.8), we conclude that $(\bar{\mathbf{u}}_i, \bar{\mathbf{v}}_i) = 0 \bmod p$. Next we claim that $(\bar{\mathbf{u}}_j, \bar{\mathbf{v}}_i) \neq 0 \bmod p$ for $i \neq j \in [k']$. We have

$$(\bar{\mathbf{u}}_j, \bar{\mathbf{v}}_i) = (\mathbf{u}_j'', \mathbf{v}_i') + (\mathbf{u}_i', \mathbf{v}_i)/p^{e-1} \tag{5.9}$$

But, since $\mathbf{u}'_i = \mathbf{u}'_j$, we also have

$$\begin{aligned}(\mathbf{u}_j, \mathbf{v}_i) &= (\mathbf{u}'_j, \mathbf{v}_i) + p^{e-1}(\mathbf{u}''_j, \mathbf{v}_i) \\ &\equiv (\mathbf{u}'_i, \mathbf{v}_i) + p^{e-1}(\mathbf{u}''_j, \mathbf{v}'_i) \\ &\not\equiv 0 \bmod p^e,\end{aligned}$$

which implies that $(\mathbf{u}''_j, \mathbf{v}'_i) + (\mathbf{u}'_i, \mathbf{v}_i)/p^{e-1} \not\equiv 0 \bmod p$. This shows that the vectors $\{\bar{\mathbf{u}}_j\}_{j=1}^{k'}$, $\{\bar{\mathbf{v}}_i\}_{i=1}^{k'}$ give a matching vector family of size $k' > k(p, n+1)$, which is a contradiction. □

5.7 Upper bounds for families modulo composites

Bounds for matching families modulo composites are obtained via reductions to the prime power case.

Lemma 5.22. Let m, n, and q be arbitrary positive integers such that $q|m$ and $(q, m/q) = 1$; then

$$k(m, n) \leq (m/q)^n \, k(q, n).$$

Proof. Let us write $m/q = r$. Let $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$, $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ be a family of S-matching vectors of $\mathbb{Z}_m^n$, for some $S \subseteq \mathbb{Z}_m \setminus \{0\}$. For each vector $\mathbf{u} \in \mathbb{Z}_m^n$ we can define the vectors $\mathbf{u}' \equiv \mathbf{u} \bmod q \in \mathbb{Z}_q^n$ and $\mathbf{u}'' \equiv \mathbf{u} \bmod r \in \mathbb{Z}_r^n$. From the definition of a matching vector family, we have that

- For all $i \in [k]$, $(\mathbf{u}'_i, \mathbf{v}'_i) = 0$ and $(\mathbf{u}''_i, \mathbf{v}''_i) = 0$;
- For all $i, j \in [k]$ such that $i \neq j$, $(\mathbf{u}'_j, \mathbf{v}'_i) \neq 0$ or $(\mathbf{u}''_j, \mathbf{v}''_i) \neq 0$.

Assume $k > (m/q)^n \, k(q, n)$. By the pigeonhole principle, there exists a vector $\mathbf{u} \in \mathbb{Z}_r^n$ such that $\mathbf{u}''_j = \mathbf{u}$ holds for $k' > k(q, n)$ values of $j \in [k]$. Let us assume that these values are $1, \ldots, k'$. Note that for any $i, j \in [k']$ we have $(\mathbf{u}''_j, \mathbf{v}''_i) = (\mathbf{u}''_i, \mathbf{v}''_i) = 0$. Hence, by the definition of a matching family, we must have

- For all $i \in [k']$, $(\mathbf{u}'_i, \mathbf{v}'_i) = 0$;
- For all $i, j \in [k']$ such that $i \neq j$, $(\mathbf{u}'_j, \mathbf{v}'_i) \neq 0$.

Thus vectors $\{\mathbf{u}'_1, \ldots, \mathbf{u}'_{k'}\}$ and $\{\mathbf{v}'_1, \ldots, \mathbf{v}'_{k'}\}$ form a matching family mod q of size larger than $k(q, n)$ which gives a contradiction. □

Theorem 5.23. Let m and n be arbitrary positive integers. Suppose p is a prime divisor of m; then

$$k(m, n) \leq 5\frac{m^n}{p^{(n-1)/2}}.$$

Proof. Let p^e be the largest power of p which divides m. By lemmas 5.22, 5.21 and 5.20, we get

$$k(m, n) \leq \left(\frac{m}{p^e}\right)^n p^{(e-1)n}\left(4p^{(n+1)/2} + 2\right) \leq 5\frac{m^n}{p^{(n-1)/2}}$$

This concludes the proof. □

The above bound is weak when n and p are constants, for instance it is meaningless for $n = 1$. We give another bound below which handles the case of small n. We start with the case when $n = 1$.

Lemma 5.24. Let $m \geq 2$ be an arbitrary positive integer; then

$$k(m, 1) \leq m^{O(1/\log\log m)} = m^{o_m(1)}.$$

Proof. Let $\mathcal{U} = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}, \mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ be a family of $\mathbb{Z}_m \setminus \{0\}$-matching vectors in $\mathbb{Z}_m^1$. We treat every vector $\mathbf{u} \in \mathcal{U}$ as an integer and observe that for any $i \neq j \in [k]$, $\gcd(\mathbf{u}_i, m) \neq \gcd(\mathbf{u}_j, m)$. (Otherwise $(\mathbf{u}_i, \mathbf{v}_i) = 0$ would yield $(\mathbf{u}_j, \mathbf{v}_i) = 0$.) An application of a standard bound [56], saying the number of distinct divisors of m is at most $m^{O(1/\log\log m)}$ concludes the proof. □

We now proceed to the case of general n.

Theorem 5.25. Let m and n be arbitrary positive integers; then

$$k(m,n) \leq m^{n-1+o_m(1)}.$$

Proof. Let $\mathcal{U} = \{\mathbf{u}_j\}_{j\in[k]}, \mathcal{V} = \{\mathbf{v}_i\}_{i\in[k]}$ be a family of $\mathbb{Z}_m \setminus \{0\}$-matching vectors in $\mathbb{Z}_m^n$. Given a vector $\mathbf{u} \in \mathbb{Z}_m^n$, we define the $\mathbb{Z}_m$-orbit of $\mathbf{u}$ to be the set of all vectors that can be written as $\lambda\mathbf{u}$ for $\lambda \in \mathbb{Z}_m$. The orbits are not disjoint. We claim that all of $\mathbb{Z}_m^n$ can be covered by no more than $m^n/\phi(m)$ orbits, where $\phi(\cdot)$ denotes the Euler function. Furthermore, we claim that each orbit can contribute at most $k(m,1)$ vectors to $\mathcal{U}$.

Let $U \subseteq \mathbb{Z}_m^n$ denote the set of all vectors $\mathbf{u}$ such that the GCD of all coordinates of $\mathbf{u}$ is in $\mathbb{Z}_m^*$. Any vector $\mathbf{u}' \in \mathbb{Z}_m^n$ can be written as $\lambda\mathbf{u}$ for $\mathbf{u} \in U$ and $\lambda \in \mathbb{Z}_m$. Thus the orbits of vectors in U cover all of $\mathbb{Z}_m^n$. For $\mathbf{u}', \mathbf{u}'' \in U$, we say that $\mathbf{u}' \equiv \mathbf{u}''$ if $\mathbf{u}''$ lies in the $\mathbb{Z}_m$ orbit of $\mathbf{u}'$. It is easy to see that this is indeed an equivalence relation on U, which divides U into equivalence classes of size $\phi(m)$. Here the equivalence class of a vector $\mathbf{u}$ is $\mathbb{Z}_m^* \cdot \mathbf{u}$. Thus if we pick $U' \subseteq U$ which contains a single representative of each equivalence class, then the orbits of U' contain all of $\mathbb{Z}_m^n$. Thus we have $|U'| = \frac{|U|}{\phi(m)} \leq \frac{m^n}{\phi(m)}$.

Now consider the orbit of a vector $\mathbf{u} \in \mathcal{U}$. Assume that it contributes the vectors $\mathbf{u}_1 = \lambda_1\mathbf{u}, \ldots, \mathbf{u}_t = \lambda_t\mathbf{u}$ to $\mathcal{U}$ where $\lambda_i \in \mathbb{Z}_m$. Assume that the matching vectors in $\mathcal{V}$ are $\mathbf{v}_1, \ldots, \mathbf{v}_t$. Then it is easy to see that $\mathcal{U}' = \{\lambda_1, \ldots, \lambda_t\}$ and $\mathcal{V}' = \{(\mathbf{u}, \mathbf{v}_1), \ldots, (\mathbf{u}, \mathbf{v}_t)\}$ are a matching vector family in one dimension, so that $t \leq k(m,1)$. Thus we conclude that

$$k(m,n) \leq \frac{m^n}{\phi(m)} k(m,1) \leq m^{n-1+o_m(1)}$$

using a standard lower bound [56], $\phi(m) \geq \Omega(m/\log\log m)$ and lemma 5.24. □

5.8 Notes

Grolmusz's original construction of set systems with restricted intersections modulo composites is given in [50, 51]. An important ingredient

to this construction is the low-degree representation of the OR-function from [10].

