The Method of Multiplicities

by

Shubhangi Saraf

B.S., Massachusetts Institute of Technology (2007)
S.M., Massachusetts Institute of Technology (2009)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2011

Author .....
Department of Electrical Engineering and Computer Science
September 1, 2011

Certified by .....
Madhu Sudan
Professor
Thesis Supervisor

Accepted by .....
Leslie A. Kolodziejski
Chairman, Department Committee on Graduate Students

The Method of Multiplicities

by

Shubhangi Saraf

Submitted to the Department of Electrical Engineering and Computer Science
on September 1, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

Polynomials have played a fundamental role in the construction of objects with interesting combinatorial properties, such as error correcting codes, pseudorandom generators and randomness extractors. Somewhat strikingly, polynomials have also been found to be a powerful tool in the analysis of combinatorial parameters of objects that have some algebraic structure. This method of analysis has found applications in works on list-decoding of error correcting codes, constructions of randomness extractors, and in obtaining strong bounds for the size of Kakeya Sets. Remarkably, all these applications have relied on very simple and elementary properties of polynomials such as the sparsity of the zero sets of low degree polynomials.

In this thesis we improve on several of the results mentioned above by a more powerful application of polynomials that takes into account the information contained in the *derivatives* of the polynomials. We call this technique the *method of multiplicities*. The derivative polynomials encode information about the *high multiplicity* zeroes of the original polynomial, and by taking into account this information, we are about to meaningfully reason about the zero sets of polynomials of degree much higher than the underlying field size. This freedom of using high degree polynomials allows us to obtain new and improved constructions of error correcting codes, and qualitatively improved analyses of Kakeya sets and randomness extractors.

Thesis Supervisor: Madhu Sudan
Title: Professor

Credits

This thesis is a result of joint work with Zeev Dvir, Swastik Kopparty, Madhu Sudan and Sergey Yekhanin. I thank all my collaborators for this collaboration. The results in Chapter 3 were joint work with Swastik Kopparty and Sergey Yekhanin. A preliminary version of these results appeared in [KSY11]. A more complete version can be found in [KSY10]. The results in Chapters 2, 4, 5, 6 were joint work with Zeev Dvir, Swastik Kopparty and Madhu Sudan. A preliminary version of these results appeared in [DKSS09a], and a more complete version can be found at [DKSS09b].

During the last year of my Ph.D., I was supported by the Microsoft Research Ph.D. Fellowship. I thank Microsoft Research for its support and for hosting me for several wonderful internships.

My thesis committee consisted of Shafi Goldwasser, Silvio Micali and Madhu Sudan.

Acknowledgments

I have been extremely fortunate to have had some wonderful mentors and teachers over the years, and I have learned a great deal from their guidance and direction. I am most thankful to my advisor Madhu Sudan for his patience, guidance and wisdom. He has been a role model for me both personally and as a researcher. Many many thanks to Neeraj Kayal for all his encouragement and enthusiasm, and for sharing with me his joyous spirit of research. I am also very grateful to Professor M. Prakash who showed me how much fun mathematics can be and to Igor Pak for getting me started on research and for loads of encouragement.

Many thanks to all my coauthors Arnab Bhattacharyya, Zeev Dvir, Neeraj Kayal, Swastik Kopparty, Vsevolod F. Lev, Amir Shpilka, Madhu Sudan, Ilya Volkovich, Sergey Yekhanin. I have learned so much from them. Thanks also to Eli Ben-Sasson, Henry Cohn, Irit Dinur, Neeraj Kayal, Omer Reingold, Elad Verbin and Sergey Yekhanin for hosting me and taking care of me on so many wonderful visits.

A big thanks to all my friends in the department for making all my years at MIT so happy and memorable: Varun Kanade, Brendan Juba, Arnab Bhattacharyya, Ben Rossman, Elena Grigorescu, Deepti Bhatnagar, Jelani Nelson, Khanh Do Ba, Debmalya Panigrahi, and so many others, thanks so much! A special thanks to Gireeja and Sukhada for all the wonderful moments we have shared together.

Most of all thanks to my husband and my dearest friend Swastik for being my support and my inspiration through it all. None of this would have been possible without you. To my beautiful new family, Swara, Mum and Pop, thank you for all your warmth and love and affection. I feel truly blessed to be a part of you. However hard I try, no words can possibly describe what I owe to Ma, Papa, Bhaiya, Bhabhi, Var and Swastik who mean everything to me. Thank you for being there for me always.

For

Ma and Papa

Contents

Chapter 1

Introduction

1.1 Polynomials and their Applications

In the last few decades, algebraic techniques have played a central role in theoretical computer science. Right from their appearance in property testing and in the design of proof systems, which eventually led to some of the major recent breakthroughs in complexity theory such as IP = PSPACE and the PCP theorem, to their use in the construction of objects with interesting combinatorial and pseudorandom properties, such as error correcting codes, pseudorandom generators and randomness extractors, polynomials have played an influential role across a broad spectrum of areas in computer science.

Many of these applications actually rely on very simple and elementary properties of polynomials. One especially noteworthy property that has found application in surprisingly versatile situations is that nonzero low degree polynomials can evaluate to zero on very few points. Equivalently, two distinct low degree polynomials cannot take the same value at too many points - otherwise, their difference, a nonzero low degree polynomial, would be zero too often.

In this thesis, we introduce novel algebraic techniques that extend some of the earlier methods by also effectively utilizing the information contained in the *derivatives* of

the polynomials. Using these methods, we get improved results for several on the questions mentioned above, such as the design of efficient error correcting codes, tight analyses for Kakeya sets, improved randomness mergers and extractors, and efficient decoding of Reed-Solomon codes.

Before describing the techniques, we first discuss some basic properties of polynomials and give a flavour of how such properties have been so useful in computer science through the example of error correcting codes.

As mentioned before, the sparsity of roots of polynomials plays a prominent role in several of the results that we will be discussing. This bound on the number of roots, also known as the Schwartz-Zippel Lemma, states that for any finite subset S of the underlying field, the number of points in S^n at which a non-zero degree d polynomial vanishes is at most $d \cdot |S|^{n-1}$.

Schwartz-Zippel Lemma: *Let $\mathbb{F}$ be a field, and let $\mathbb{F}[X_1, \ldots, X_n]$ denote the ring of n variate polynomials over $\mathbb{F}$. Let S be a finite subset of $\mathbb{F}$. Then if P is a nonzero polynomial of degree d in $\mathbb{F}[X_1, \ldots, X_n]$,* $\Pr_{\mathbf{a} \in S^n}[P(\mathbf{a}) = 0] \leq d/|S|$.

The univariate version of the Schwartz-Zippel Lemma, which asserts that a nonzero degree d *univariate* polynomial can evaluate to zero at at most d locations, is a well known result. The multivariate analog is a simple (but very useful) generalization.

There are several other simple but important facts about polynomials that are used often. For instance, the linear relation between coefficients of the monomials of a polynomial P, and values of P at points in the domain is an extremely useful property. Given evaluations of a polynomial at a large enough set of points, it is possible to interpolate and recover the coefficients of the underlying polynomial using simple linear algebra. The property that evaluations of low degree multivariate polynomials look like low degree univariate polynomials when viewing them restricted to "lines" or other low degree curves is another (easily verifiable) property that we use several times.

Error correcting codes are a great example of how some of these properties of polynomials come together to give some beautiful results. We now briefly describe a few

of these codes. We then go on to discuss a surprising and elegant application of polynomials for *analyzing* vector spaces that have some algebraic structure, a method now known as the "polynomial method" in computer science. These applications are especially relevant to us. In the rest of the thesis we build upon them, strengthen some of them and provide extensions.

1.1.1 Application to Error Correcting Codes

Error correcting codes give a method to encode k symbol messages into n symbol codewords so that any two distinct messages get mapped to codewords that are "far" in Hamming distance. Thus, even after a small constant fraction of the symbols of any codeword get corrupted, the corrupted codeword still contains enough information to recover the original message. The Reed-Solomon codes are an extremely elegant and important example of such codes that are based on evaluations of univariate polynomials.

Reed-Solomon Codes Let $\mathbb{F}_q$ be a finite field and S be a subset of points in $\mathbb{F}_q$. Each codeword of the Reed-Solomon code consists of evaluations of some low degree polynomial in $\mathbb{F}_q[X]$ at the points in S, where the degree of the polynomial is much smaller than the size of S. The univariate Schwartz-Zippel Lemma immediately implies that any two distinct codewords of the Reed-Muller code differ in a large fraction of the locations. Reed-Solomon codes are one of the simplest examples of efficient error correcting codes that have found several important applications in theoretical computer science as well as in data storage and data transmission.

Locally decodable codes are error-correcting codes, that in addition to having all the properties of regular error correcting codes, also admit extremely efficient decoding algorithms. Even after a constant fraction of the bits of the codeword get corrupted, any symbol of the original message can be recovered by only looking at $q(k)$ bits of the corrupted codeword for some small sublinear function q. Locally

decodable codes have been implicitly studied in coding theory for a very long time, starting with Reed's "majority-logic decoder" for binary Reed-Muller codes. In theoretical computer science, locally decodable codes (and in particular, locally decodable codes based on multivariate polynomials) have played an important part in the Proof-Checking Revolution of the early 90s as well as in other fundamental results in complexity theory and cryptography. Locally decodable codes were first formally defined by Katz and Trevisan [KT00], and since then, the quest for understanding locally decodable codes has generated many developments. Most of the early constructions of locally decodable codes were based on classical Reed-Muller codes that we now describe.

Reed-Muller Codes Let $\mathbb{F}_q$ be a finite field and S be a subset of points in $\mathbb{F}_q$. Each codeword of the Reed-Muller code consists of evaluations of some low degree polynomial in $\mathbb{F}_q[X_1, \ldots, X_m]$ at the points in S, where the degree of the polynomial is much smaller than the size of S. The Schwartz-Zippel Lemma immediately implies that any two distinct codewords of the Reed-Muller code differ in a large fraction of the locations. The decoder recovers the value of the unknown polynomial at a point by picking a random line/low-dimensional space passing through the point, querying all the points on that line/low-dimensional space, and decoding it using noisy polynomial interpolation. This family of codes is remarkably versatile. It yields locally decodable codes of all possible query complexities, (i.e., one can choose the query-complexity t to be an arbitrary non-decreasing function of k) that tolerate a constant fraction of errors (with some tradeoff with the rate of the code).

1.1.2 Applications to the Polynomial Method

In addition to being a useful tool in the construction of objects with interesting combinatorial properties, in recent years, somewhat strikingly, polynomials have also been found to be a powerful tool in the *analysis* of combinatorial parameters of subsets of vector spaces that are "algebraically nice". This algebraic method of analysis, which

we refer to as as the *polynomial method* (of combinatorics), proceeds in three steps: Given the subset K satisfying the algebraic conditions, one first constructs a non-zero low-degree polynomial that vanishes on K. Next, one uses the algebraic conditions on K to show that the polynomial vanishes at other points outside K as well. Finally, one uses the fact that the polynomial is zero too often to derive bounds on the combinatorial parameters of interest. The polynomial method has seen utility in the computer science literature in works on "list-decoding" starting with Sudan [Sud97] and subsequent works. Recently the method has been applied to analyze "extractors" by Guruswami, Umans, and Vadhan [GUV07]. Most relevant to this thesis are its applications to lower bound the cardinality of "Kakeya sets" by Dvir [Dvi08], and the subsequent constructions of "mergers" by Dvir and Wigderson [DW08].

1.1.3 Some Limitations of Previous Techniques

In many of the above applications polynomials play a crucial role, and the property that a low degree polynomial cannot evaluate to zero too often is used repeatedly. An important requirement for making the arguments go through is that the domain, or underlying field on which the polynomials are evaluated, needs to have size much larger than the degree of the polynomials being used, or else the Schwartz-Zippel Lemma does not say anything meaningful. Indeed it is possible for *high* degree polynomials to be zero everywhere. Since only low degree polynomials are hence useful, this often limits the extent to which polynomials find their utility, and there is much scope for improvements in various parameters of interest.

For instance, in the earlier mentioned constructions of error correcting codes (and in particular the Reed-Muller codes), we were limited to using polynomials of degree smaller than the field size, which in turn resulted in the "rate" (i.e., the ratio k/n) of the corresponding codes to decay with m, the number of variables in the polynomials being used. This ends up hurting the tradeoff between the rate and the locality (number of queries) of the decoding algorithms, and all earlier constructions of codes that admitted local decoding algorithms with locality $k^{1/c}$ had rate $2^{-O(c)}$; and no

code with non-trivial locality ($q = o(k)$) and rate $> 1/2$ was known. Most earlier constructions were polynomial-based, and the "low degree" restriction was the main bottleneck for the low rate.