Our proof of lemma 5.6 follows [49, theorem 2.16]. Lemma 5.8 is from [35, 18]. The construction in section 5.3 and upper bounds in sections 5.5–5.7 are from [35].

In combinatorics there is a body of work on bounding the size of set systems with restricted modular intersections. This is related to our study of the maximal cardinality of a $(\mathbb{Z}_m \setminus \{0\})$-matching family of vectors in $\mathbb{Z}_m^n$. The precise problem there is to bound the size of the largest set family $\mathcal{F}$ on $[n]$, where the sets in $\mathcal{F}$ have cardinality 0 modulo some integer m, while their intersections have non-zero cardinality modulo m. The classical result in this area shows that when m is a prime power an upper bound of $n^{O(m)}$ holds [6]. No such bound applies when m is composite [50]. The best bound for general m is $|\mathcal{F}| \leq 2^{n/2}$ [86].

6

Lower bounds

In this chapter we review existing lower bounds for the codeword length of general locally decodable codes. The proofs of these bounds fit the following high level strategy. Firstly, one converts a locally decodable code into a normal form where the decoder is restricted to operate by outputting a modulo 2 sum of some r codeword coordinates coming from a family of disjoint r-tuples. Secondly, one argues that any code presented in a normal form requires a large codeword length.

In section 6.1 we deal with the conversion to the normal form. In section 6.2 we establish polynomial lower bounds for the codeword length of r-query codes for general r. The bound rapidly deteriorates as r increases. In section 6.3 we deal with the narrow case of 2-query codes and establish tight exponential lower bounds. Throughout the chapter we restrict our attention to binary codes of constant query complexity.

6.1 Preliminaries

General locally decodable codes can be quite complex. Decoders may invoke complicated adaptive procedures to decide which codeword bits

to query. They may also perform arbitrary computation to come up with the output. In order to prove lower bounds for locally decodable codes it is convenient to first turn them into the following normal form.

Definition 6.1. A binary code $C : \mathbb{F}_2^k \to \mathbb{F}_2^N$ is said to be (r, η, β)-normally decodable if for each $i \in [k]$ there a collection M_i of $\eta \cdot N$ disjoint tuples of exactly r indices from $[N]$ such that for every $t \in M_i$ the following holds

$$\Pr_{\mathbf{x} \in \mathbb{F}_2^n} \left[\mathbf{x}_i = \sum_{j \in t} C(\mathbf{x})_j \right] \geq \frac{1}{2} + \beta, \tag{6.1}$$

where the probability is taken uniformly over $\mathbf{x}$.

Hence to decode $\mathbf{x}_i$ from $C(\mathbf{x})$, the decoder can just add up the indices in a randomly chosen tuple t from M_i. Note that normally decodable codes are somewhat weaker objects than usual locally decodable codes. Specifically, normally decodable codes only provide an "average-case" guarantee of correct decoding. Our main goal in this section is to prove the following lemma.

Lemma 6.2. Suppose there exists a (r, δ, ϵ)-locally decodable code encoding k-bit messages to N-bit codewords where $\epsilon < 1/2$; then there exists a (r, η, β)-normally decodable code encoding k-bit messages to $O(N)$-bit codewords where

$$\eta \geq \frac{(1/2 - \epsilon)\delta}{3 \cdot r^2 2^{r-1}} \quad \text{and} \quad \beta \geq \frac{1/2 - \epsilon}{2^{2r}}.$$

Proof. Our proof proceeds in four steps. On the first step we turn a potentially adaptive decoder of the code C into a non-adaptive one, i.e., a one that makes all queries to the codeword simultaneously. On the second step we turn the locally decodable code into a smooth one, i.e., a one where no codeword coordinate in queried too often. On the third step, we ensure that r-tuples of coordinates that may be read by the decoder interested in the i-th message bit are all disjoint. Finally on

the fourth step, we ensure that the decoder always returns a modulo 2 sum of the accessed codeword coordinates. On each step we incur a certain loss in code parameters. Let $\alpha = 1/2 - \epsilon$ be the advantage of the local decoder of the code C over random guessing.

Step 1: (*Non-adaptivity.*) Let $\mathcal{A}$ be the potentially adaptive local decoder for C. We now construct a non-adaptive local decoder $\mathcal{A}'$ for the same code C at a price of reducing the value of α to $\alpha/2^{r-1}$. The decoder $\mathcal{A}'$ guesses the values of the first $r-1$ coordinates that may be accessed by $\mathcal{A}$, and submits the set of queries based upon this guess. If $\mathcal{A}'$ guesses correctly the decoding procedure works with probability $1/2+\alpha$; otherwise, $\mathcal{A}'$ returns a random bit which is correct with probability $1/2$.

Step 2: (*Smoothness.*) We now adjust the nonadaptive r-query decoding procedure $\mathcal{A}$ that we got from the previous step to obtain a new decoding procedure $\mathcal{A}'$ such that for all $\mathbf{x} \in \mathbb{F}_2^k$ and $i \in [k]$, we have

$$\Pr\left[\mathcal{A}'(C(\mathbf{x}), i) = \mathbf{x}_i\right] \geq 1/2 + \alpha/2^{r-1}, \tag{6.2}$$

and for every $i \in [k]$ and $j \in [N]$,

$$\Pr\left[\mathcal{A}'(\cdot, i) \ \text{ reads index } j\right] \leq r/\delta N. \tag{6.3}$$

For every $i \in [k]$, let $S_i \subseteq [N]$ denote the set of codeword coordinates that are accessed by $\mathcal{A}$ on an input i with probability above $r/\delta N$. Since $\mathcal{A}$ reads at most r indices in every invocation, for every $i \in [k]$, we have $|S_i| \leq \delta \cdot N$. We define the new decoder $\mathcal{A}'$ as follows: $\mathcal{A}'(\cdot, i)$ runs $\mathcal{A}(\cdot, i)$ in a black-box manner by reading indices from the codeword, and returning their values to $\mathcal{A}$. The only exception is that if $\mathcal{A}$ requests an index in S_i, $\mathcal{A}'$ does not read that index, but instead simply returns 0 to $\mathcal{A}$. Thus the output of $\mathcal{A}'$ on $C(\mathbf{x})$ is the same as the output of $\mathcal{A}$ on a certain string $\mathbf{y}$ such that $\Delta(C(\mathbf{x}), \mathbf{y}) \leq \delta$. It remains to note that given access to any such string $\mathcal{A}$ outputs $\mathbf{x}_i$ with probability at least $1/2 + \alpha/2^{r-1}$.

Step 3: (*Disjointness.*) We now modify the decoding procedure $\mathcal{A}$ that we got from the previous step to ensure that for every i, the tuples of coordinates that may be read by the decoder interested in the i-the message bit are all disjoint. Fix $i \in [k]$. Let $S \subseteq [N]$, $|S| \leq r$ be

arbitrary. We say that S is γ-good if

$$\Pr_{\mathbf{x}}[\mathcal{A}(C(\mathbf{x}), i) = \mathbf{x}_i \mid \mathcal{A} \text{ reads coordinates in } S] \geq 1/2 + \gamma. \tag{6.4}$$

Consider a hypergraph H that contains N vertices labeled by elements of $[N]$. The hyperedges of H, denoted E are defined by

$$E = \{e \subseteq [N] \mid e \text{ is } \alpha/2^r\text{-good}\}.$$

We now argue that the probability that $\mathcal{A}(\cdot, i)$ reads an edge from E is at least $\alpha/2^{r-1}$. To see this note that by formula (6.2)

$$\begin{aligned} 1/2 + \alpha/2^{r-1} &\leq \\ \Pr_{\mathbf{x}}[\mathcal{A}(C(\mathbf{x}), i) = \mathbf{x}_i \mid \mathcal{A}(\cdot, i) \text{ reads } E] \cdot \Pr[\mathcal{A}(\cdot, i) \text{ reads } E] &+ \\ \Pr_{\mathbf{x}}[\mathcal{A}(C(\mathbf{x}), i) = \mathbf{x}_i \mid \mathcal{A}(\cdot, i) \text{ reads } E^c] \cdot \Pr[\mathcal{A}(\cdot, i) \text{ reads } E^c] &\leq \\ \Pr[\mathcal{A}(\cdot, i) \text{ reads } E] + (1/2 + \alpha/2^r) \cdot (1 - \Pr[\mathcal{A}(\cdot, i) \text{ reads } E]). & \end{aligned}$$

For each hyperedge $e \in E$ let p_e denote the probability that $\mathcal{A}(\cdot, i)$ reads e. The argument above implies that

$$\sum_{e \in E} p_e \geq \alpha/2^{r-1}.$$

Furthermore, for every $j \in [N]$ formula (6.3) yields

$$\sum_{e \in E \mid j \in e} p_e \leq r/\delta N.$$

Let V be a vertex cover for the hypergraph H. Since for every $e \in E$ we have $e \cap V \neq \emptyset$, it follows that

$$\sum_{e \in E \mid e \cap V \neq \emptyset} p_e \geq \alpha/2^{r-1}.$$

Therefore

$$\alpha/2^{r-1} \leq \sum_{e \in E \mid e \cap V \neq \emptyset} p_e \leq \sum_{j \in V} \sum_{e \in E \mid j \in e} p_e \leq |V| r/\delta N,$$

which implies that the minimum vertex cover for H has size at least $m = \alpha\delta N/r2^{r-1}$. Recall that every hyperedge in H has cardinality at

most r. An application of a standard graph theory result implies that H contains a matching M, i.e., a collection of disjoint edges of size at least $|M| \geq m/r = \alpha\delta N/r^2 2^{r-1}$.