In this thesis we improve on several of the results mentioned above by a more powerful application of polynomials that also takes into account the information contained in the evaluations of the *derivatives* of the polynomials. This encodes information about the *higher multiplcity vanishings* of polynomials at points in the domain. We show that polynomials of not-too-high degree cannot vanish with high multiplicity at a lot of points. This allows us to break free from the "low degree" constraint, and in several of the applications we can deal with polynomials of degree independent of the field size. We call this technique the *method of multiplicities*, and we elaborate on it in the next section.

1.2 The Method of Multiplicities

The method of multiplicities is a name we have chosen for the techniques in this thesis that use evaluations of polynomials along with their derivatives to give strengthened results for various problems that were earlier tackled using evaluations of polynomials alone. As in the usual analytic notion of a polynomial having a high multiplicity zero at a point, we say that a polynomial P vanishes at a point $\mathbf{a}$ with multiplicity m, if the polynomial and all its derivatives up to order $m-1$ vanish at $\mathbf{a}$. Similarly we say that two polynomials agree with multiplicity m at a point $\mathbf{a}$ if the values of the polynomials as well as their derivatives up to order $m-1$ agree at $\mathbf{a}$. Thus evaluating a polynomial and its derivatives at a set of points is like giving high multiplicity information of the polynomial at those points. We show that in several problems of interest, this high multiplicity information can be used effectively to improve upon earlier results.

One of the main reasons we seems to profit from considering high multiplicity information is that this information allows us to deal with much higher degree polynomials

than we could before. For instance, the Schwartz-Zippel Lemma implies that two n variate polynomials of degree d where d is less than the field size cannot have the same evaluations at all points in $\mathbb{F}^n$. However this is not true for $d \geq |\mathbb{F}|$; for polynomials of degree larger than the field size, the set of evaluations does not uniquely identify the polynomial. We extend the Schwartz-Zippel Lemma to show that if one provides higher multiplicity information of the polynomials, then this however does suffice to identify the polynomial. We show that two n-variate polynomials of degree d, where d is less than $m \times |\mathbb{F}|$, cannot have the same *multiplicity* m evaluations at all points in $\mathbb{F}^n$. We formally set up notation and make everything precise with proofs in Chapter 2. Below we more precisely state this extension of the Schwartz-Zippel Lemma.

Multiplicity enhanced Schwartz-Zippel Lemma: *Let $\mathbb{F}$ be a field, and let $\mathbb{F}[X_1, \ldots, X_n]$ denote the ring of n variate polynomials over $\mathbb{F}$. Let S be a finite subset of $\mathbb{F}$. Then if P is a nonzero polynomial of degree d in $\mathbb{F}[X_1, \ldots, X_n]$,* $\mathbb{E}_{\mathbf{a} \in S^n}[\mathrm{mult}(P, (\mathbf{a}))] \leq d/|S|$.

The notions of derivative and multiplicity are not new to computer science, and they have played an important role in several prior works in coding theory and theoretical computer science. See Section 1.4 for a summary of some of the prior works using such ideas. The results in this thesis add to this body of work, and strengthen many of the tools and techniques used in prior works to obtain nearly tight analyses for several problems. We now discuss some of these applications of the method of multiplicities to obtain improved constructions of error correcting codes, and a strengthened version of the polynomial method which lets us obtain qualitatively tighter analyses of Kakeya sets, error correcting codes and randomness extractors.

1.3 Main Contributions of this Thesis

In this section, we describe the two main incarnations in which we use the method of multiplicities. Though there are many ideas that are common to both, at a high level,

both applications use polynomials and derivatives in very different ways. In the first application, derivatives are used to give improved *constructions* of polynomial based error correcting codes. In the second application, we use derivatives and multiplicities to give significantly tighter *analyses* of certain combinatorial parameters of subsets of vector spaces, improving upon earlier methods of analysis that used just polynomials.

Multiplicity Codes We apply the method of multiplicities to construct a new and natural family of locally decodable codes that achieve significantly better tradeoffs and flexibility in the rate and minimum distance from traditional polynomial based (Reed-Muller) codes. These codes, which we call multiplicity codes, are based on evaluating *high degree* multivariate polynomials and their *derivatives*. Typically, when we use polynomial based codes, for the code to have any distance, the degrees of the polynomials need to be smaller than the field size. This causes the rate of the codes to degrade exponentially with the number of variables in the polynomials being used. We manage to work with much higher degree polynomials (and thus get significantly improved rates), and we compensate for the loss in distance by also evaluating the derivatives of the polynomials. Thus, for the first time, we are able to construct codes which achieve very high rates (arbitrarily close to 1) with strong local decoding properties ($q(k) = k^{\epsilon}$ for arbitrarily small ϵ), thereby giving new performance tradeoffs between the rate and locality of decoding.

More formally,

- We show that for every $\epsilon > 0, \alpha > 0$, and for infinitely many k, there exists a code which encodes k-bit messages with rate $1 - \alpha$, and is locally decodable from some constant fraction of errors using $O(k^{\epsilon})$ time and queries.

Multiplicity enhanced polynomial method We use the method of multiplicities to extend the *polynomial method*, and derive tighter bounds on several combinatorial parameters of interest (that we list below). Recall that the polynomial method is used to analyze combinatorial parameters of subsets of vector spaces that have some nice

algebraic structure, and it proceeds in three steps: Given the subset K satisfying the algebraic conditions, one first constructs a non-zero low-degree polynomial that vanishes on K. Next, one uses the algebraic conditions on K to show that the polynomial vanishes at other points outside K as well. Finally, one uses the fact that the polynomial is zero too often to derive bounds on the combinatorial parameters of interest. In our extension, we construct a polynomial that vanishes with *high multiplicity* on every point of K. But then, unlike in previous works, we will augment the second step and show that under appropriate conditions, the interpolating polynomial vanishes *with high multiplicity* outside the set as well. This novelty will lead to significantly tighter analyses of the following results, which are of interest in combinatorics and randomness extraction.

- We show that every Kakeya set in $\mathbb{F}_q^n$, the n-dimensional vector space over the finite field on q elements, must be of size at least $q^n/2^n$. This bound is tight to within a $2 + o(1)$ factor for every n as $q \to \infty$.

- We give improved "randomness mergers": Mergers are seeded functions that take as input ℓ (possibly correlated) random variables in $\{0,1\}^N$ and a short random seed and output a single random variable in $\{0,1\}^N$ that is statistically close to having entropy $(1-\delta)\cdot N$ when one of the ℓ input variables is distributed uniformly. The seed we require is only $(1/\delta)\cdot \log \ell$-bits long, which significantly improves upon previous construction of mergers.

- We give improved randomness extractors, based on our improved mergers. Specifically, we show how to construct randomness extractors that use logarithmic length seeds while extracting $1 - o(1)$ fraction of the min-entropy of a weak random source. Previous results could extract only a constant fraction of the entropy while maintaining logarithmic seed length.

- We re-derive, algebraically, a known bound on the list-size in the list-decoding of Reed-Solomon codes.

In all the results mentioned above we improve upon earlier applications of polynomials by now instead using the derivatives of polynomials as well (and considering high multiplicity vanishings of polynomials). The notion of using derivatives and multiplicities is not new to computer science however. In the next section, we summarize some of the results which utilized such techniques. In this thesis, we add to this body of work, and introduce several new tools, in particular the multiplicity enhanced Schwartz-Zippel Lemma, that help us to utilize the power of multiplicities more effectively.

1.4 Prior Applications of Derivatives and Multiplicities:

The notions of derivative and multiplicity are not new to computer science, and they have played a critical role in several prior works in coding theory and theoretical computer science. Multiplicities have been used in conjunction with the "polynomial method" in a number of contexts in recent years [GS99, PV05, GR08]; these ideas are a precursor to our "multiplicity enhanced polynomial method". Rozenbloom and Tsfasman [RT97] 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 [Xin03] considers the space of differentials on an algebraic curve to prove the existence of error-correcting codes above the Tsfasman-Vladut-Zink bound. Woodruff and Yekhanin [WY05] use evaluations of polynomials and their derivatives to construct private information retrieval schemes with improved communication complexity. 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[1].

We now give a more detailed discussion of how the method of multiplicities leads

[1] Some restrictions apply.

to the mentioned results. In Section 1.5 we further elaborate on our contribution of multiplicity codes and in Section 1.6 we elaborate on our multiplicity enhanced polynomial method.

1.5 The Method of Multiplicities and Error Correcting Codes

Classical error-correcting codes allow one to encode a k-bit message $\mathbf{x}$ into an n-bit codeword $C(\mathbf{x})$, in such a way that $\mathbf{x}$ can still be recovered even if $C(\mathbf{x})$ gets corrupted in a number of coordinates. The traditional way to recover information about $\mathbf{x}$ given access to a corrupted version of $C(\mathbf{x})$ is to run a decoder for C, which would read and process the entire corrupted codeword, and then recover the entire original message $\mathbf{x}$. Suppose that one is only interested in recovering a single bit or a few bits of $\mathbf{x}$. In this case, codes with more efficient decoding schemes are possible, allowing one to read only a small number of code positions. Such codes are known as Locally Decodable Codes (LDCs). Locally decodable codes allow reconstruction of an arbitrary bit $\mathbf{x}_i$, by looking only at $t \ll k$ randomly chosen coordinates of (a possibly corrupted) $C(\mathbf{x})$.

The main parameters of a locally decodable code that measure its utility are the codeword length n (as a function of the message length k) and the query complexity of local decoding. The length 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 a (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 cannot minimize the codeword length and the query complexity simultaneously; there is a trade-off. On one end of the spectrum we have LDCs with the codeword length close to the message length, decodable with somewhat large query complexity. Such codes are useful for data storage and transmission. On the other end we have LDCs where the query complexity is a small constant but the codeword length is large compared to the message length. Such codes find applications

in complexity theory and cryptography. The true shape of the trade-off between the codeword length and the query complexity of LDCs is not known. Determining it is a major open problem (see [Yek10] for a recent survey of the LDC literature).

While most prior work focuses on the low query (and even constant query) regime, in this work we will look at the other extreme and consider the setting of locally decodable codes with very low redundancy, which may be of even greater practical interest. More precisely, we will be interested in minimizing the query complexity of local decoding for codes of large *rate* (defined as the ratio k/n, where the code encodes k bits into n bits). For codes of rate $> 1/2$, it was unknown how to get any nontrivial local decoding whatsoever. For smaller rates, it was known how to construct codes (in fact, the classical Reed-Muller codes based on evaluating multivariate polynomials have this property) which admit local decoding with $O(k^\epsilon)$ queries and time, at the cost of reducing the rate to $\epsilon^{\Omega(1/\epsilon)}$. In practical applications of coding theory to data storage and transmission, the rate of encoding has always been paramount; using codes of very small rate translates into increasing the storage required or transmission time manifold, and is unacceptable for most applications.

In this thesis, we introduce a new and natural family of locally decodable codes, which achieve high rates while admitting local decoding with low query complexity. These codes, which we call multiplicity codes, are based on evaluating multivariate polynomials and their derivatives. They inherit the local-decodability of the traditional multivariate polynomial codes, while achieving better tradeoffs and flexibility in the rate and minimum distance. Using multiplicity codes, we prove (see Theorem 3.1.4) that it is possible to have codes that simultaneously have (a) rate approaching 1, and (b) allow for local decoding with arbitrary polynomially-small time and query complexity.

Main Theorem (informal): *For every $\epsilon > 0, \alpha > 0$, and for infinitely many k, there exists a code which encodes k-bit messages with rate $1 - \alpha$, and is locally decodable from some constant fraction of errors using $O(k^\epsilon)$ time and queries.*

1.5.1 Previous Work on Locally Decodable Codes

Locally decodable codes have been implicitly studied in coding theory for a very long time, starting with Reed's "majority-logic decoder" for binary Reed-Muller codes [Ree54]. In theoretical computer science, locally decodable codes (and in particular, locally decodable codes based on multivariate polynomials) have played an important part (again implicitly) in the Proof-Checking Revolution of the early 90s [BF90, Lip90, LFKN92, Sha92, BFLS91, BFL91, AS98, ALM+98] as well as in other fundamental results in complexity theory [BFNW93, IW97, AS03, STV99, SU05].

Locally decodable codes were first formally defined by Katz and Trevisan [KT00] (see also [STV99]). Since then, the quest for understanding locally decodable codes has generated many developments. Most of the previous work on LDCs has focussed on local decoding with a constant number of queries. For a long time, it was generally believed that for decoding with constantly many queries, a k bit message must be encoded into at least $\exp(k^\alpha)$ bits, for constant $\alpha > 0$. Recently, in a surprising sequence of works [Yek08, Rag07, Efr09, IS10, DGY10, BET10, CFL+10] this was shown to be soundly false; today we know constant query locally decodable codes which encode k bits into as few as $\exp(\exp(\log^\alpha(k)))$ bits for constant $\alpha > 0$.