We define the new decoder $\mathcal{A}'$ as follows: on input i, $\mathcal{A}'$ picks one of the edges in the matching M uniformly at random, reads the corresponding codeword coordinates and runs $\mathcal{A}(\cdot, i)$ in a black box manner.

Step 4: (*Decoding by taking XOR.*) We first adjust the code C (by making it sometimes read some extra coordinates) and the decoding procedure $\mathcal{A}$ (by fixing some randomness of the decoder) to ensure that for all $i \in [k]$, $\mathcal{A}(\cdot, i)$ operates by randomly choosing a tuple t of *exactly* r codeword coordinates coming from a matching M_i, and then applying a *deterministic* function to $f_{i,t}(C(\mathbf{x})|_t)$ to obtain the output. For all $i \in [k]$ and $t \in M_i$ we have

$$\Pr_{\mathbf{x}\in\mathbb{F}_2^n}\left[\mathbf{x}_i = f_{i,t}\left(C(\mathbf{x})|_t\right)\right] \geq 1/2 + \alpha/2^r. \tag{6.5}$$

In what follows we modify the decoder $\mathcal{A}$ to ensure that for all indices i and tuples $t \in M_i$, the function $f_{i,t}$ is simply a modulo 2 sum.

Fix some $i \in [k]$ and $t \in M_i$. Consider a function $f = f_{i,t}$. Let $(c_1, \ldots, c_r)$ be the restriction of a codeword $C(\mathbf{x})$ to coordinates in t. Switching from the $\{0,1\}$ notation to the $\{1,-1\}$ notation yields

$$\mathbb{E}_{\mathbf{x}}\left[f(c_1, \ldots, c_r) \cdot \mathbf{x}_i\right] \geq \alpha/2^{r-1}.$$

Representing f in the Fourier basis we get

$$\frac{1}{2^r}\mathbb{E}_{\mathbf{x}}\left[\sum_{\chi} \hat{f}(\chi) \cdot \chi(c_1, \ldots, c_r) \cdot \mathbf{x}_i\right] \geq \alpha/2^{r-1},$$

where for every additive character χ of $\mathbb{F}_2^r$,

$$\hat{f}(\chi) = \sum_{\mathbf{c}} f(c_1, \ldots, c_r)\chi(c_1, \ldots, c_r).$$

Equivalently,

$$\sum_{\chi} \hat{f}(\chi)/2^r \cdot \mathbb{E}_{\mathbf{x}}\left[\chi(c_1, \ldots, c_r) \cdot \mathbf{x}_i\right] \geq \alpha/2^{r-1}.$$

Observe that for all $\chi \in \hat{\mathbb{F}}_2^r$ we have $|\hat{f}(\chi)/2^r| \leq 1$. Therefore there exists a character $\chi \in \hat{\mathbb{F}}_2^r$ such that

$$\mathbb{E}_{\mathbf{x}}\left[\chi(c_1, \ldots, c_r) \cdot \mathbf{x}_i\right] \geq \alpha/2^{2r-1}.$$

Returning to the $\{0,1\}$ notation, for some set $S \subseteq [r]$ we must have either

$$\Pr_{\mathbf{x}\in\mathbb{F}_2^k}\left[\mathbf{x}_i = \sum_{j\in S} C(\mathbf{x})_j\right] \geq \frac{1}{2} + \alpha/2^{2r},$$

or

$$\Pr_{\mathbf{x}\in\mathbb{F}_2^k}\left[\bar{\mathbf{x}}_i = \sum_{j\in S} C(\mathbf{x})_j\right] \geq \frac{1}{2} + \alpha/2^{2r},$$

Replacing every coordinate c of $C(\mathbf{x})$ with a triple $\{0, c, \bar{c}\}$, and having the local decoder access exactly one coordinate in every triple it touches, we bring the decoder to the normal form.

For each $i \in [k]$ the decoder operates by picking one of r-tuples of coordinates from a matching M_i at random, and outputting the modulo 2 sum. It is not hard to verify that our construction yields matchings of size at least $(1/2-\epsilon)\delta N/3 \cdot r^2 2^{r-1}$. The advantage over random guessing is at least $(1/2-\epsilon)/2^{2r}$. □

6.2 Polynomial lower bound for r-query codes

In this section we prove an $\Omega\left(k^{r/(r-1)}\right)$ lower bound for the codeword length of an arbitrary r-query locally decodable code. The main idea of the proof is that of a random restriction. We show that if a locally decodable code C is short, then a restriction of C to a randomly chosen small subset of coordinates carries too much information about the message.

Let $\mathcal{H}(\cdot)$ denote the standard entropy function. We need the following information theory lemma.

Lemma 6.3. Let $C : \mathbb{F}_2^k \to D$ be an arbitrary function. Assume there exists a randomized algorithm $\mathcal{A}$ such that for all $i \in [k]$,

$$\Pr_{\mathbf{x}}\left[\mathcal{A}(C(\mathbf{x}), i) = \mathbf{x}_i\right] \geq \frac{1}{2} + \beta,$$

where the probability is taken over the random coins of $\mathcal{A}$ as well as over all strings $\mathbf{x}$; then

$$\log|D| \geq (1 - \mathcal{H}(1/2+\beta))k.$$

Proof. Let $I(\mathbf{x}; C(\mathbf{x}))$ denote the mutual information between $\mathbf{x}$ and $C(\mathbf{x})$. We have

$$I(\mathbf{x}; C(\mathbf{x})) \leq \mathcal{H}(C(\mathbf{x})) \leq \log |D|.$$

Note that we also have

$$\begin{aligned} I(\mathbf{x}; C(\mathbf{x})) &= \mathcal{H}(\mathbf{x}) - \mathcal{H}(\mathbf{x}|C(\mathbf{x})) \\ &\geq \mathcal{H}(\mathbf{x}) - \sum_{i=1}^{k} \mathcal{H}(\mathbf{x}_i|C(\mathbf{x})) \\ &\geq (1 - \mathcal{H}(1/2 + \beta))k. \end{aligned}$$

Combining the inequalities above completes the proof. □

We are now ready to establish

Theorem 6.4. Suppose there exists an (r, δ, ϵ)-locally decodable code encoding k-bit messages to N-bit codewords; then we necessarily have

$$N \geq \Omega\left(\left(\frac{(1/2 - \epsilon)\delta}{r^2}\right)^{1/(r-1)} \left(\left(1 - \mathcal{H}\left(\frac{1}{2} + \frac{1/2 - \epsilon}{2^{2r}}\right)\right) \cdot k\right)^{r/(r-1)}\right).$$

provided that k is sufficiently large.

Proof. Assume the contrary. Then for infinitely many k we have codes violating the inequality from the theorem statement. Consider such a code C. Apply lemma 6.2 to turn C into a normal form. This yields an (r, η, β)-normally decodable code, where

$$\eta \geq \frac{(1/2 - \epsilon)\delta}{3 \cdot r^2 2^{r-1}} \quad \text{and} \quad \beta \geq \frac{1/2 - \epsilon}{2^{2r}}.$$

Let $\{M_i\}, i \in [k]$ be the collection of k matchings used by the decoder of C. Let α be a constant to be fixed later. Pick a set $S \subseteq [N]$ at random, including every element of $[N]$ into S with probability $\alpha k/N$. Let y be the random variable counting the number of matchings $\{M_i\}, i \in [k]$ that have at least one hyperedge completely contained in S. It is not hard to verify that

$$\mathbb{E}[y] \geq \left[1 - \left[1 - \left(\frac{\alpha k}{N}\right)^r\right]^{\eta N}\right] \cdot k \geq \left[1 - \left(\frac{1}{e}\right)^{\eta(\alpha k)^r / N^{r-1}}\right] \cdot k.$$

Since C violates the inequality from the theorem statement we have $N = O_{r,\delta,\epsilon}\left(k^{r/(r-1)}\right)$, where the absolute constant inside the $O(\cdot)$ notation can be set arbitrarily small. Thus the right hand side of the inequality above is at least $\Omega_{r,\delta,\epsilon}(k)$. Note that the random variable y takes non-negative integer values up to k. Therefore there is a positive constant probability that y is larger than $\mathbb{E}[y]/2$. Also note that by the Chernoff bound the probability that $|S| > 2\alpha k$ is exponentially small in k. Thus there exists a set $S \subseteq [N]$ such that $|S| \leq 2\alpha k$ and S contains a hyperedge from at least

$$m = 0.5 \cdot \left[1 - \left(\frac{1}{e}\right)^{\eta(\alpha k)^r / N^{r-1}}\right] \cdot k$$

distinct matchings $\{M_i\}, i \in [k]$. This implies that the restriction of C to coordinates in S allows one to make $(1/2+\beta)$-accurate predictions about m coordinates of $\mathbf{x}$. By lemma 6.3 we necessarily have

$$\left[1 - \left(\frac{1}{e}\right)^{\eta(\alpha k)^r / N^{r-1}}\right] \cdot (1 - \mathcal{H}(1/2+\beta)) \cdot k \leq 4\alpha k$$

Setting $\alpha = (1 - \mathcal{H}(1/2+\beta))/8$, after some basic manipulations we obtain

$$N \geq \Omega\left(k^{r/(r-1)} \cdot \eta^{1/(r-1)} \cdot \alpha^{r/(r-1)}\right).$$

Expressing η and α in terms of δ and ϵ we conclude the proof. □

6.3 Exponential lower bound for 2-query codes

In this section we prove an asymptotically tight $2^{\Omega(k)}$ lower bound for the codeword length of an arbitrary 2-query locally decodable code. The proof uses quantum information theory. We argue that short 2-query locally decodable codes yield short quantum random access codes and then apply a theorem of Nayak [72] bounding the length of such codes. We start with a brief introduction to quantum information theory needed for the proof. A comprehensive treatment of this area can be found in [73].

6.3.1 Quantum information theory

Let n be a positive integer. For our purposes an n-qubit quantum state is vector $\mathbf{q} \in \mathbb{R}^{2^n}$ such that $\sum_{j \in [2^n]} \mathbf{q}_j^2 = 1$. Let $B = \{\mathbf{b}_j\}, j \in [2^n]$ be an orthonormal basis of $\mathbb{R}^{2^n}$. Measuring a quantum state $\mathbf{q}$ in the basis B yields an output $j \in [2^n]$ with probability $(\mathbf{y}, \mathbf{b}_j)^2$.

A quantum *random access code* is an encoding $\mathbf{x} \to \mathbf{q}_\mathbf{x}$ of k-bit strings $\mathbf{x}$ into n-qubit states $\mathbf{q}_\mathbf{x}$, such that any individual bit $\mathbf{x}_i, i \in [k]$ can be recovered with some probability $p \geq 1/2 + \beta$ from $\mathbf{q}_\mathbf{x}$, where the probability is over a uniform choice of $\mathbf{x}$ and the measurement randomness. The following theorem which is a special case of the Holevo bound [57] is due to Nayak [72].

Theorem 6.5. Any encoding $\mathbf{x} \to \mathbf{q}_\mathbf{x}$ of k-bit strings into n-qubit states with recovery probability at least $1/2 + \beta$, necessarily has

$$n \geq (1 - \mathcal{H}(1/2 + \beta))k.$$

6.3.2 Lower bound

We are now ready to establish

Theorem 6.6. If there exists an $(2, \delta, \epsilon)$-locally decodable code C encoding k-bit messages to N-bit codewords; then

$$N \geq 2^{\Omega\left((1/2-\epsilon)^4 \delta^2 k\right)}.$$

Proof. Apply lemma 6.2 to turn the code C into a normal form. This yields an $(2, \eta, \beta)$-normally decodable code, where

$$\eta \geq \Omega((1/2 - \epsilon)\delta) \quad \text{and} \quad \beta \geq \Omega(1/2 - \epsilon).$$

We pad the code with zeros to ensure that the codeword length N is a power of two, $N = 2^n$. For every $\mathbf{x} \in \{0,1\}^k$ consider a n-qubit state $\mathbf{q}_\mathbf{x}$, where for all $j \in [N]$,

$$\mathbf{q}_j = (-1)^{C(\mathbf{x})_j} / \sqrt{N}. \tag{6.6}$$

We claim that the map $\mathbf{x} \to \mathbf{q_x}$ is a quantum random access code. Let $i \in [k]$ be arbitrary. To recover the bit $\mathbf{x}_i$ from the quantum state $\mathbf{q_x}$, we make a measurement in a suitable basis. Let $\mathbf{e}_m$ denote the m-th unit vector in $\mathbb{R}^N$, and let $M_i = \left\{(c_1^\ell, c_2^\ell)\right\}_{\ell \in [\eta N]}$ be the matching used by the normal decoder for C. Consider an orthonormal basis $B = \{\mathbf{b}_j\}$ for $\mathbb{R}^N$, where

$$\mathbf{b}_j = \begin{cases} \mathbf{e}_j & \text{if } j \notin \operatorname{supp}(M_i); \\ \frac{1}{\sqrt{2}}\left(\mathbf{e}_{c_1^\ell} + \mathbf{e}_{c_2^\ell}\right) & \text{if } j = c_1^\ell \text{ for some } \ell; \\ \frac{1}{\sqrt{2}}\left(\mathbf{e}_{c_1^\ell} - \mathbf{e}_{c_2^\ell}\right) & \text{if } j = c_2^\ell \text{ for some } \ell. \end{cases}$$

Observe that

$$(\mathbf{b}_j, \mathbf{q_x})^2 = \begin{cases} 1/N & \text{if } j \notin \operatorname{supp}(M_i); \\ 2/N & \text{if } j = c_1^\ell \text{ for some } \ell, \text{ and } C(\mathbf{x})_{c_1^\ell} \oplus C(\mathbf{x})_{c_2^\ell} = 0; \\ 0 & \text{if } j = c_1^\ell \text{ for some } \ell, \text{ and } C(\mathbf{x})_{c_1^\ell} \oplus C(\mathbf{x})_{c_2^\ell} = 1; \\ 2/N & \text{if } j = c_2^\ell \text{ for some } \ell, \text{ and } C(\mathbf{x})_{c_1^\ell} \oplus C(\mathbf{x})_{c_2^\ell} = 1; \\ 0 & \text{if } j = c_2^\ell \text{ for some } \ell, \text{ and } C(\mathbf{x})_{c_1^\ell} \oplus C(\mathbf{x})_{c_2^\ell} = 0. \end{cases}$$

The decoder for the quantum random access code interested in $\mathbf{x}_i$ measures the state $\mathbf{q_x}$ in the basis B. The decoder outputs a random bit if the measurement yields $j \notin \operatorname{supp}(M_i)$. Otherwise it outputs

$$\begin{cases} 0, & \text{if } \ j = c_1^\ell \text{ for some } (c_1^\ell, c_2^\ell) \in \operatorname{supp}(M_i); \\ 1, & \text{if } \ j = c_2^\ell \text{ for some } (c_1^\ell, c_2^\ell) \in \operatorname{supp}(M_i). \end{cases}$$

It is not hard to verify that the decoder above has an advantage of $\eta\beta$ over random guessing. Thus by theorem 6.5 we must have

$$n \geq (1 - \mathcal{H}(1/2 + \eta\beta)) \cdot k.$$

Expressing η and β in terms of δ and ϵ and using the fact that $1 - \mathcal{H}(1/2 + \tau) = \Theta(\tau^2)$ we conclude the proof. □

Extending the bound of theorem 6.6 to r-query codes for $r > 2$ remains an elusive goal.

6.4 Notes

Lemma 6.2 is from [60]. The $\Omega\left(k^{r/(r-1)}\right)$ lower bound established in section 6.2 is also from [60]. Somewhat stronger lower bounds of $\tilde{\Omega}\left(k^{1+1/(\lceil r/2 \rceil - 1)}\right)$ have been obtained in [94, 95].

The exponential lower bound for the length of 2-query codes given in section 6.3 is due to Kerenidis and de Wolf [62]. Our presentation follows [33, 92].

The dependence on δ and ϵ in theorem 6.6 can be improved to $(1/2 - \epsilon)^2\delta$ [62]. An alternative proof of this theorem is given [20], using an extension of the Bonami-Beckner hypercontractive inequality. However, that proof still follows the outline of the quantum-inspired proof presented here, albeit in linear-algebraic language.

Lower bounds locally decodable codes over infinite fields were studied in [37, 7]. Lower bounds for 2-query locally correctable codes were addressed in [22].