There has also been considerable work [KT00, KdW04, GKST02, WdW05, Woo07, DJK+02, Oba02] on the problem of proving lower bounds on the length of locally decodable codes. In particular, it is known [KT00] that for codes of constant rate, local decoding requires at least $\Omega(\log k)$ queries. For codes locally decodable with $\omega(\log k)$ queries, no nontrivial lower bound on the length on the code is known. For error-correction with $O(k^\epsilon)$ queries, Dvir [Dvi10] recently conjectured a lower bound on the length of some closely related objects called *locally self-correctable codes*. Precisely, the conjecture of [Dvi10] states that for every field $\mathbb{F}$, there exist positive constants α and ϵ such that there are no linear codes over $\mathbb{F}$ of length n, rate $1 - \alpha$ and locally self-correctable with query complexity $O(n^\epsilon)$ from a certain sub-constant fraction of errors. Dvir [Dvi10] then showed that establishing this conjecture would

yield progress on some well-known open questions in arithmetic circuit complexity.

Our results refute Dvir's conjecture over finite fields; using multiplicity codes, we show that for arbitrary $\alpha, \epsilon > 0$, for every finite field $\mathbb{F}$, for infinitely many n, there is a linear code over $\mathbb{F}$ of length n with rate $1 - \alpha$, which is locally self-correctable from even a constant fraction of errors with $O(n^\epsilon)$ queries[2].

1.5.2 Multiplicity codes

We now give a quick introduction to multiplicity codes and demonstrate the principles on which they are based.

To minimize extraneous factors and for ease of exposition, in this subsection we will deal with the problem of constructing "locally self-correctable codes" over "large alphabets", which we now define. We have a set Σ (the "alphabet"), and we want to construct a subset $\mathcal{C}$ (the "code") of Σ^n, of size $|\Sigma|^k$ (we call k the "message length"), with the following local self-correction property: given access to any $r \in \Sigma^n$ which is close to some codeword $c \in \mathcal{C}$, and given $i \in [n]$, it is possible to make few queries to the coordinates of r, and with high probability output c_i. The goal is to construct such a subset $\mathcal{C}$ with rate k/n large. Note that this differs from the notion of locally decodable code in that we seek to recover a coordinate of the nearby codeword c, not of the original message which encodes to c. We also do not require that Σ has size 2, which is what the Main Theorem mentioned earlier refers to. Translating from local self-correctability over large alphabets to local decodability over small alphabets is a standard transformation.

Our plan is as follows. We will first recall an example of the classical Reed-Muller codes based on bivariate polynomials and why it is locally self-correctable. We will then introduce the simplest example of a multiplicity code based on bivariate polynomials, which has improved rate, and see how to locally self-correct it with essentially

[2] [Dvi10] contains two conjectures; which are called the "strong conjecture" and the "weak conjecture". We refute only the strong conjecture. The weak conjecture, which has weaker implications for questions related to arithmetic circuit complexity, remains open.

the same query complexity. Finally, we mention how general multiplicity codes are defined and some of the ideas that go into locally self-correcting them.

Bivariate Reed-Muller codes: Let q be a prime power, let $\delta > 0$ and let $d = (1-\delta)q$. The Reed Muller code of degree d bivariate polynomials over $\mathbb{F}_q$ (the finite field of cardinality q) is the code defined as follows. The coordinates of the code are indexed by elements of $\mathbb{F}_q^2$, and so $n = q^2$. The codewords are indexed by bivariate polynomials of degree at most d over $\mathbb{F}_q$. The codeword corresponding the polynomial $P(X,Y)$ is the vector

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

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 $\binom{d+1}{2}$ monomials, and so the message length $k = \binom{d+1}{2}$. Thus the rate of this code is $\binom{d+1}{2}/q^2 \approx (1-\delta)^2/2$. Notice that this code cannot have rate more than $1/2$.

Local Self-Correction of Reed-Muller codes: Given a received word $r \in (\mathbb{F}_q)^{q^2}$ such that r is close in Hamming distance to the codeword corresponding to $P(X,Y)$, let us recall how the classical local self-correction algorithm works. Given a coordinate $\mathbf{a} \in \mathbb{F}_q^2$, we want to recover the "corrected" symbol at coordinate $\mathbf{a}$, namely $P(\mathbf{a})$. The algorithm picks a random direction $\mathbf{b} \in \mathbb{F}_q^2$ and looks at the restriction of r to coordinates in the line $L = \{\mathbf{a} + \mathbf{b}t \mid t \in \mathbb{F}_q\}$. With high probability over the choice of $\mathbf{b}$, r and $C(P)$ will agree on many positions of L. Now $C(P)|_L$ is simply the vector consisting of evaluations of the univariate polynomial $Q(T) = P(\mathbf{a} + \mathbf{b}T) \in \mathbb{F}_q[T]$, which is of degree $\leq d$. Thus $r|_L$ gives us q "noisy" evaluations of a polynomial $Q(T)$ of degree $\leq (1-\delta) \cdot q$; this enables us to recover $Q(T)$. Evaluating $Q(T)$ at $T = 0$ gives us $P(\mathbf{a})$, as desired. Notice that this decoding algorithm makes q queries, which is $O(k^{1/2})$.

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 self-correctable with only a constant factor more queries.

Let q be a prime power, let $\delta > 0$ and let $d = 2(1-\delta)q$ (which is twice what it was in the Reed-Muller example). The multiplicity code of **order 2** 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 $P(X,Y)$ is the vector

$$C(P) = \langle (P(\mathbf{a}), \frac{\partial P}{\partial X}(\mathbf{a}), \frac{\partial P}{\partial Y}(\mathbf{a})) \rangle_{(\mathbf{a}) \in \mathbb{F}_q^2} \in (\mathbb{F}_q^3)^{q^2}.$$

In words, the $\mathbf{a}$ coordinate consists of the evaluation of P and its partial derivatives $\frac{\partial P}{\partial X}$ and $\frac{\partial P}{\partial Y}$ at $\mathbf{a}$. Because two distinct polynomials of degree at most d can agree with multiplicity 2 on at most $d/2q$-fraction of the points in $\mathbb{F}_q^2$, 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 $\binom{d+1}{2}/3$. Thus the rate of this code is $(\binom{d+1}{2}/3)/q^2 \approx 2(1-\delta)^2/3$.

Summarizing the differences between this multiplicity code with the Reed-Muller code described earlier: (a) instead of polynomials of degree $(1-\delta)q$, we consider polynomials of degree double of that, (b) instead of evaluating the polynomials, we take their "order-2" evaluation. This yields a code with the same distance, while the rate improved from $< 1/2$ to nearly $2/3$.

Local Self-Correction of Multiplicity codes: Given a received word $r \in (\mathbb{F}_q^3)^{q^2}$ such that r is close in Hamming distance to the codeword corresponding to $P(X,Y)$, we will show how to locally self-correct. Given a point $\mathbf{a} \in \mathbb{F}_q^2$, we want to recover the "corrected" symbol at coordinate $\mathbf{a}$, namely $(P(\mathbf{a}), \frac{\partial P}{\partial X}(\mathbf{a}), \frac{\partial P}{\partial Y}(\mathbf{a}))$. Again, the algorithm picks a random direction $\mathbf{b} = (b_1, b_2) \in \mathbb{F}_q^2$ and looks at the restriction of r to coordinates in the line $L = \{\mathbf{a} + \mathbf{b}t \mid t \in \mathbb{F}_q\}$. With high probability over the choice of $\mathbf{b}$, we will have that $r|_L$ and $C(P)|_L$ agree in many locations. Our intermediate goal will be to recover the univariate polynomial[3] $Q(T) = P(\mathbf{a} + \mathbf{b}T)$. The impor-

[3]Unlike in the Reed-Muller case, here there is a distinction between recovering $Q(T)$ and recovering $C(P)|_L$. It turns out that recovering $C(P)|_L$ given only $r|_L$ is impossible.

tant observation is that for every $t \in \mathbb{F}_q$, the $\mathbf{a} + \mathbf{b}t$ coordinate of $C(P)$ completely determines both the value and the 1st derivative of the univariate polynomial $Q(T)$ at the point t; indeed, by the chain rule we have:

$$(Q(t), \frac{\partial Q}{\partial T}(t)) = (P(\mathbf{a} + \mathbf{b}t), b_1 \frac{\partial P}{\partial X}(\mathbf{a} + \mathbf{b}t) + b_2 \frac{\partial P}{\partial Y}(\mathbf{a} + \mathbf{b}t)).$$

Thus our knowledge of $r|_L$ gives us access to q "noisy" evaluations of the polynomial $Q(T)$ and its derivative $\frac{\partial Q}{\partial T}(T)$, where $Q(T)$ is of degree $\leq 2(1-\delta)q$. It turns out that this is enough to recover the polynomial $Q(T)$. Evaluating $Q(T)$ at $T = 0$ gives us $P(\mathbf{a})$. Evaluating the derivative $\frac{\partial Q}{\partial T}(T)$ at $T = 0$ gives us the directional derivative of P at $\mathbf{a}$ in the direction $\mathbf{b}$ (which equals $b_1 \frac{\partial P}{\partial X}(\mathbf{a}) + b_2 \frac{\partial P}{\partial Y}(\mathbf{a})$). We have clearly progressed towards our goal of computing the tuple $(P(\mathbf{a}), \frac{\partial P}{\partial X}(\mathbf{a}), \frac{\partial P}{\partial Y}(\mathbf{a}))$, but we are not yet there. The final observation is that if we pick another direction $\mathbf{b}'$, and repeat the above process to recover the directional derivative of P at $\mathbf{a}$ in direction $\mathbf{b}'$, then the two directional derivatives of P at $\mathbf{a}$ in directions $\mathbf{b}, \mathbf{b}'$ together suffice to recover $\frac{\partial P}{\partial X}(\mathbf{a})$ and $\frac{\partial P}{\partial Y}(\mathbf{a})$, as desired. This algorithm makes $2q$ queries, which is $O(k^{1/2})$.

General Multiplicity codes: The basic example of a multiplicity code above already achieves rate $R > 1/2$ while allowing local decoding with sublinear query complexity (which was not known before). To get codes of rate approaching 1, we modify the above example by considering evaluations of all derivatives of P up to an even higher order. In order to locally recover the higher-order derivatives of P at a point $\mathbf{a}$, the decoding algorithm will pick many random lines passing through $\mathbf{a}$, try to recover the restriction of P 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 m. The local decoding algorithm for this case, in order to locally recover at a point $\mathbf{a} \in \mathbb{F}_q^m$, decodes by picking random lines passing through $\mathbf{a}$; the reduced query complexity occurs because lines (with only q points) are now much smaller relative to a higher dimensional space $\mathbb{F}_q^m$. Increasing both the maximum order of derivative taken and the number of variables simultaneously yields

multiplicity codes with the desired rate and local decodability.

In Section 3.3, we present our local self-correction algorithm, which implements the plan outlined above, along with an extra "robustification" so that the fraction of errors which can be recovered from is a constant fraction of the distance of the code. We also show how the algorithm can be made to run in sublinear time (almost as small as the query complexity).

1.6 The Multiplicity Enhanced Polynomial Method

We use the method of multiplicities to improve on an algebraic method that has lately been applied, quite effectively, to analyze combinatorial parameters of subsets of vector spaces that satisfy some given algebraic/geometric conditions. This technique, which we refer to as as the *polynomial method* (of combinatorics), proceeds in three steps: Given a subset K satisfying the algebraic conditions, one first constructs a non-zero low-degree polynomial that vanishes on K. Next, one uses the algebraic conditions on K to show that the polynomial vanishes at other points outside K as well. Finally, one uses the fact that the polynomial is zero too often to derive bounds on the combinatorial parameters of interest. The polynomial method has seen utility in the computer science literature in works on "list-decoding" starting with Sudan [Sud97] and subsequent works. Recently the method has been applied to analyze "extractors" by Guruswami, Umans, and Vadhan [GUV07]. Most relevant to this current thesis are its applications to lower bound the cardinality of "Kakeya sets" by Dvir [Dvi08], and the subsequent constructions of "mergers" and "extractors" by Dvir and Wigderson [DW08]. (We will elaborate on some of these results shortly.)

The notion of multiplicities has been used in the past in conjunction with the polynomial method. The way it was used was the following: one constructed polynomials that vanished with *high multiplicity* on the subset K. This requirement often forced one to use polynomials of higher degree than in the polynomial method, but it gained in the second step by using the high multiplicity of zeroes to conclude "more easily"

that the polynomial was zero at other points. This typically lead to a tighter analysis of the combinatorial parameters of interest. This use of multiplicities has been applied widely in list-decoding starting with the work of Guruswami and Sudan [GS99] and continuing through many subsequent works, most significantly in the works of Parvaresh and Vardy [PV05] and Guruswami and Rudra [GR08] leading to rate-optimal list-decodable codes. Very recently this method was also applied to improve the lower bounds on the size of "Kakeya sets" by Saraf and Sudan [SS08].