7

Applications

In this chapter we discuss some prominent applications of locally decodable codes, namely, applications to private information retrieval (section 7.1), secure multiparty computation (section 7.2), and average case complexity (section 7.3).

7.1 Private information retrieval

Private Information Retrieval (PIR) schemes are cryptographic protocols designed to safeguard the privacy of database users. They allow clients to retrieve records from public databases while completely hiding the identity of the retrieved records from database owners. The possibility of retrieving database records without revealing their identities to the owner of the database may seem beyond hope. Note, however, that a trivial solution is available: When users want a single record, they can ask for a copy of the whole database. This solution involves enormous communication overhead and is likely to be unacceptable. It turns out that for users who want to keep their privacy fully protected (in the information-theoretic sense), this trivial solution is optimal.

Fortunately, the negative result applies only to databases stored on

a single server, rather than those replicated across several servers. In 1995, Chor et al. [29] came up with PIR schemes that enable private retrieval of records from replicated databases, with a nontrivially small amount of communication. In such protocols, users query each server holding the database. The protocol ensures that each individual server (by observing only the query it receives) gets no information about the identity of the items of user interest.

We now make the notion of private information retrieval schemes more concrete. We model the database as a k-long q-ary string $\mathbf{x}$ that is replicated between r non-communicating servers. The user holds an index i (which is an integer between 1 and k) and is interested in obtaining the value of the i-th coordinate of $\mathbf{x}$. To achieve this goal, the user tosses some random coins, queries each of the r servers and gets replies from which the desired value can be computed. The query to each server is distributed independently of i therefore each server gets no information about what the user is after. Formally,

---

Definition 7.1. A r-server private information retrieval protocol is a triplet of non-uniform algorithms $\mathcal{P} = (\mathcal{Q}, \mathcal{A}, \mathcal{C})$. We assume that each algorithm is given k as an advice. At the beginning of the protocol, the user $\mathcal{U}$ tosses random coins and obtains a random string rand. Next $\mathcal{U}$ invokes $\mathcal{Q}(i, \text{rand})$ to generate an r-tuple of queries $(\text{que}_1, \ldots, \text{que}_r)$. For $j \in [r]$, $\mathcal{U}$ sends que_j to the server $\mathcal{S}_j$. Each server $\mathcal{S}_j$, $j \in [r]$ responds with an answer $\text{ans}_j = \mathcal{A}(j, \mathbf{x}, \text{que}_j)$. Finally, $\mathcal{U}$ computes its output by applying the reconstruction algorithm $\mathcal{C}(\text{ans}_1, \ldots, \text{ans}_r, i, \text{rand})$. A protocol as above should satisfy the following requirements:

- **Correctness :** For any k, $\mathbf{x} \in [q]^k$ and $i \in [k]$, $\mathcal{U}$ outputs the correct value of $\mathbf{x}_i$ with probability 1 (where the probability is over the random strings rand).
- **Privacy :** Each server individually learns no information about i. More precisely, we require that for any k and for any $j \in [r]$, the distributions $\text{que}_j(i, \text{rand})$ are identical for all values $i \in [k]$.

The *communication complexity* of a PIR protocol $\mathcal{P}$, is a function of k measuring the total number of bits communicated between the user and the servers, maximized over all choices of $\mathbf{x} \in [q]^k$, $i \in [k]$, and random inputs. The major goal of PIR related research to design r-server private information retrieval schemes with optimal (i.e., the smallest possible) amount of communication for every r.

Following the paper of Chor et al. [29] there has been a large a body of work on private information retrieval [3, 14, 15, 97, 94, 99, 79, 38, 59]. A large number of extensions of the basic PIR model have also been studied. These include extensions to t-private protocols, in which the user is protected against collusions of up to t servers [14, 9]; extensions which protect the servers holding the database in addition to the user, termed symmetric PIR [47, 71]; extensions to computational schemes [64, 25, 66, 45] that only ensure that a server cannot get any information about the user's intensions unless it solves a certain computationally hard problem; and other extensions [16, 17, 32, 46, 75]. In many of those extensions the protocols are obtained by adding some extra layers on top of a basic private information retrieval scheme. Therefore improving parameters of basic private information retrieval schemes yields improvements for many other problems. See [42, 100] for surveys of PIR literature.

The gap between upper and lower bounds for communication complexity of private information retrieval schemes is fairly large. Currently, the most efficient r-server schemes for $r \geq 3$ are obtained through r-query locally decodable codes. Communication complexity of such schemes is roughly logarithmic in the codeword length of corresponding codes. This, for instance, yields 3-server schemes with $\exp\left(\sqrt{\log k \log\log k}\right)$ communication to access a k-bit database [38]. Two server private information retrieval schemes do not rely on LDCs. The most efficient such schemes to date require $O(k^{1/3})$ communication [29]. The best lower bound for the communication complexity of two server PIR is $5 \log k$ due to Wehner and de Wolf [94]. Single server PIR schemes require $\Theta(k)$ communication [29].

In what follows we review existing constructions of private information retrieval schemes in more detail. In section 7.1.1 we discuss the construction of schemes from locally decodable codes and in section 7.1.2

we present a two server scheme based on polynomial interpolation.

7.1.1 From codes to schemes

The following lemma obtains an r-server private information retrieval scheme out of any perfectly smooth r-query locally decodable code, i.e., a code where each decoder's query is distributed perfectly uniformly over the set of codeword coordinates.

Lemma 7.2. Suppose there exists a perfectly smooth q-ary r-query locally decodable code C encoding k-long messages to N-long codewords; then there exists an r-server private information retrieval scheme with $O(r \cdot \log_2(Nq))$ communication to access a q-ary k-long database.

Proof. At the preprocessing stage servers $\mathcal{S}_1, \ldots, \mathcal{S}_r$ encode the k-long database $\mathbf{x}$ with the code C. Next the user $\mathcal{U}$ who is interested in obtaining the value of the i-th coordinate of $\mathbf{x}$, tosses random coins and generates an r-tuple of queries $(\mathrm{que}_1, \ldots, \mathrm{que}_r)$, such that $\mathbf{x}_i$ can be computed from $C(x)_{\mathrm{que}_1}, \ldots, C(x)_{\mathrm{que}_r}$. For every $j \in [r]$, the user sends the query que_j to the server $\mathcal{S}_j$. Each server $\mathcal{S}_j$ responds with a q-ary value $C(\mathbf{x})_{\mathrm{que}_j}$. The user combines servers' responses to obtain $\mathbf{x}_i$.

It is straightforward to verify that the protocol above is private since for every $j \in [r]$ the query que_j is uniformly distributed over the set of codeword coordinates. The total communication is given by $r \cdot (\lceil \log_2 N \rceil + \lceil \log_2 q \rceil)$. □

Combining lemma 7.2 with theorem 4.10 we get

Theorem 7.3. For every integer $t \geq 2$, and for all $k \geq 2$, there exists a $3 \cdot 2^{t-2}$-server private information retrieval scheme with

$$\exp_t \left((\log k)^{1/t} (\log \log k)^{1-1/t} \right) -$$

bit communication to access a k-bit database.

7.1.2 Two server private information retrieval

Below we present a two-server PIR scheme due to Woodruff et al. [97]. The scheme involves $O\left(k^{1/3}\right)$ communication to access a k-bit

database. Arguably this scheme is more intuitive compared to other schemes of the same communication complexity [29, 14]. The ideas behind the scheme are similar to those behind Reed Muller locally decodable codes (section 2.2).

Let n be an arbitrary positive integer. Set $k = \binom{n}{3}$. In what follows we obtain a 2-server scheme with $O(n)$ bits of communication to access an k-bit database. Pick $\gamma : [k] \to \{0,1\}^n$ to be an arbitrary bijection between the set $[k]$ and the set of n-dimensional $\{0,1\}$-vectors of Hamming weight three. For $i \in [k]$ and $j \in \{1,2,3\}$ let $\gamma(i)_j$ denote the j-th nonzero coordinate of $\gamma(i)$.

Let $q \geq 3$ be a fixed prime power. Given a database $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_k) \in \{0,1\}^k$ each server obtains the following polynomial F in the ring $\mathbb{F}_q[z_1, \ldots, z_n]$,

$$F(z_1, \ldots, z_n) = \sum_{i=1}^{k} \mathbf{x}_i \cdot z_{\gamma(i)_1} \cdot z_{\gamma(i)_2} \cdot z_{\gamma(i)_3}.$$

The key properties of the polynomial F are the following:

- F encodes the database: For every $i \in [k]$, $F(\gamma(i)) = \mathbf{x}_i$;
- F has low degree: $\deg f = 3$.

The basic idea behind our private information retrieval scheme is the idea of polynomial interpolation. Suppose the user wants to retrieve the i-th coordinate of the database. Given i, the user obtains the vector $\mathbf{w} = \gamma(i) \in \mathbb{F}_q^n$. Now the user's goal is to recover the value of the polynomial F (held by the servers) at the point $\mathbf{w}$.