In this thesis we provide an extension to this method, that we call the *extended method of multiplicities*, which develops this method (hopefully) fully to derive even tighter bounds on the combinatorial parameters. In our extension, we start as in the earlier method by constructing a polynomial that vanishes with high multiplicity on every point of K. But then we extend the second step where we exploit the algebraic conditions to show that the polynomial vanishes with *high multiplicity* on some points outside K as well. Finally we extend the third step to show that this gives better bounds on the combinatorial parameters of interest.

By these extensions we derive nearly optimal lower bounds on the size of Kakeya sets and qualitatively improved analysis of mergers leading to new extractor constructions. We also re-derive algebraically a known bound on the list-size in the list-decoding of Reed-Solomon codes. We describe these contributions in detail next, before going on to describe some of the technical observations used to derive the extended method of multiplicities (which we believe are of independent interest).

1.6.1 Kakeya Sets over Finite Fields

Let $\mathbb{F}_q$ denote the finite field of cardinality q. A set $K \subseteq \mathbb{F}_q^n$ is said to be a *Kakeya set* if it "contains a line in every direction". In other words, for every "direction" $\mathbf{b} \in \mathbb{F}_q^n$ there should exist an "offset" $\mathbf{a} \in \mathbb{F}_q^n$ such that the "line" through $\mathbf{a}$ in direction $\mathbf{b}$, i.e., the set $\{\mathbf{a} + t\mathbf{b} | t \in \mathbb{F}_q\}$, is contained in K. A question of interest in combinatorics/algebra/geometry, posed originally by Wolff [Wol99], is: "What is the size of the smallest Kakeya set, for a given choice of q and n?"

The trivial upper bound on the size of a Kakeya set is q^n and this can be improved to roughly $\frac{1}{2^{n-1}}q^n$ (precisely the bound is $\frac{1}{2^{n-1}}q^n + O(q^{n-1})$, see [SS08] for a proof of this bound due to Dvir). An almost trivial lower bound is $q^{n/2}$ (every Kakeya set "contains" at least q^n lines, but there are at most $|K|^2$ lines that intersect K at least twice). Till recently even the exponent of q was not known precisely (see [Dvi08] for details of work prior to 2008). This changed with the result of [Dvi08] (combined with an observation of Alon and Tao) who showed that for every n, $|K| \geq c_n q^n$, for some constant c_n depending only on n.

Subsequently the work [SS08] explored the growth of the constant c_n as a function of n. The result of [Dvi08] shows that $c_n \geq 1/n!$, and [SS08] improve this bound to show that $c_n \geq 1/(2.6)^n$. This still leaves a gap between the upper bound and the lower bound and we effectively close this gap.

Theorem 1.6.1 *If K is a Kakeya set in $\mathbb{F}_q^n$ then $|K| \geq \frac{1}{2^n}q^n$.*

Note that our bound is tight to within a $2 + o(1)$ multiplicative factor as long as $q = \omega(2^n)$ and in particular when $n = O(1)$ and $q \to \infty$.

1.6.2 Randomness Mergers and Extractors

A general quest in the computational study of randomness is the search for simple primitives that manipulate random variables to convert their randomness into more useful forms. The exact notion of utility varies with applications. The most common notion is that of "extractors" that produce an output variable that is distributed statistically close to uniformly on the range. Other notions of interest include "condensers", "dispersers" etc. One such object of study (partly because it is useful to construct extractors) is a "randomness merger". A randomness merger takes as input Λ random variables $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$, which are possibly correlated, along with a short uniformly random seed B, which is independent of $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$, and "merges" the randomness of $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$. Specifically the output of the merger should be statistically

close to a high-entropy-rate source of randomness provided at least one of the input variables $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$ is uniform.

Mergers were first introduced by Ta-Shma [TS96a] in the context of explicit constructions of extractors. A general framework was given in [TS96a] that reduces the problem of constructing good extractors into that of constructing good mergers. Subsequently, in [LRVW03], mergers were used in a more complicated manner to create extractors which were optimal to within constant factors. The mergers of [LRVW03] had a very simple algebraic structure: the output of the merger was a random linear combination of the blocks over a finite vector space. The [LRVW03] merger analysis was improved in [DS07] using the connection to the finite field Kakeya problem and the (then) state of the art results on Kakeya sets.

The new technique in [Dvi08] inspired Dvir and Wigderson [DW08] to give a very simple, algebraic, construction of a merger which can be viewed as a derandomized version of the [LRVW03] merger. They associate the domain of each random variable A_i with a vector space $\mathbb{F}_q^n$. With the Λ-tuple of random variables $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$, they associate a curve $C : \mathbb{F}_q \to \mathbb{F}_q^n$ of degree $\leq \Lambda$ which 'passes' through all the points $\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda$ (that is, the image of C contains these points). They then select a random point $u \in \mathbb{F}_q$ and output $C(u)$ as the "merged" output. They show that if $q \geq \mathrm{poly}(\Lambda \cdot n)$ then the output of the merger is statistically close to a distribution of entropy-rate arbitrarily close to 1 on $\mathbb{F}_q^n$.

While the polynomial (or at least linear) dependence of q on Λ is essential to the construction above, the requirement $q \geq \mathrm{poly}(n)$ appears only in the analysis. In our work we remove this restriction to show:

Informal Theorem [Merger]: *For every Λ, q the output of the Dvir-Wigderson merger is close to a source of entropy rate $1 - \log_q \Lambda$. In particular there exists an explicit merger for Λ sources (of arbitrary length) that outputs a source with entropy rate $1 - \delta$ and has seed length $(1/\delta) \cdot \log(\Lambda/\epsilon)$ for any error ϵ.*

The above theorem (in its more formal form given in Theorem 5.2.3) allows us to merge Λ sources using seed length which is only logarithmic in the number of sources

and has no dependence on the length of each source. Earlier constructions of mergers required the seed to depend either linearly on the number of blocks [LRVW03, Zuc07] or to depend also on the *length* of each block [DW08]. [4]

One consequence of our improved merger construction is an improved construction of extractors. Recall that a (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a deterministic function that takes any random variable X with min-entropy at least k over $\{0,1\}^n$ and an independent uniformly distributed seed $\mathsf{Y} \in \{0,1\}^d$ and converts it to the random variable $E(\mathsf{X}, \mathsf{Y})$ that is ϵ-close in statistical distance to a uniformly distributed random variable over $\{0,1\}^m$. Such an extractor is efficient if E is polynomial time computable.

A diverse collection of efficient extractors are known in the literature (see the survey [Sha02] and the more recent [GUV07, DW08] for references) and many applications have been found for explicit extractors in various research areas spanning theoretical computer science. Yet all previous constructions lost a linear fraction of the min-entropy of the source (i.e., achieved $m = (1-\epsilon)k$ for some constant $\epsilon > 0$) or used super-logarithmic seed length ($d = \omega(\log n)$). We show that our merger construction yields, by combining with several of the prior tools in the arsenal of extractor constructions, an extractor which extracts a $1 - \frac{1}{\text{polylog(n)}}$ fraction of the min-entropy of the source, while still using $O(\log n)$-length seeds. We now state our extractor result in an informal way (see Theorem 5.3.3 for the formal statement).

Informal Theorem [Extractor]: *There exists an explicit (k, ϵ)-extractor for all min-entropies k with $O(\log n)$ seed, entropy loss $O(k/\text{polylog}(n))$ and error $\epsilon = 1/\text{polylog}(n)$, where the powers in the $\text{polylog}(n)$ can be arbitrarily high constants.*

As seen in the above theorem, our extractors can handle error which is only polylogarithmic. For smaller values of ϵ there are better constructions of extractors (e.g those of [GUV07]).

[4] The result we refer to in [Zuc07, Theorem 5.1] is actually a condenser (which is stronger than a merger).

1.6.3 List-Decoding of Reed-Solomon Codes

The problem of list-decoding Reed-Solomon codes is the following: Given a sequence of points

$$(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n) \in \mathbb{F}_q \times \mathbb{F}_q,$$

and parameters k and t, find the list of all polynomials $p_1, \ldots, p_L$ of degree at most k that agree with the given set of points on t locations, i.e., for every $j \in \{1, \ldots, L\}$ the set $\{i | p_j(\alpha_i) = \beta_i\}$ has at least t elements. The associated combinatorial problem is: How large can the list size, L, be for a given choice of k, t, n, q (when maximized over all possible sets of distinct input points)?

A somewhat nonstandard, yet reasonable, interpretation of the list-decoding algorithms of [Sud97, GS99] is that they give algebraic proofs, by the polynomial method and the method of multiplicities, of known combinatorial upper bounds on the list size, when $t > \sqrt{kn}$. Their proofs *happen* also to be algorithmic and so lead to algorithms to find a list of all such polynomials.

However, the bound given on the list size in the above works does not match the best known combinatorial bound. The best known bound to date seems to be that of Cassuto and Bruck [CB04] who show that, letting $R = k/n$ and $\gamma = t/n$, if $\gamma^2 > R$, then the list size L is bounded by $O(\frac{\gamma}{\gamma^2 - R})$ (in contrast, the Johnson bound and the analysis of [GS99] gives a list size bound of $O(\frac{1}{\gamma^2 - R})$, which is asymptotically worse for, say, $\gamma = (1 + O(1))\sqrt{R}$ and R tending to 0). In Theorem 6.1.2 we recover the bound of [CB04] using our extended method of multiplicities.

1.6.4 Technique: Extended Method of Multiplicities

The common insight to all the above improvements is that the extended method of multiplicities can be applied to each problem to improve the parameters. Here we attempt to describe the technical novelties in the development of the extended method of multiplicities.

For concreteness, let us take the case of the Kakeya set problem. Given a set $K \subseteq \mathbb{F}_q^n$, the method first finds a non-zero polynomial $P \in \mathbb{F}_q[X_1, \ldots, X_n]$ that vanishes with high multiplicity m on each point of K. The next step is to prove that P vanishes with fairly high multiplicity ℓ at every point in $\mathbb{F}_q^n$ as well. This step turns out to be somewhat subtle (and is evidenced by the fact that the exact relationship between m and ℓ is not simple). Our analysis here crucially uses the fact that the (Hasse) derivatives of the polynomial P, which are the central to the notion of multiplicity of roots, are themselves polynomials, and also vanish with high multiplicity at points in K. This fact does not seem to have been needed/used in prior works and is central to ours.

A second important technical novelty arises in the final step of the method of multiplicities, where we need to conclude that if the degree of P is "small", then P must be identically zero. Unfortunately in our application the degree of P may be much larger than q (or nq, or even q^n). To prove that it is identically zero we need to use the fact that P vanishes *with high multiplicity* at every point in $\mathbb{F}_q^n$, and this requires some multiplicity-enhanced version of the standard Schwartz-Zippel lemma. We prove such a strengthening, showing that the expected multiplicity of zeroes of a degree d polynomial (even when $d \gg q$) at a random point in $\mathbb{F}_q^n$ is at most d/q (see Lemma 2.1.7). Using this lemma, we are able to derive much better benefits from the "polynomial method". Indeed we feel that this allows us to go beyond the limitations of the function space mapping $\mathbb{F}_q^n$ to $\mathbb{F}_q$ and capitalize on the full power of the polynomial ring $\mathbb{F}_q[\mathbf{X}]$.

Putting these ingredients together, the analysis of the Kakeya sets follows easily. The analysis of the mergers follows a similar path and may be viewed as a "statistical" extension of the Kakeya set analysis to "curve" based sets, i.e., here we consider sets S that have the property that for a noticeable fraction points $\mathbf{x} \in \mathbb{F}_q^n$ there exists a low-degree curve passing through $\mathbf{x}$ that has a noticeable fraction of its points in S. We prove such sets must also be large and this leads to the analysis of the Dvir-Wigderson merger.

1.7 Organization of this Thesis

In Chapter 2 we define the notion of the multiplicity of the roots of a polynomial, using the notion of the Hasse derivative. We present some basic facts about multiplicities and Hasse derivatives, and also present the multiplicity based version of the Schwartz-Zippel lemma.

In Chapter 3 we describe the full construction of high rate locally decodable codes using polynomials and derivatives, extending earlier constructions of polynomial based Reed-Muller codes. In Section 3.1, we state our main theorems on the existence of locally decodable/self-correctable codes. In Section 3.2, we formally define multiplicity codes, calculate their rate and distance, state the theorem implying their local decodability, and show how they imply the main theorems from the previous section. In Section 3.3 we give our local self-correction algorithms for multiplicity codes. Section 3.4 contains some concluding remarks.

In Chapters 4, 5 and 6 we describe how we use the multiplicity enhanced polynomial method to obtain improved results for a variety of problems. In Chapter 4 we present our lower bounds for Kakeya sets. In Chapter 5 we extend this analysis for "curves" and for "statistical" versions of the Kakeya property. This leads to our analysis of the Dvir-Wigderson merger in Section 5.2. We then show how to use our mergers to construct the novel extractors in Section 5.3. Finally, in Chapter 6, we include the algebraic proof of the list-size bounds for the list-decoding of Reed-Solomon codes.

Chapter 2

Hasse Derivatives and Multiplicities

2.1 Preliminaries

In this section we formally define the notion of "mutliplicity of zeroes" along with the companion notion of the "Hasse derivative". We also describe basic properties of these notions, concluding with the proof of the "multiplicity-enhanced version" of the Schwartz-Zippel lemma.

2.1.1 Basic Definitions

We start with some notation. We use $[n]$ to denote the set $\{1, \ldots, n\}$. For a vector $\mathbf{i} = \langle i_1, \ldots, i_n \rangle$ of non-negative integers, its *weight*, denoted $\mathrm{wt}(\mathbf{i})$, equals $\sum_{j=1}^{n} i_j$.

Let $\mathbb{F}$ be any field, and $\mathbb{F}_q$ denote the finite field of q elements. For $\mathbf{X} = \langle X_1, \ldots, X_n \rangle$, let $\mathbb{F}[\mathbf{X}]$ be the ring of polynomials in $X_1, \ldots, X_n$ with coefficients in $\mathbb{F}$. For a polynomial $P(\mathbf{X})$, we let $H_P(\mathbf{X})$ denote the homogeneous part of $P(\mathbf{X})$ of highest total degree. Specifically, if $P(\mathbf{X})$ is a polynomial of degree d, then $H_P(\mathbf{X})$ is the homogenous degree d polynomial such that $P(\mathbf{X}) - H_P(\mathbf{X})$ has degree less than d.

For a vector of non-negative integers $\mathbf{i} = \langle i_1, \ldots, i_n \rangle$, let $\mathbf{X}^{\mathbf{i}}$ denote the monomial $\prod_{j=1}^{n} X_j^{i_j} \in \mathbb{F}[\mathbf{X}]$. Note that the (total) degree of this monomial equals $\mathrm{wt}(\mathbf{i})$. For n-tuples of non-negative integers $\mathbf{i}$ and $\mathbf{j}$, we use the notation

$$\binom{\mathbf{i}}{\mathbf{j}} = \prod_{k=1}^{n} \binom{i_k}{j_k}.$$

Note that the coefficient of $\mathbf{Z}^{\mathbf{i}}\mathbf{W}^{\mathbf{r}-\mathbf{i}}$ in the expansion of $(\mathbf{Z}+\mathbf{W})^{\mathbf{r}}$ equals $\binom{\mathbf{r}}{\mathbf{i}}$.

Definition 2.1.1 ((Hasse) Derivative) *For $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ and non-negative vector $\mathbf{i}$, the $\mathbf{i}$th* (Hasse) derivative *of P, denoted $P^{(\mathbf{i})}(\mathbf{X})$, is the coefficient of $\mathbf{Z}^{\mathbf{i}}$ in the polynomial $\tilde{P}(\mathbf{X}, \mathbf{Z}) \stackrel{\text{def}}{=} P(\mathbf{X}+\mathbf{Z}) \in \mathbb{F}[\mathbf{X}, \mathbf{Z}]$.*

Thus,

$$P(\mathbf{X}+\mathbf{Z}) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{X})\mathbf{Z}^{\mathbf{i}}. \tag{2.1}$$

We are now ready to define the notion of the (zero-)multiplicity of a polynomial at any given point.

Definition 2.1.2 (Multiplicity) *For $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ and $\mathbf{a} \in \mathbb{F}^n$, the* multiplicity *of P at $\mathbf{a} \in \mathbb{F}^n$, denoted* $\mathrm{mult}(P, \mathbf{a})$*, is the largest integer M such that for every non-negative vector $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) < M$, we have $P^{(\mathbf{i})}(\mathbf{a}) = 0$ (if M may be taken arbitrarily large, we set $\mathrm{mult}(P, \mathbf{a}) = \infty$).*

Note that $\mathrm{mult}(P, \mathbf{a}) \geq 0$ for every $\mathbf{a}$. Also, $P(\mathbf{a}) = 0$ if and only if $\mathrm{mult}(P, \mathbf{a}) \geq 1$. We also note that the condition $P^{(\mathbf{i})}(\mathbf{a}) = 0$ imposes a homogenous linear constraint on the coefficients of $P(\mathbf{X})$ (for every $\mathbf{a} \in \mathbb{F}^n$. A consequence that we will use often later is the condition $\mathrm{mult}(P, \mathbf{a}) \geq M$ imposes at most $\binom{M+n-1}{n}$ homogenous linear constraints on the coefficients of $P(\mathbf{X})$, one such constraint corresponding to each vector $\mathbf{i}$ of weight less than M.

The above notations and definitions also extend naturally to a tuple

$\mathbf{P}(\mathbf{X}) = \langle P_1(\mathbf{X}), \ldots, P_m(\mathbf{X}) \rangle$ of polynomials with $P^{(\mathbf{i})} \in \mathbb{F}[\mathbf{X}]^m$ denoting the vector

$$\langle (P_1)^{(\mathbf{i})}, \ldots, (P_m)^{(\mathbf{i})} \rangle.$$

In particular, we define $\mathrm{mult}(\mathbf{P}, \mathbf{a}) = \min_{j \in [m]} \{\mathrm{mult}(P_j, \mathbf{a})\}$.

The definition of multiplicity above is similar to the standard (analytic) definition of multiplicity with the difference that the standard partial derivative has been replaced by the Hasse derivative. The Hasse derivative is also a reasonably well-studied quantity (see, for example, [HKT08, pages 144-155]) and seems to have first appeared in the CS literature (without being explicitly referred to by this name) in the work of Guruswami and Sudan [GS99]. It typically behaves like the standard derivative, but with some key differences that make it more useful/informative over finite fields. For completeness we review basic properties of the Hasse derivative and multiplicity in the following subsections.

2.1.2 Properties of Hasse Derivatives

The following proposition lists basic properties of the Hasse derivatives. Parts (1)-(3) below are the same as for the usual derivative, while Part (4) is 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 it does show 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, and this is critical for our purposes.

Proposition 2.1.3 (Basic Properties of Derivatives) *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 $P(\mathbf{X})$ is homogeneous of degree d, then either $P^{(\mathbf{i})}(\mathbf{X})$ is homogeneous of degree $d - \mathrm{wt}(i)$, or $P^{(\mathbf{i})}(\mathbf{X}) = 0$.*

3. *Either* $(H_P)^{(\mathbf{i})}(\mathbf{X}) = H_{P^{(\mathbf{i})}}(\mathbf{X})$, *or* $(H_P)^{(\mathbf{i})}(\mathbf{X}) = 0$.

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

Item 1 follows immediately from Equation (2.1).

For item 2, observe that if $P(\mathbf{X})$ is homogenous of degree d, so is $P(\mathbf{X}+\mathbf{Z})$. Thus by Equation (2.1), $P^{(\mathbf{i})}(\mathbf{X})\mathbf{Z}^{\mathbf{i}}$ must either be homogenous of degree d or be 0. Hence $P^{(\mathbf{i})}(\mathbf{X})$ is either homogeneous of degree $d - \mathrm{wt}(i)$, or 0.

For item 3, let $P(\mathbf{X}) = H_P(\mathbf{X}) + Q(\mathbf{X})$, where $P(\mathbf{X})$ is of degree d, $H_P(\mathbf{X})$ is homogenous of degree d, and $Q(\mathbf{X})$ is of degree $< d$. By item 1, $(H_P)^{(\mathbf{i})}(\mathbf{X}) = P^{(\mathbf{i})}(\mathbf{X}) - Q^{(\mathbf{i})}(\mathbf{X})$. By item 2, if $(H_P)^{(\mathbf{i})}(\mathbf{X})$ is nonzero, then it must be homogenous of degree $d - \mathrm{wt}(i)$. Hence $P^{(\mathbf{i})}(\mathbf{X}) - Q^{(\mathbf{i})}(\mathbf{X})$ is homogenous of degree $d - \mathrm{wt}(i)$, implying that $P^{(\mathbf{i})}(\mathbf{X}) - Q^{(\mathbf{i})}(\mathbf{X}) = H_{P^{(\mathbf{i})} - Q^{(\mathbf{i})}}(\mathbf{X})$. Since by item 2, the degree of $Q^{(\mathbf{i})}$ is strictly less than $d - \mathrm{wt}(i)$, it must be that $H_{P^{(\mathbf{i})} - Q^{(\mathbf{i})}}(\mathbf{X}) = H_{P^{(\mathbf{i})}}(\mathbf{X})$, and item 3 follows.

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. ■

2.1.3 Properties of Multiplicities

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

Lemma 2.1.4 (Basic Properties of multiplicities) *If* $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ *and* $\mathbf{a} \in \mathbb{F}^n$ *are such that* $\mathrm{mult}(P, \mathbf{a}) = m$*, then* $\mathrm{mult}(P^{(\mathbf{i})}, \mathbf{a}) \geq m - \mathrm{wt}(\mathbf{i})$.

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 2.1.3, $(P^{(\mathbf{i})})^{(\mathbf{j})}(\mathbf{a}) = \binom{\mathbf{i}+\mathbf{j}}{\mathbf{i}} P^{(\mathbf{i}+\mathbf{j})}(\mathbf{a})$. Since $\mathrm{wt}(\mathbf{i}+\mathbf{j}) = \mathrm{wt}(\mathbf{i}) + \mathrm{wt}(\mathbf{j}) < m$, we deduce that $(P^{(\mathbf{i})})^{(\mathbf{j})}(\mathbf{a}) = 0$. Thus $\mathrm{mult}(P^{(\mathbf{i})}, \mathbf{a}) \geq m - \mathrm{wt}(\mathbf{i})$. ■

We now discuss the behavior of multiplicities under composition of polynomial tuples. Let $\mathbf{X} = (X_1, \ldots, X_n)$ and $\mathbf{Y} = (Y_1, \ldots, Y_\ell)$ be formal variables. Let $\mathbf{P}(\mathbf{X}) = (P_1(\mathbf{X}), \ldots, P_m(\mathbf{X})) \in \mathbb{F}[\mathbf{X}]^m$ and $\mathbf{Q}(\mathbf{Y}) = (Q_1(Y), \ldots, Q_n(Y)) \in \mathbb{F}[\mathbf{Y}]^n$. We define the composition polynomial $\mathbf{P} \circ \mathbf{Q}(\mathbf{Y}) \in \mathbb{F}[\mathbf{Y}]^m$ to be the polynomial $\mathbf{P}(Q_1(\mathbf{Y}), \ldots, Q_n(\mathbf{Y}))$. In this situation we have the following proposition.

Proposition 2.1.5 *Let* $\mathbf{P}(\mathbf{X}), \mathbf{Q}(\mathbf{Y})$ *be as above. Then for any* $\mathbf{a} \in \mathbb{F}^\ell$*,*

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

In particular, since $\mathrm{mult}(\mathbf{Q} - \mathbf{Q}(\mathbf{a}), \mathbf{a}) \geq 1$*, we have* $\mathrm{mult}(\mathbf{P} \circ \mathbf{Q}, \mathbf{a}) \geq \mathrm{mult}(\mathbf{P}, \mathbf{Q}(\mathbf{a}))$.