Obviously, the user cannot explicitly request the value of F at $\mathbf{w}$ from any of the servers, since such a request would ruin the privacy of the protocol; that is, some server will get to know which database bit the user is after. Instead, the user obtains the value of $F(\mathbf{w})$ indirectly, relying on the rich structure of local dependencies between the evaluations of a cubic polynomial F at multiple points. Specifically, the user randomly selects an affine line $L \in \mathbb{F}_q^n$ containing the point $\mathbf{w}$ and discloses certain points on L to the servers. Each server computes and returns the value of F and the values of partial derivatives of F at the point that it is given. Finally, the user reconstructs the restriction of

F to L. In particular the user obtains the desired value $F(\mathbf{w})$. Below is a more formal description.

We use the standard mathematical notation $\left.\frac{\partial F}{\partial z_l}\right|_{\mathbf{y}}$ to denote the value of the partial derivative [65] of F with respect to a variable z_l at a point $\mathbf{y}$. Note that first formal derivatives are identical to first Hasse derivatives used in chapter 3.

Let $\lambda_1, \lambda_2 \in \mathbb{F}_q$ be distinct and non-zero. Let $\mathcal{U}$ denote the user and $\mathcal{S}_1, \mathcal{S}_2$ denote the servers. The protocol proceeds as follows,

$\mathcal{U}$	:	Picks $\mathbf{v} \in \mathbb{F}_q^n$ uniformly at random.
$\mathcal{U} \rightarrow \mathcal{S}_h$	:	$\mathbf{w} + \lambda_h \mathbf{v}$
$\mathcal{U} \leftarrow \mathcal{S}_h$	:	$F(\mathbf{w} + \lambda_h \mathbf{v}), \left.\frac{\partial F}{\partial z_1}\right\|_{\mathbf{w}+\lambda_h\mathbf{v}}, \ldots, \left.\frac{\partial F}{\partial z_m}\right\|_{\mathbf{w}+\lambda_h\mathbf{v}}$

Note that in the protocol above the input of each server $\mathcal{S}_h, h \in \{1, 2\}$ is a uniformly random point in $\mathbb{F}_q^n$. Therefore the protocol is private. It is also easy to verify that both the queries that the user sends to servers and the servers' responses are of length $O(n) = O(k^{1/3})$. (Every query is simply a point in $\mathbb{F}_q^n$. Every response is a list of n values of partial derivatives of F plus the value of F itself.) It remains to show how the user obtains $F(\mathbf{w})$ from the servers' responses.

Consider the line $L = \{\mathbf{w} + \lambda \mathbf{v} \mid \lambda \in \mathbb{F}_q\}$. Let $f(\lambda) = f(\mathbf{w} + \lambda \mathbf{v}) \in \mathbb{F}_q[\lambda]$ be the restriction of F to L. Clearly, $f(\lambda_h) = F(\mathbf{w} + \lambda_h \mathbf{v})$. Thus the user knows the values $\{f(\lambda_h)\}$ for $h \in \{1, 2\}$. This, however, does not suffice to reconstruct the polynomial f, since the degree of f may be up to three. The main observation underlying our protocol is that knowing the values of partial derivatives $\left.\frac{\partial F}{\partial z_1}\right|_{\mathbf{w}+\lambda_h\mathbf{v}}, \ldots, \left.\frac{\partial F}{\partial z_n}\right|_{\mathbf{w}+\lambda_h\mathbf{v}}$, the user can reconstruct the value of $f'(\lambda_h)$. The proof is a straightforward application of the chain rule:

$$\left.\frac{\partial f}{\partial \lambda}\right|_{\lambda_h} = \left.\frac{\partial F(\mathbf{w} + \lambda \mathbf{v})}{\partial \lambda}\right|_{\lambda_h} = \sum_{l=1}^{n} \left.\frac{\partial F}{\partial z_l}\right|_{\mathbf{w}+\lambda_h\mathbf{v}} \mathbf{v}_l.$$

Thus the user can reconstruct $\{f(\lambda_h)\}$ and $\{f'(\lambda_h)\}$ for $h \in \{1, 2\}$. Combining this observation with the standard algebraic fact that a cubic univariate polynomial is uniquely determined by its values and derivatives at two points [65], we conclude that the user can reconstruct f and obtain $\mathbf{x}_i = F(\mathbf{w}) = f(0)$.

7.2 Secure multiparty computation

A fundamental result of Ben-Or et al. [21] and Chaum et al. [27] from 1988 asserts that information-theoretic secure multiparty computation is feasible. Specifically, in [21, 27] it is shown that $r \geq 3$ players that are allowed to exchange messages over secure channels, can jointly compute any function of their local inputs while hiding the inputs from each other; i.e., one can always arrange a protocol as to ensure that after performing the joint computation any specific player gets no information about the inputs of other players (apart from the information contained in the value of the function).

In all known protocols for secure multiparty computation the communication complexity of the protocol grows linearly with the circuit size of the function being computed. This results in $2^{\Omega(k)}$ amount of communication for securely computing most of the functions of k-bit inputs. A natural question that was explicitly asked in several papers from the late 1980's and early 1990's [13, 12] is whether *all* functions can be securely computed with only a polynomial (or at least a subexponential) amount of communication in the input length. It was observed by Ishai and Kushilevtiz [58] that this question is closely related to the complexity of private information retrieval schemes.

The construction of private information retrieval schemes given in section 7.1 yields quantitative progress on the question mentioned above (via the reduction of [58]). Specifically, theorem 7.3 implies that a group of 18 or more players can securely compute any function of their k-bit inputs with a total communication of $\exp\left(\sqrt{k \log k}\right)$, for all k.

7.3 Average-case complexity

One of the earliest applications of locally decodable codes was to worst-case to average-case reductions in computational complexity theory. This application requires LDCs with polynomial length and polylogarithmic query complexity. Such codes can be obtained from Reed Muller codes. Below we give a sketch of this application. Our presentation follows [93, Section 3.5].

Let L be an EXP-complete problem, and for an input length t let us consider the restriction of L to inputs of length t. We can see L, restricted to these inputs, as a binary string of length 2^t. Let us encode this string using a polynomial-length locally decodable code C that has polylogarithmic query complexity and can tolerate some constant fraction of errors. We get a string of length $2^{O(t)} = 2^{t'}$, and let us think of this string as defining a new problem L' on inputs of length t'. If L is in EXP, then so is L'. The properties of the code C imply that a polynomial-time algorithm for L' that is good on average (i.e., solves L' correctly on, say, some fraction $1 - \epsilon$ of the inputs in polynomial time) yields a probabilistic algorithm for L that works on all inputs, and EXP $\subseteq$ BPP. This argument shows that if every problem in EXP can be solved well on average, then EXP $\subseteq$ BPP. A similar statement can be proved for PSPACE using a variant of this argument.

7.4 Notes

Locally decodable codes have other applications in complexity theory [34, 7], data structures [30, 28], derandomization [37], and theory of fault tolerant computation [82].

8

Future directions

In this chapter we list and comment on the most exciting open questions relating to locally decodable codes and private information retrieval schemes.

8.1 3-query locally decodable codes

It is very interesting to determine the optimal length of 3-query codes. The best upper bound to date is $\exp\exp\left(\sqrt{\log k \cdot \log\log k}\right)$ (section 4.6.1). The best lower bound is $\tilde{\Omega}(k^2)$ (section 6.2).

A natural approach to improving the upper bound is through the matching vector codes machinery detailed in chapters 4 and 5. This calls for constructing families of S-matching vectors in $\mathbb{Z}_m^n$, for small sets S of size larger than what one gets from the Grolmusz construction. We remark that improving the Grolmusz construction for constant values of m will have significant implications other than improved upper bounds for locally decodable codes, e.g., [50] (if explicit) it will give an explicit family of Ramsey graphs beyond the Frankl-Wilson bound different from [8]. One approach to improving the Grolmusz construction is to improve upper bounds for the degree of polynomial representation

of the OR-function modulo composites [10, 91].

8.2 r-query locally decodable codes

Currently matching vector codes are the best known LDCs in the regime of low query complexity, Reed Muller codes are the best known LDCs in the regime of medium query complexity, and multiplicity codes are the best known LDCs in the regime of high query complexity. The two natural next benchmarks for code constructions are the following:

- Construct codes with $r = k^{o(1)}$ and positive rate;
- Construct codes with $r = O(\log k)$ and polynomial stretch.