Proof Let $m_1 = \text{mult}(\mathbf{P}, \mathbf{Q}(\mathbf{a}))$ and $m_2 = \text{mult}(\mathbf{Q} - \mathbf{Q}(\mathbf{a}), \mathbf{a})$. Clearly $m_2 > 0$. If $m_1 = 0$ the result is obvious. Now assume $m_1 > 0$ (so that $\mathbf{P}(\mathbf{Q}(\mathbf{a})) = 0$).

$$\begin{aligned}
\mathbf{P}(\mathbf{Q}(\mathbf{a}+\mathbf{Z})) &= \mathbf{P}\left(\mathbf{Q}(\mathbf{a}) + \sum_{\mathbf{i}\neq 0} \mathbf{Q}^{(\mathbf{i})}(\mathbf{a})\mathbf{Z}^{\mathbf{i}}\right) \\
&= \mathbf{P}\left(\mathbf{Q}(\mathbf{a}) + \sum_{\text{wt}(\mathbf{i})\geq m_2} \mathbf{Q}^{(\mathbf{i})}(\mathbf{a})\mathbf{Z}^{\mathbf{i}}\right) \quad \text{since mult}(\mathbf{Q} - \mathbf{Q}(\mathbf{a}), \mathbf{a}) = m_2 > 0 \\
&= \mathbf{P}\left(\mathbf{Q}(\mathbf{a}) + h(\mathbf{Z})\right) \qquad \text{where } h(\mathbf{Z}) = \textstyle\sum_{\text{wt}(\mathbf{i})\geq m_2} \mathbf{Q}^{(\mathbf{i})}(\mathbf{a})\mathbf{Z}^{\mathbf{i}} \\
&= \mathbf{P}(\mathbf{Q}(\mathbf{a})) + \sum_{\mathbf{j}\neq 0} \mathbf{P}^{(\mathbf{j})}(\mathbf{Q}(\mathbf{a}))h(\mathbf{Z})^{\mathbf{j}} \\
&= \sum_{\text{wt}(\mathbf{j})\geq m_1} \mathbf{P}^{(\mathbf{j})}(\mathbf{Q}(\mathbf{a}))h(\mathbf{Z})^{\mathbf{j}} \qquad \text{since mult}(\mathbf{P}, \mathbf{Q}(\mathbf{a})) = m_1 > 0
\end{aligned}$$

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

Corollary 2.1.6 *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 $P_{\mathbf{a},\mathbf{b}}(T)$ be the polynomial $P(\mathbf{a} + T \cdot \mathbf{b}) \in \mathbb{F}[T]$. Then for any $t \in \mathbb{F}$,*

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

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

2.1.4 Strengthening of the Schwartz-Zippel Lemma

We are now ready to state and prove the strengthening of the Schwartz-Zippel lemma. In the standard form 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 $\min\{1, \text{mult}(P, \mathbf{a})\}$ as the indicator variable that is 1 if $P(\mathbf{a}) = 0$, this lemma can be restated as saying $\sum_{\mathbf{a} \in S^n} \min\{1, \text{mult}(P, \mathbf{a})\} \leq d \cdot |S|^{n-1}$. Our version below strengthens this lemma by replacing $\min\{1, \text{mult}(P, \mathbf{a})\}$ with $\text{mult}(P, \mathbf{a})$ in this inequality.

Lemma 2.1.7 *Let $P \in \mathbb{F}[\mathbf{X}]$ be an n-variate nonzero polynomial of total degree at most d. Then for any finite $S \subseteq \mathbb{F}$,*

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

Proof We prove it by induction on n.

For the base case when $n = 1$, we first show that if $\text{mult}(P, a) = m$ 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 $\sum_{a \in S} \text{mult}(P, a)$ 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 $m_{a_1, \ldots, a_{n-1}} = \text{mult}(P_t, (a_1, \ldots, a_{n-1}))$. We will show that

$$\sum_{a_n \in S} \text{mult}(P, (a_1, \ldots, a_n)) \leq m_{a_1, \ldots, a_{n-1}} \cdot |S| + t. \tag{2.2}$$

Given this, we may then bound

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

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 Equation (2.2). 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 $P_t^{(\mathbf{i})}(X_1,\ldots,X_{n-1})\neq 0$. 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 2.1.4 and Corollary 2.1.6 (applied on the line through $(a_1,\ldots,a_{n-1},0)$ in direction $(0,\ldots,0,a_n)$), we know that

$$\begin{aligned}\mathrm{mult}(P(X_1,\ldots,X_n),(a_1,\ldots,a_n)) &\leq \mathrm{wt}(\mathbf{i},0)+\mathrm{mult}(P^{(\mathbf{i},0)}(X_1,\ldots,X_n),(a_1,\ldots,a_n))\\ &\leq m_{a_1,\ldots,a_{n-1}}+\mathrm{mult}(P^{(\mathbf{i},0)}(a_1,\ldots,a_{n-1},X_n),a_n).\end{aligned}$$

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 Equation (2.2). This completes the proof of the lemma. ■

The following corollary simply states the above lemma in contrapositive form, with $S=\mathbb{F}_q$.

Corollary 2.1.8 *Let $P\in\mathbb{F}_q[\mathbf{X}]$ be a polynomial of total degree at most d. If $\sum_{\mathbf{a}\in\mathbb{F}_q^n}\mathrm{mult}(P,\mathbf{a}) > d\cdot q^{n-1}$, then $P(\mathbf{X})=0$.*

Chapter 3

High Rate Codes with Sublinear Time Decoding

3.1 Main Results on the Existence of Locally Decodable Codes

In this section, we state our main results on the existence of locally decodable codes with rate approaching 1. We begin with some standard definitions.

For a set Σ and two vectors $c, c' \in \Sigma^n$, we define their relative Hamming distance $\Delta(c, c')$ to be the fraction of coordinates where they differ: $\Delta(c, c') = \Pr_{i \in [n]}[c_i \neq c'_i]$.

An error-correcting code of length n over the alphabet Σ is a subset $\mathcal{C} \subseteq \Sigma^n$. The *rate* of $\mathcal{C}$ is the defined to be $\frac{\log |\mathcal{C}|}{n \log |\Sigma|}$. The (minimum) distance of $\mathcal{C}$ is defined to be the smallest $\delta > 0$ such that for every distinct $c_1, c_2 \in \mathcal{C}$, we have $\Delta(c_1, c_2) \geq \delta$.

For q a prime power, let $\mathbb{F}_q$ denote the finite field on q elements. If $\Sigma = \mathbb{F}_q$, then a code $\mathcal{C}$ over the alphabet Σ is called a *linear* code if $\mathcal{C}$ is a $\mathbb{F}_q$-linear subspace of Σ^n.

We now define locally self-correctable codes and locally decodable codes. For an algorithm A and a string r, we will use A^r to represent the situation where A is given query access to r.

Definition 3.1.1 (Locally Self-correctable Code) *A code $\mathcal{C} \subseteq \Sigma^n$ is said to be locally self-correctable from δ'-fraction errors with t queries if there is a randomized algorithm A, such that:*

- **Self-correction:** *Whenever $c \in \mathcal{C}$ and $r \in \Sigma^n$ are such that $\Delta(r, c) < \delta'$, then for each $i \in [n]$,*
$$\Pr[\mathsf{A}^r(i) = c_i] \geq 2/3.$$

- **Query complexity t:** *$\mathsf{A}^r(i)$ always makes at most t queries to r.*

Definition 3.1.2 (Locally Decodable Code) *Let $\mathcal{C} \subseteq \Sigma^n$ be a code with $|\mathcal{C}| = |\Sigma|^k$. Let $E : \Sigma^k \to \mathcal{C}$ be a bijection (we refer to E as the encoding map for $\mathcal{C}$; note that k/n equals the rate of the code $\mathcal{C}$). We say that $(\mathcal{C}, E)$ is locally decodable from δ'-fraction errors with t queries if there is a randomized algorithm A, such that:*

- **Decoding:** *Whenever $x \in \Sigma^k$ and $r \in \Sigma^n$ are such that $\Delta(r, E(x)) < \delta'$, then for each $i \in [k]$,*
$$\Pr[\mathsf{A}^r(i) = x_i] \geq 2/3.$$

- **Query complexity t:** *$\mathsf{A}^r(i)$ always makes at most t queries to r.*

Suppose $\Sigma = \mathbb{F}_q$, and that $\mathcal{C}$ is a linear code over Σ. By simple linear algebra, it follows that there is an encoding function E such that for each $x \in \Sigma^k$ and each $i \in [k]$ there is a $j \in [n]$, such that $E(x)_j = x_i$. This implies that if $\mathcal{C}$ is locally self-correctable (from some fraction of errors with some query complexity), then $(\mathcal{C}, E)$ is locally decodable (from the same fraction of errors and with the same query complexity). This will allow us to focus on constructing linear codes which are locally self-correctable.

We now state the two main theorems, which assert the existence of locally self-correctable codes with improved rate and query complexity. The first theorem, which does this over a large alphabet (and does not give a linear code), will be a direct consequence of what we show about multiplicity codes in the next section.

Theorem 3.1.3 (Locally self-correctable codes over large alphabets) *For every $0 < \epsilon, \alpha < 1$, for infinitely many n, there is a code C over an alphabet Σ, with $|\Sigma| \leq n^{O(1)}$, such that C has length n, rate at least $1 - \alpha$, distance $\delta \geq \epsilon\alpha/2$, and is locally self-correctable from $\delta/10$-fraction errors with $O(n^{\epsilon})$ queries.*

The next theorem is the analogue of theorem 3.1.3 for small alphabets (and gives linear codes). These codes are obtained by simply concatenating multiplicity codes with suitable good linear codes over the small alphabet. In particular, this shows the existence of locally decodable codes with similar parameters.

Theorem 3.1.4 (Locally self-correctable codes over small alphabets) *Let p be a prime power. For every $\epsilon, \alpha > 0$, there exists $\delta > 0$, such that for infinitely many n, there is a linear code $\mathcal{C}$ over the alphabet $\Sigma = \mathbb{F}_p$, such that $\mathcal{C}$ has length n, rate at least $1 - \alpha$, distance at least δ, and is locally self-correctable from $\delta/20$-fraction errors with $O(n^{\epsilon})$ queries.*

Remark The codes in both the above theorems are efficiently constructible. Furthermore, both the local self-correction algorithms can be made to run in time $O(n^{2\epsilon})$.

The proofs of the theorems above appear in Section 3.2.3.

3.2 Multiplicity Codes

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

First, we recall some preliminaries on derivatives and multiplicities from Chapter 2 that will be useful for our construction. 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.

3.2.1 Derivatives and Multiplicities

We start with some notation. We use $[m]$ to denote the set $\{1, \ldots, m\}$. For a vector $\mathbf{i} = \langle i_1, \ldots, i_m \rangle$ of non-negative integers, its *weight*, denoted $\mathrm{wt}(\mathbf{i})$, equals $\sum_{j=1}^{m} i_j$.

For a field $\mathbb{F}$, let $\mathbb{F}[X_1, \ldots, X_m] = \mathbb{F}[\mathbf{X}]$ be the ring of polynomials in the variables $X_1, \ldots, X_m$ with coefficients in $\mathbb{F}$.

For a vector of non-negative integers $\mathbf{i} = \langle i_1, \ldots, i_m \rangle$, let $\mathbf{X}^{\mathbf{i}}$ denote the monomial $\prod_{j=1}^{m} X_j^{i_j} \in \mathbb{F}[\mathbf{X}]$. Note that the (total) degree of this monomial equals $\mathrm{wt}(\mathbf{i})$.

Definition 3.2.1 ((Hasse) Derivative) *For $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ and non-negative vector $\mathbf{i}$, the $\mathbf{i}$th* (Hasse) derivative *of P, denoted $P^{(\mathbf{i})}(\mathbf{X})$, is the coefficient of $\mathbf{Z}^{\mathbf{i}}$ in the polynomial $\tilde{P}(\mathbf{X}, \mathbf{Z}) \stackrel{\text{def}}{=} P(\mathbf{X} + \mathbf{Z}) \in \mathbb{F}[\mathbf{X}, \mathbf{Z}]$.*

Thus,

$$P(\mathbf{X} + \mathbf{Z}) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{X}) \mathbf{Z}^{\mathbf{i}}. \tag{3.1}$$

Observe that for all $P, Q \in \mathbb{F}[X]$, and $\lambda \in \mathbb{F}$,

$$(\lambda P)^{(\mathbf{i})}(\mathbf{X}) = \lambda P^{(\mathbf{i})}(\mathbf{X}) \quad \text{and} \quad \mathrm{P}^{(\mathbf{i})}(\mathbf{X}) + \mathrm{Q}^{(\mathbf{i})}(\mathbf{X}) = (\mathrm{P} + \mathrm{Q})^{(\mathbf{i})}(\mathbf{X}). \tag{3.2}$$

We are now ready to define the notion of the (zero-)multiplicity of a polynomial at any given point.

Definition 3.2.2 (Multiplicity) *For $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ and $\mathbf{a} \in \mathbb{F}^m$, the* multiplicity *of P at $\mathbf{a} \in \mathbb{F}^m$, denoted* $\mathrm{mult}(P, \mathbf{a})$*, is the largest integer M such that for every non-negative vector $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) < M$, we have $P^{(\mathbf{i})}(\mathbf{a}) = 0$ (if M may be taken arbitrarily large, we set $\mathrm{mult}(P, \mathbf{a}) = \infty$).*

Note that $\mathrm{mult}(P, \mathbf{a}) \geq 0$ for every $\mathbf{a}$.

The multiplicity enhanced Schwartz-Zippel Lemma is the main technical fact we will need about derivatives and multiplicities. We state this lemma below. The proof appears in Chapter 2.

Lemma 3.2.3 *Let $P \in \mathbb{F}[\mathbf{X}]$ be a nonzero polynomial of total degree at most d. Then for any finite $S \subseteq \mathbb{F}$,*

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

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

$$\Pr_{\mathbf{a} \in S^m}[\text{mult}(P, \mathbf{a}) \geq s] \leq \frac{d}{s|S|}.$$

3.2.2 The Definition of Multiplicity Codes

We now come to the definition of multiplicity codes.

Definition 3.2.4 (Multiplicity code) *Let s, d, m be nonnegative integers and let q be a prime power. Let $\Sigma = \mathbb{F}_q^{\binom{m+s-1}{m}} = \mathbb{F}_q^{\{\mathbf{i}:\text{wt}(\mathbf{i})<s\}}$. For $P(X_1, \ldots, X_m) \in \mathbb{F}_q[X_1, \ldots, X_m]$, we define* the order s evaluation of P at $\mathbf{a}$, *denoted $P^{(<s)}(\mathbf{a})$, to be the vector $\langle P^{(\mathbf{i})}(\mathbf{a}) \rangle_{\text{wt}(\mathbf{i})<s} \in \Sigma$.*

We define the multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$ *as follows. The code is over the alphabet Σ, and has length q^m (where the coordinates are indexed by elements of $\mathbb{F}_q^m$). For each polynomial $P(\mathbf{X}) \in \mathbb{F}_q[X_1, \ldots, X_m]$ with $\deg(P) \leq d$, there is a codeword in $\mathcal{C}$ given by:*

$$\text{Enc}_{s,d,m,q}(P) = \langle P^{(<s)}(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q^m} \in (\Sigma)^{q^m}.$$

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

Lemma 3.2.5 (Rate and distance of multiplicity codes) *Let $\mathcal{C}$ be the multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$. Then*

$\mathcal{C}$ has distance $\delta = 1 - \frac{d}{sq}$ and rate $\frac{\binom{d+m}{m}}{\binom{s+m-1}{m}q^m}$, which is at least

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

Proof The alphabet size equals $q^{\binom{m+s-1}{m}}$. The length equals q^m.

To calculate the distance, consider any two codewords $c_1 = \mathrm{Enc}_{s,d,m,q}(P_1), c_2 = \mathrm{Enc}_{s,d,m,q}(P_2)$, where $P_1 \neq P_2$. For any coordinate $\mathbf{a} \in \mathbb{F}_q^m$ where the codewords c_1, c_2 agree (i.e., $(c_1)_{\mathbf{a}} = (c_2)_{\mathbf{a}}$), we have that $P_1^{(<s)}(\mathbf{a}) = P_2^{(<s)}(\mathbf{a})$. Thus for any such $\mathbf{a}$, we have $(P_1 - P_2)^{(\mathbf{i})}(\mathbf{a}) = 0$ for each $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) < s$, and hence $\mathrm{mult}(P_1 - P_2, \mathbf{a}) \geq s$. From the bound on the number of high-multiplicity zeroes of multivariate polynomials, Lemma 3.2.3, the fraction of $\mathbf{a} \in \mathbb{F}_q^m$ on which this can happen is at most $\frac{d}{sq}$. The minimum distance δ of the multiplicity code is therefore at least $\delta = 1 - \frac{d}{sq}$.

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

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

■

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

Theorem 3.2.6 (Multiplicity codes are locally self-correctable) *Let $\mathcal{C}$ be the multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$. Let $\delta = 1 - d/sq$ be the distance of $\mathcal{C}$. Suppose $q \geq \max\{10m, \frac{d+6}{s}, 5(s+1)\}$. Then $\mathcal{C}$ is locally self-correctable from $\frac{\delta}{10}$-fraction errors with $O(s)^m \cdot q$-queries.*

The proof of this theorem appears in Section 3.3.2. In Section 3.3.3, we will show that the local self-corrector can also be made to run very efficiently, in time $O(s)^m \cdot q^{O(1)}$.

Multiplicity codes can also be locally decoded with a factor $\exp(m+s)$-increase in the query complexity, for a suitable choice of encoding function (even though multiplicity codes are not linear). We omit the details.

3.2.3 Proof of the Main Theorems

We now show how to instantiate multiplicity codes to prove our main theorems on the existence of locally self-correctable codes with improved rate and query-complexity (assuming Theorem 3.2.6).

Proof of Theorem 3.1.3: Recall that we are trying to construct, for ever $0 < \epsilon, \alpha < 1$, for infinitely many n, a code over an alphabet of size $n^{O(1)}$, with length n, rate $\geq 1-\alpha$, distance $\delta \geq \epsilon\alpha/2$, and locally self-correctable with $O(n^\epsilon)$ queries from $\delta/10$-fraction errors.

Pick $m = \lceil 1/\epsilon \rceil$. For every large enough prime power q, we will construct such a code with $n = q^m$. Pick s so that

$$1 - \frac{m^2}{s} > \frac{1-\alpha}{(1-\delta)^m},$$

(this can be done with $s = O(m^2)$). Observe that m and s are constants. Let $d = (1-\delta) \cdot s \cdot q$. Observe that for all α, ϵ, $1 - \alpha < (1 - \epsilon\alpha/2)^{2/\epsilon} < (1 - \epsilon\alpha/2)^{\lceil 1/\epsilon \rceil}$, and hence $1 - \alpha < (1-\delta)^m$.

Let $\mathcal{C}$ be the multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$. Observe that $\mathcal{C}$ has length n and is over an alphabet of size $q^{\binom{m+s-1}{m}} = n^{O(1)}$. By Lemma 3.2.5, $\mathcal{C}$ has distance δ and rate at least $(1 - \frac{m^2}{s}) \cdot (1 - \delta)^m > 1 - \alpha$. By Theorem 3.2.6, $\mathcal{C}$ can be locally self-corrected from $\delta/10$-fraction errors using $O(n^{1/m}) = O(n^\epsilon)$ queries. This completes the proof of Theorem 3.1.3. ■

Finally, we complete the proof of Theorem 3.1.4, by concatenating suitable multiplic-

ity codes with good linear codes over small alphabets.

Proof of Theorem 3.1.4: Set $\alpha_1 = \alpha/2$ and $\epsilon_1 = \epsilon/2$. As in the proof of Theorem 3.1.3, there are constants m and s such that for every prime power q, there is a multiplicity code with length $n_1 = q^m$, rate $1 - \alpha_1$, distance $\delta_1 \geq \epsilon_1\alpha_1/2$, over an alphabet Σ_1 of size $q^{\binom{m+s-1}{m}}$, and locally self-correctable from $\delta_1/10$ with $O(n_1^{\epsilon_1})$ queries. We will take such codes $\mathcal{C}_1$ where $q = p^t$ for integers $t > 0$.

We now pick another code $\mathcal{C}_2$ of length $\binom{m+s-1}{m} \cdot t$ that is $\mathbb{F}_p$-linear and has rate $1-\alpha_1$ and use it to encode the symbols of $\mathcal{C}_1$. The resulting concatenated code $\mathcal{C}$ is $\mathbb{F}_p$-linear (this follows from the linearity of $\mathcal{C}_2$ and the "pseudo-linearity" of $\mathcal{C}_1$ coming from Equation (3.2)), and has distance δ and rate R that are at least the products of the corresponding parameters of $\mathcal{C}_1$ and $\mathcal{C}_2$. In particular, if we take $\mathcal{C}_2$ to be a code of constant distance $\delta_2 > 0$ ($\mathcal{C}_2$ can even be taken to be efficiently constructible, and such that there are efficient error-correction algorithms for decoding upto half the minimum distance), then $\mathcal{C}$ has length $n = q^m \cdot \binom{m+s-1}{m} \cdot t \cdot \frac{1}{1-\alpha_1}$, rate at least $1 - \alpha$ and constant (as n grows) distance $\delta > 0$.

We now argue that the code $\mathcal{C}$ is locally self-correctable. To locally self-correct some coordinate of a codeword of $\mathcal{C}$ given access to a corrupted codeword of $\mathcal{C}$, we first run the local self-corrector for $\mathcal{C}_1$ to decode the coordinate of $\mathcal{C}_1$ that contains that coordinate of $\mathcal{C}$. Whenever this local self-corrector wants to query a certain coordinate of $\mathcal{C}_1$, we recover that symbol by decoding the corresponding codeword of $\mathcal{C}_2$ (if we only care about query complexity, this can be done by brute force; if we are interested in having sublinear running time, then $\mathcal{C}_2$ should be chosen so that this step can be done in time polynomial in the length of $\mathcal{C}_2$). The query complexity of the local self-corrector for $\mathcal{C}$ is clearly $O(n^{\epsilon_1} \log n) = O(n^{\epsilon})$. It remains to note that in case the total fraction of errors is below $\delta/20$, all but $\delta_1/10$ fraction of the $\mathcal{C}_2$ blocks will have $< \delta_2/2$-fraction errors, and can be correctly recovered by the decoder for $\mathcal{C}_2$. Thus the local self-corrector for $\mathcal{C}_1$ will run correctly, and this yields the desired corrected coordinate of $\mathcal{C}$. ■

3.3 Local Self-correction of Multiplicity Codes

In this section, we prove that multiplicity codes are locally self-correctable.

Suppose we are dealing with the multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$. Let Σ be the alphabet for this code. Let $r : \mathbb{F}_q^m \to \Sigma$ be a received word. Suppose P is a polynomial over $\mathbb{F}_q$ in m variables of degree at most d such that $\Delta(r, \mathrm{Enc}_{s,d,m,q}(P))$ is small. Let $\mathbf{a} \in \mathbb{F}_q^m$. Let us show how to locally recover $P^{(<s)}(\mathbf{a})$ given oracle access to r.

As indicated in the introduction, the idea is to pick many random lines containing $\mathbf{a}$, and to consider the restriction of r to those lines. With high probability over a random direction $\mathbf{b} \in \mathbb{F}_q^m \setminus \{0\}$, by looking at the restriction of r to the line $\mathbf{a} + \mathbf{b}T$ and "decoding" it, we will be able to recover the univariate polynomial $P(\mathbf{a} + \mathbf{b}T)$. Knowing this univariate polynomial will tell us a certain linear combination of the various derivatives of P at $\mathbf{a}$, $\langle P^{(\mathbf{i})}(\mathbf{a})\rangle_{\mathrm{wt}(\mathbf{i})<s}$. Combining this information for various directions $\mathbf{b}$, we will know a system of various linear combinations of the numbers $\langle P^{(\mathbf{i})}(\mathbf{a})\rangle_{\mathrm{wt}(\mathbf{i})<s}$. Solving this linear system, we get $P^{(\mathbf{i})}(\mathbf{a})$ for each $\mathbf{i}$, as desired.

To implement this strategy we need to relate the derivatives of the restriction of a multivariate polynomial P to a line to the derivatives of P itself. Fix $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^n$, and consider the polynomial $Q(T) = P(\mathbf{a} + \mathbf{b}T)$.

- **The relationship of $Q(T)$ with the derivatives of P at $\mathbf{a}$:** By the definition of Hasse derivatives,

$$Q(T) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{a})\mathbf{b}^{\mathbf{i}} T^{\mathrm{wt}(\mathbf{i})}.$$

  Grouping terms, we see that:

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} P^{(\mathbf{i})}(\mathbf{a})\mathbf{b}^{\mathbf{i}} = \text{coefficient of } T^e \text{ in } Q(T). \tag{3.3}$$

- **The relationship of the derivatives of Q at t with the derivatives of P at $\mathbf{a} + t\mathbf{b}$:** Let $t \in \mathbb{F}_q$. By the definition of Hasse derivatives, we get the

following two identities:

$$P(\mathbf{a}+\mathbf{b}(t+R)) = Q(t+R) = \sum_j Q^{(j)}(t)R^j.$$

$$P(\mathbf{a}+\mathbf{b}(t+R)) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{a}+\mathbf{b}t)(\mathbf{b}R)^{\mathbf{i}}.$$

Thus,

$$Q^{(j)}(t) = \sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=j} P^{(\mathbf{i})}(\mathbf{a}+\mathbf{b}t)\mathbf{b}^{\mathbf{i}}. \tag{3.4}$$

In particular, $Q^{(j)}(t)$ is simply a linear combination of the various $P^{(\mathbf{i})}(\mathbf{a}+\mathbf{b}t)$ (over different $\mathbf{i}$).

We are now in a position to describe our decoding algorithm. Before describing the main local self-correction algorithm for correcting from $\Omega(\delta)$-fraction errors, we describe 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 comment on how to make them run time-efficiently. In Section 3.3.3, we show how the various steps of the algorithms can be made to run in a time-efficient manner as well.

3.3.1 Simplified Error-Correction from Few Errors

Simplified Local Self-correction Algorithm

Input: received word $r : \mathbb{F}_q^m \to \Sigma$, point $\mathbf{a} \in \mathbb{F}_q^m$. We are trying to recover $P^{(<s)}(\mathbf{a})$, where $P(\mathbf{X})$ is such that $\mathrm{Enc}_{s,d,m,q}(P)$ is close to r. Abusing notation, we will write $r^{(\mathbf{i})}(\mathbf{a})$ when we mean the $\mathbf{i}$ coordinate of $r(\mathbf{a})$.