Multiplicity codes come very close but fail to achieve the first benchmark, i.e., for every $\epsilon > 0$ they give k^{ϵ}-query codes of positive rate. Reed Muller codes codes come close to the second benchmark. In particular the length of RM codes of query complexity $\log k$ is only slightly superpolynomial (section 2.3). Matching vector codes based on Grolmusz matching families improve upon RM codes for all values of $r < \log k/(\log\log k)^c$. Is is plausible that further progress on MV codes may help us achieve the benchmark. This calls for new bounded families of matching vectors in $\mathbb{Z}_m^n$, where m is comparable to (or larger than) n. This regime has almost not been addressed in the past.

It is also interesting to understand the power of matching vector codes for values of $r > \log k$. The following conjecture has been made in this regard in [35]. (Recall that $k(m, n)$ denotes the size of the largest $\mathbb{Z}_m \setminus \{0\}$-matching family in $\mathbb{Z}_m^n$.)

Conjecture 8.1. Let m and n be arbitrary positive integers; then

$$k(m, n) \leq O\left(m^{n/2}\right).$$

By lemma 5.20 the conjecture holds for prime m. If the conjecture holds in general; then any matching vector code must have length $N = \Omega(k^2)$, and thus MV codes are inferior to Reed Muller codes once $r \geq \log^2 k$ by an argument similar to the one in section 4.7.2.

8.3 Locally correctable codes

To date Reed Muller codes and multiplicity codes constitute the only known classes of locally correctable codes. It is interesting to see if there are locally correctable codes in the regime of low query complexity that are shorter than Reed Muller codes. In particular we do not know if matching vector codes can be made locally correctable.

8.4 Two server private information retrieval

Unlike PIR schemes involving three or more servers, existing two server schemes are not based on locally decodable codes. While a number of different two server schemes are known [29, 3, 14, 15, 97], all of them have the same asymptotic communication complexity of $O\left(k^{1/3}\right)$ as the earliest such schemes proposed in [29]. The best lower bound is $5 \cdot \log k$ from [94].

One approach to improving the bounds for the communication complexity of two server private information retrieval has been proposed in [80] where it was shown that under some weak technical restriction two-server schemes with $O(\log N)$-communication to access a k-bit database are equivalent to matrices M of size $N \times N$ with entries from the alphabet $\{x_1, \ldots, x_k, *\}$ such that:

(1) Every variable $x_i, i \in [k]$ appears exactly once in each row and each column of M;
(2) For all 2^k assignments of $\mathbb{F}_2$ values to variables $\{x_i\}_{i \in [k]}$, there is a completion of the matrix, (i.e., assignment of $\mathbb{F}_2$ values to locations containing stars) such that the $\mathbb{F}_2$-rank of the resulting matrix is $O(\log N)$.

Thus to improve the communication complexity of the best known two-server scheme it suffices to exhibit a matrix M as above with $k = \omega(\log^3 N)$. Similarly, to improve the lower bounds it suffices to show that no matrix M as above can have $k = \Omega(N^\epsilon)$, for a constant ϵ.

8.5 Private information retrieval with preprocessing

Our review of the state of the art in private information retrieval has concentrated on the most studied aspect of PIR schemes, namely, their communication complexity. Another important aspect of such schemes is the amount of computation that servers need to perform in order to respond to user queries. In fact, it is the overwhelming *computational complexity* of PIR schemes, that currently presents the main bottleneck to their practical deployment.

Computational complexity of early private information retrieval schemes has been addressed in [16, 97] where it was shown that preprocessing the database can lead to notable savings. It will be interesting to see further results in this direction as well as address the computational complexity of private information retrieval schemes arising from matching vector codes.

Acknowledgements

We would like to thank Parikshit Gopalan for carefully reading a draft of this survey and providing detailed and helpful comments.

References

[1] Noga Alon. Eigenvalues, geometric expanders, sorting in rounds, amd Ramsey theory. *Combinatorica*, 6:207–219, 1986.

[2] Noga Alon and Joel Spencer. *The probabilistic method*. 2000.

[3] Andris Ambainis. Upper bound on the communication complexity of private information retrieval. In *32nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 1256 of Lecture Notes in Computer Science, pages 401–407. Springer, Berlin, Heidelberg, 1997.

[4] Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23:365–426, 2003.

[5] Laszlo Babai, Lance Fortnow, Leonid Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *23rd ACM Symposium on Theory of Computing (STOC)*, pages 21–31, 1991.

[6] Laszlo Babai and Peter Frankl. *Linear algebra methods in combinatorics*. 1998.

[7] Boaz Barak, Zeev Dvir, Avi Wigderson, and Amir Yehudayoff. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *43nd ACM Symposium on Theory of Computing (STOC)*, 2011.

[8] Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and Ramsey graphs beating the Frankl-Wilson construction. In *38th ACM Symposium on Theory of Computing (STOC)*, pages 671–680, 2006.

[9] Omer Barkol, Yuval Ishai, and Enav Weinreb. On locally decodable codes, self-correctable codes, and t-private PIR. In *International Workshop on Randomization and Computation (RANDOM)*, pages 311–325, 2007.

[10] David A. Barrington, Richard Beigel, and Steven Rudich. Representing Boolean functions as polynomials modulo composite numbers. *Computational Complexity*, 4:67–382, 1994.

[11] Donald Beaver and Joan Feigenbaum. Hiding instances in multioracle queries. In *7th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 415 of Lecture Notes in Computer Science, pages 37–48. Springer, Berlin, Heidelberg, 1990.

[12] Donald Beaver, Joan Feigenbaum, Joe Kilian, and Phillip Rogaway. Security with low communication overhead. In *International Cryptology Conference (CRYPTO)*, pages 62–76, 1990.

[13] Donald Beaver, Silvio Micali, and Pillip Rogaway. The round complexity of secure protocols. In *22nd ACM Symposium on Theory of Computing (STOC)*, pages 503–513, 1990.

[14] Amos Beimel, Yuval Ishai, and Eyal Kushilevitz. General constructions for information-theoretic private information retrieval. *Journal of Computer and System Sciences*, 71:213–247, 2005.

[15] Amos Beimel, Yuval Ishai, Eyal Kushilevitz, and Jean-Francios Raymond. Breaking the $O\left(n^{1/(2k-1)}\right)$ barrier for information-theoretic private information retrieval. In *43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 261–270, 2002.

[16] Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the servers' computation in private information retrieval: PIR with preprocessing. In *International Cryptology Conference (CRYPTO)*, volume 1880 of Lecture Notes in Computer Science, pages 56–74. Springer, Berlin, Heidelberg, 2000.

[17] Amos Beimel and Yoav Stahl. Robust information theoretic private information retrieval. In *3rd Conference on Security in Communication Networks*, 2002.

[18] Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. Local list decoding with a constant number of queries. In *51st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 715–722, 2010.

[19] Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. A note on amplifying the error-tolerance of locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR10-134, 2010.

[20] Avraham Ben-Aroya, Oded Regev, and Ronald de Wolf. A hypercontractive inequality for matrix-valued functions with applications to quantum computing and ldcs. In *49rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 477–486, 2008.

[21] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *20th ACM Symposium on Theory of Computing (STOC)*, pages 1–10, 1988.

[22] Arnab Bhattacharyya, Zeev Dvir, Shubhangi Saraf, and Amir Shpilka. Tight lower bounds for 2-query lccs over finite fields. In *52st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 638–647, 2011.

[23] Manuel Blum and Sampath Kannan. Designing programs that check their work. In *21th ACM Symposium on Theory of Computing (STOC)*, pages 86–97, 1989.

[24] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993.

[25] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In *International Cryptology Conference (EUROCRYPT)*, volume 1592 of Lecture Notes in Computer Science, pages 402–414. Springer, Berlin, Heidelberg, 1999.

[26] Peter J. Cameron. *Combinatorics: topics, techniques, algorithms*. 1994.

[27] David Chaum, Claude Crepeau, and Ivan Damgard. Multiparty unconditionally secure protocols. In *20th ACM Symposium on Theory of Computing (STOC)*, pages 11–19, 1988.

[28] Victor Chen, Elena Grigorescu, and Ronald de Wolf. Efficient and error-correcting data structures for membership and polynomial evaluation. In *27th Symposium on Theoretical Aspects of Computer Science (STACS)*, 2010.

[29] Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. *Journal of the ACM*, 45:965–981, 1998.

[30] Ronald de Wolf. Error-correcting data structures. In *26th Annual Symposium on Theoretical Aspects of Computer Science (STACS 09)*, pages 313–324, 2009.

[31] A. Deshpande, R. Jain, T. Kavitha, S. Lokam, and J. Radhakrishnan. Better lower bounds for locally decodable codes. In *20th IEEE Computational Complexity Conference (CCC)*, pages 184–193, 2002.

[32] Giovanni Di-Crescenzo, Yuval Ishai, and Rafail Ostrovsky. Universal service-providers for private information retrieval. *Journal of Cryptology*, 14:37–74, 2001.

[33] Andrew Drucker and Ronald de Wolf. Quantum proofs for classical theorems. Arxiv 0910.3376, 2009.

[34] Zeev Dvir. On matrix rigidity and locally self-correctable codes. In *25th IEEE Computational Complexity Conference (CCC)*, pages 102–113, 2010.

[35] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM Journal on Computing*, 2011. to appear.

[36] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to Kakeya sets and mergers. In *50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 181–190, 2009.

[37] Zeev Dvir and Amir Shpilka. Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits. *SIAM Journal on Computing*, 36(5):1404–1434, 2006.

[38] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *41st ACM Symposium on Theory of Computing (STOC)*, pages 39–44, 2009.

[39] Peter Elias. List decoding for noisy channels. Research laboratory for electronics, MIT, 1957.

[40] G. David Forney. *Concatenated codes*. MIT Press, Cambridge, 1966.

[41] Anna Gal and Andrew Mills. Three query locally decodable codes with higher correctness require exponential length. In *28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 673–684, 2011.

[42] William Gasarch. A survey on private information retrieval. *The Bulletin of the EATCS*, 82:72–107, 2004.

[43] Peter Gemmell, Richard Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self testing / correcting for polynomials and for approximate functions. In *23th ACM Symposium on Theory of Computing (STOC)*, pages 32–42, 1991.

[44] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43:169–174, 1992.

[45] Craig Gentry and Zulfikar Ramzan. Single-database private information retrieval with constant communication rate. In *32nd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 803–815, 2005.

[46] Yael Gertner, Shafi Goldwasser, and Tal Malkin. A random server model for private information retrieval. In *International Workshop on Randomization and Computation (RANDOM)*, volume 1518 of Lecture Notes in Computer Science, pages 200–217. Springer, Berlin, Heidelberg, 1998.

[47] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting data privacy in private information retrieval schemes. *Journal of Computer and System Sciences*, 60:592–629, 2000.

[48] Oded Goldreich, Howard Karloff, Leonard Schulman, and Luca Trevisan. Lower bounds for locally decodable codes and private information retrieval. In *17th IEEE Computational Complexity Conference (CCC)*, pages 175–183, 2002.

[49] Parikshit Gopalan. *Computing with Polynomials over Composites*. PhD thesis, Georgia Institute of Technology, 2006.

[50] Vince Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, 20:71–86, 2000.

[51] Vince Grolmusz. Constructing set-systems with prescribed intersection sizes. *Journal of Algorithms*, 44:321–337, 2002.

[52] Venkat Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.

[53] Venkatesan Guruswami. *List decoding of error-correcting codes*. PhD thesis, Massachusetts Institite of Technology, 2001.

[54] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54:135–150, 2008.

[55] Venkatesan Guruswami, Amit Sahai, and Madhu Sudan. Soft-decision decoding of Chinese Remainder codes. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pages 159–168, Redondo Beach, California, 12-14 November 2000.

[56] G. Hardy and E. Wright. *An introduction to the theory of numbers*. Clarendon Press, Oxford, 1985.

[57] Alexander Holevo. Bounds for the quantity of information transmitted by a quantum communication channel. *Problems of Information Transmission*, 9:177–183, 1973.

[58] Yuval Ishai and Eyal Kushilevitz. On the hardness of information-theoretic multiparty computation. In *Eurocrypt 2004*, volume 3027 of Lecture Notes in Computer Science, pages 439–455. Springer, Berlin, Heidelberg, 2004.

[59] Toshiya Itoh and Yasuhiro Suzuki. New constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions on Information and Systems*, pages 263–270, 2010.

[60] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *32nd ACM Symposium on Theory of Computing (STOC)*, pages 80–86, 2000.

[61] Kiran S. Kedlaya and Sergey Yekhanin. Locally decodable codes from nice subsets of finite fields and prime factors of Mersenne numbers. *SIAM Journal on Computing*, 38:1952–1969, 2009.

[62] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences*, 69:395–420, 2004.

[63] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. In *43nd ACM Symposium on Theory of Computing (STOC)*, pages 167–176, 2011.

[64] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single-database computationally-private information retrieval. In *38rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 364–373, 1997.

[65] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Cambridge University Press, Cambridge, 1983.

[66] Helger Lipmaa. An oblivious transfer protocol with log-squared communication. International Association for Cryptologic Research, Technical Report 2004/063, 2004.

[67] Richard Lipton. Efficient checking of computations. In *7th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 415 of Lecture Notes in Computer Science, pages 207–215. Springer, Berlin, Heidelberg, 1990.

[68] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error Correcting Codes*. North Holland, Amsterdam, New York, 1977.

[69] Yeow Meng Chee, Tao Feng, San Ling, Huaxiong Wang, and Liangfeng Zhang. Query-efficient locally decodable codes of subexponential length. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR10-173, 2010.

[70] D. E. Muller. Application of boolean algebra to switching circuit design and to error detection. *IEEE Transactions on Computers*, 3:6–12, 1954.

[71] Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *29th ACM Symposium on Theory of Computing (STOC)*, pages 245–254, 1999.

[72] Ashwin Nayak. Optimal lower bounds for quantum automata and random access codes. In *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 369–377, 1999.

[73] Michael Nielsen and Isaac Chuang. *Quantum computation and quantum information*. Cambridge university press, Cambridge, 2000.

[74] Kenji Obata. Optimal lower bounds for 2-query locally decodable linear codes. In *6th International Workshop on Randomization and Computation (RANDOM)*, volume 2483 of Lecture Notes in Computer Science, pages 39–50. Springer, Berlin, Heidelberg, 2002.

[75] Rafail Ostrovsky and Victor Shoup. Private information storage. In *29th ACM Symposium on Theory of Computing (STOC)*, pages 294–303, 1997.

[76] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *46th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–294, 2005.

[77] W. Wesley Peterson and Jr. E. J. Weldon. *Error correcting codes*. 1972.

[78] Alexander Polishchuk and Daniel Spielman. Nearly-linear size holographic proofs. In *26th ACM Symposium on Theory of Computing (STOC)*, pages 194–203, 1994.

[79] Prasad Raghavendra. A note on Yekhanin's locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR07-016, 2007.

[80] Alexander Razborov and Sergey Yekhanin. An $\Omega(n^{1/3})$ lower bound for bilinear group based private information retrieval. *Theory of Computing*, 3:221–238, 2007.

[81] Irving S. Reed. A class of multiple-error-correcting codes and the decoding scheme. *IEEE Transactions on Information Theory*, 4:38–49, 1954.

[82] Andrei Romashchenko. Reliable computations based on locally decodable codes. In *23rd International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 3884 of Lecture Notes in Computer Science, pages 537–548. Springer, Berlin, Heidelberg, 2006.

[83] Michael Rozenbloom and Michael Tsfasman. Codes for the m-metric. *Problems Inform. Transmission*, 33:45–52, 1997.

[84] Shubhangi Saraf and Sergey Yekhanin. Noisy interpolation of sparse polynomials, and applications. In *26th IEEE Computational Complexity Conference (CCC)*, 2011.

[85] Shunhangi Saraf and Madhu Sudan. Improved lower bound on the size of Kakeya sets over finite fields. *Analysis and PDE*, 1:375–379, 2008.

[86] Jiri Sgall. Bounds on pairs of families with restricted intersections. *Combinatorica*, 19:555–566, 1999.

[87] V. Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, Cambridge, 2005.

[88] Madhu Sudan. Ideal error-correcting codes: Unifying algebraic and number-theoretic algorithms. In *AAECC*, pages 36–45, 2001.

[89] Madhu Sudan. Personal communication, 2009.

[90] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. In *39th ACM Symposium on Theory of Computing (STOC)*, pages 537–546, 1999.

[91] Gabor Tardos and David Barrington. A lower bound of the mod 6 degree of the OR function. *Computational Complexity*, 7:99–108, 1998.

[92] Luca Trevisan. Coding theory and complexity. Lecture notes, 2003.

[93] Luca Trevisan. Some applications of coding theory in computational complexity. *Quaderni di Matematica*, 13:347–424, 2004.

[94] Stephanie Wehner and Ronald de Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In *32nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of Lecture Notes in Computer Science, pages 1424–1436. Springer, Berlin, Heidelberg, 2005.

[95] David Woodruff. New lower bounds for general locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR07-006, 2007.

[96] David Woodruff. A quadratic lower bound for three query linear locally decodable codes over any field. In *International Workshop on Randomization and Computation (RANDOM)*, 2010.

[97] David Woodruff and Sergey Yekhanin. A geometric approach to information theoretic private information retrieval. In *20th IEEE Computational Complexity Conference (CCC)*, pages 275–284, 2005.

[98] Chaoping Xing. Nonlinear codes from algebraic curves improving the Tsfasman-Vladut-Zink bound. *IEEE Transactions on Information Theory*, 49:1653–1657, 2003.

[99] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM*, 55:1–16, 2008.

[100] Sergey Yekhanin. Private information retrieval. *Communications of the ACM*, 53(4):68–73, 2010.