1. **Pick a set B of directions:** Choose $B \subseteq \mathbb{F}_q^m \setminus \{0\}$, a uniformly random subset of size $w = \binom{m+s-1}{m}$.

2. **Recover $P(\mathbf{a}+\mathbf{b}T)$ for directions $\mathbf{b} \in B$:** For each $\mathbf{b} \in B$, consider the function $\ell_{\mathbf{b}} : \mathbb{F}_q \to \mathbb{F}_q^s$ given by

$$(\ell_{\mathbf{b}}(t))_j = \sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=j} r^{(\mathbf{i})}(\mathbf{a}+\mathbf{b}t)\mathbf{b}^{\mathbf{i}}.$$

Find the polynomial $Q_{\mathbf{b}}(T) \in \mathbb{F}_q[T]$ of degree at most d (if any), such that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q_{\mathbf{b}}), \ell_{\mathbf{b}}) < \delta/2$.

3. **Solve a linear system to recover $P^{(<s)}(a)$:** For each e with $0 \le e < s$, consider the following system of equations in the variables $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ (with one equation for each $\mathbf{b} \in B$):

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} \mathbf{b}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coefficient of } T^e \text{ in } Q_{\mathbf{b}}(T). \tag{3.5}$$

Find all $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ which satisfy at all these equations. If there are 0 or > 1 solutions, output FAIL.

4. Output the vector $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})<s}$.

We will show that the above algorithm is a local self-corrector from a $\frac{\delta}{100\binom{m+s-1}{m}}$-fraction of errors. Fix a received word $r : \mathbb{F}_q^m \to \Sigma$ and $\mathbf{a} \in \mathbb{F}_q^m$. Let $P(X_1, \ldots, X_m)$ be a polynomial such that $\Delta(\mathrm{Enc}_{s,d,m,q}(P), r) < \frac{\delta}{100\binom{m+s-1}{m}}$. We will call the set of points where r and $\mathrm{Enc}_{s,d,m,q}(P)$ differ the "errors".

Step 1: All the $\mathbf{b} \in B$ are "good". For $\mathbf{b} \in \mathbb{F}_q^m \setminus \{0\}$, we will be interested in the fraction of errors on the line $\{\mathbf{a} + t\mathbf{b} \mid t \in \mathbb{F}_q \setminus \{0\}\}$ through $\mathbf{a}$ in direction $\mathbf{b}$. Since these lines cover $\mathbb{F}_q^m \setminus \{\mathbf{a}\}$ uniformly, we can conclude that at most $\frac{1}{50\binom{m+s-1}{m}}$ of the lines containing $\mathbf{a}$ have more than $\delta/2$-fraction error on them. Hence with probability at least 0.9 over the choice of B, all the $\mathbf{b} \in B$ will be such that the line through $\mathbf{a}$ in direction $\mathbf{b}$ has fewer than $\delta/2$ errors on it.

Step 2: $Q_{\mathbf{b}}(T) = P(\mathbf{a}+\mathbf{b}T)$ for each $\mathbf{b} \in B$. Assume that B is such that the above event occurs. In this case, by Equation (3.4), for each $\mathbf{b} \in B$, the corresponding

function $\ell_{\mathbf{b}}$ will be such that $\Delta(\mathrm{Enc}_{s,d,1,q}(P(\mathbf{a}+\mathbf{b}T), \ell_{\mathbf{b}}) < \delta/2$. Thus for each $\mathbf{b} \in B$, the algorithm will find $Q_{\mathbf{b}}(T) = P(\mathbf{a} + \mathbf{b}T)$. (Note that at most one polynomial $Q(T)$ of degree at most d has the property that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q), \ell_{\mathbf{b}}) < \delta < 1/2$. This is because for distinct $Q(T), Q'(T)$ of degree at most d, Lemma 3.2.5 implies that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q), \mathrm{Enc}_{s,d,1,q}(Q')) \geq \delta$.)

Step 3: $u_{\mathbf{i}} = P^{(\mathbf{i})}(\mathbf{a})$ for each i. Since $Q_{\mathbf{b}}(T) = P(\mathbf{a} + \mathbf{b}T)$ for each $\mathbf{b} \in B$, Equation (3.3) now implies that for each $0 \leq e < s$, the vector $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ with $u_{\mathbf{i}} = P^{(\mathbf{i})}(\mathbf{a})$ will satisfy all the equations in the system (3.5). Finally, we observe that this solution $u_{\mathbf{i}}$ is unique. Indeed, with probability at least 0.9 over the choice of B, the elements of B will form an interpolating set for polynomials of degree $< s$ (this holds as long as q is large enough in terms of m and s); in particular, there is no nonzero polynomial of degree $< s$ that vanishes on all the points of B.

Hence for each e, the vector $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ that satisfies all the equations in the system (3.5) is unique. If not, then the difference $\langle u_{\mathbf{i}} - u'_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ of two such vectors $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}, \langle u'_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ will be the vector of coefficients of a polynomial of degree $< s$ that vanishes on all of B (for every $\mathbf{b} \in B$, we have: $\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e}(u_{\mathbf{i}} - u'_{\mathbf{i}})(\mathbf{b}^{\mathbf{i}}) = 0$), contradicting the fact that B is an interpolating set for polynomials of degree $< s$.

Overall, with probability at least 0.8, the algorithm will output $P^{(\mathbf{i})}(\mathbf{a})$, as desired.

3.3.2 Error-Correction from $\Omega(\delta)$-Fraction Errors

We now come to the main local self-correcting algorithm and the proof of Theorem 3.2.6. As above, to decode at a point $\mathbf{a}$, we will pick several lines $\mathbf{a}+\mathbf{b}T$ through $\mathbf{a}$, and try to recover the univariate polynomial $P(\mathbf{a} + \mathbf{b}T)$. However, unlike the above algorithm, we will not count on the event that all these lines have less than $\delta/2$-fraction errors. Instead, we will 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 self-correct from a significantly larger fraction of errors. To compensate

for the weakening, we will need to make the next step of the algorithm, that of solving a linear system, more robust; we will 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 $\binom{m+s-1}{m}$ random lines through $\mathbf{a}$ and used them to decode from $\Omega(\frac{\delta}{\binom{m+s-1}{m}})$-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 $\Theta(\binom{m+s-1}{m})$ independent and uniformly random lines through the point $\mathbf{a}$. The algorithm we present below picks these lines differently; the directions of these lines will come from a random affinely transformed grid. This way of choosing lines admits a simpler analysis, and the noisy system of linear equations that we end up needing to solve turns becomes an instance of the noisy polynomial interpolation problem on a grid, for which time-efficient algorithms are known.

Main Local Self-Correction Algorithm:

Input: received word $r : \mathbb{F}_q^m \to \Sigma$, point $\mathbf{a} \in \mathbb{F}_q^m$. Abusing notation again, we will write $r^{(\mathbf{i})}(\mathbf{a})$ when we mean the $\mathbf{i}$ coordinate of $r(\mathbf{a})$.

1. **Pick a set B of directions:** Pick $\mathbf{z}, \mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_m \in \mathbb{F}_q^m$ independently and uniformly at random. Let $S \subset \mathbb{F}_q$ be any set of size $5(s+1)$. Let $B = \{\mathbf{z} + \sum_{i=1}^m \alpha_i \mathbf{y}_i \mid \alpha_i \in S\}$.

2. **Recover $P(\mathbf{a} + \mathbf{b}T)$ for directions $\mathbf{b} \in B$:** For each $\mathbf{b} \in B$, consider the function $\ell_{\mathbf{b}} : \mathbb{F}_q \to \mathbb{F}_q^s$ given by

$$(\ell_{\mathbf{b}}(t))_j = \sum_{\mathbf{i} | \mathrm{wt}(\mathbf{i})=j} r^{(\mathbf{i})}(\mathbf{a} + \mathbf{b}t)\mathbf{b}^{\mathbf{i}}.$$

   Find the polynomial $Q_{\mathbf{b}}(T) \in \mathbb{F}_q[T]$ of degree at most d (if any), such that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q_{\mathbf{b}}), \ell_{\mathbf{b}}) < \delta/2$.

3. **Solve a noisy linear system to recover $P^{(<s)}(a)$:** For each e with $0 \le e < s$, consider the following system of equations in the variables $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ (with one

equation for each $\mathbf{b} \in B$):

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} \mathbf{b}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coeff of } T^e \text{ in } Q_{\mathbf{b}}(T). \tag{3.6}$$

Find all $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ which satisfy at least 3/5 of these equations. If there are 0 or > 1 solutions, output FAIL.

4. Output the vector $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})<s}$.

We now proceed to analyze the above algorithm (and thus complete the proof of Theorem 3.2.6).

Proof of Theorem 3.2.6: For $m = 1$ the theorem is trivial, and so we assume $m \geq 2$. Recall that we have $q \geq 10m$, $q \geq \frac{d+6}{s}$ (so that $q \geq \frac{6}{\delta}$) and that $q \geq 5(s+1)$. We will show that the above algorithm is a local self-corrector from a $\frac{\delta}{10}$-fraction of errors. Fix a received word $r : \mathbb{F}_q^m \to \Sigma$ and $\mathbf{a} \in \mathbb{F}_q^m$. Let $P(X_1, \ldots, X_m)$ be a polynomial such that $\Delta(\mathrm{Enc}_{s,d,m,q}(P), r) < \frac{\delta}{10}$. We will call the set of points where r and $\mathrm{Enc}_{s,d,m,q}(P)$ differ the "errors".

Step 1: Many $\mathbf{b} \in B$ are "good". For $\mathbf{b} \in \mathbb{F}_q^m \setminus \{0\}$, we will be interested in the fraction of errors on the line $\{\mathbf{a} + t\mathbf{b} \mid t \in \mathbb{F}_q\}$ through $\mathbf{a}$ in direction $\mathbf{b}$. Since these lines cover $\mathbb{F}_q^m \setminus \{\mathbf{a}\}$ uniformly, we can conclude that at least $\frac{2}{3}$ of $\mathbf{b} \in \mathbb{F}_q^m$ are such that the line $\{\mathbf{a} + t\mathbf{b} \mid t \in \mathbb{F}_q\}$ has a fraction of errors which is less than $\left(\frac{\delta}{3} + \frac{1}{q}\right) < \frac{\delta}{2}$. We call these directions $\mathbf{b}$ good. In the claim below, we show that the set B samples the set of good directions well.

Claim 3.3.1 *Let m be a positive integer, and let $S \subset \mathbb{F}_q$ be any set such that $|S|^m \geq 50$. Pick $\mathbf{z}, \mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_m \in \mathbb{F}_q^m$ independently and uniformly at random. Let $B = \{\mathbf{z} + \sum_{i=1}^m \alpha_i \mathbf{y}_i \mid \alpha_i \in S\}$. Then for every set $E \subseteq \mathbb{F}_q^m$ of size at least $2q^m/3$, the probability that fewer than $3/5$ of the points of B lie in E 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 $\langle \mathbf{z} + \sum_{i=1}^m \alpha_i \mathbf{y}_i \rangle_{(\alpha_1,\ldots,\alpha_m) \in S^m}$ is pairwise independent. ■

Hence (recall that we have $m \geq 2$, and so $(5(s+1))^m > 50$), with probability at least 0.9 over the choice of B, 3/5-fraction of the $\mathbf{b} \in B$ will be good.

Step 2: $Q_{\mathbf{b}}(T) = P(\mathbf{a} + \mathbf{b}T)$ for each good $\mathbf{b} \in B$. By Equation (3.4), for each good $\mathbf{b} \in B$, the corresponding function $\ell_{\mathbf{b}}$ will be such that $\Delta(\mathrm{Enc}_{s,d,1,q}(P(\mathbf{a} + \mathbf{b}T)), \ell_{\mathbf{b}}) < \delta/2$. Thus for each good $\mathbf{b}$, the algorithm will find $Q_{\mathbf{b}}(T) = P(\mathbf{a} + \mathbf{b}T)$. (Note that at most one polynomial $Q(T)$ of degree at most d has the property that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q), \ell_{\mathbf{b}}) < \delta/2$. This is because for distinct $Q(T), Q'(T)$ of degree at most d, Lemma 3.2.5 implies that $\Delta(\mathrm{Enc}_{s,d,1,q}(Q), \mathrm{Enc}_{s,d,1,q}(Q')) \geq \delta$.)

Step 3: $u_{\mathbf{i}} = P^{(\mathbf{i})}(\mathbf{a})$ for each i. Since $Q_{\mathbf{b}}(T) = P(\mathbf{a} + \mathbf{b}T)$ for each good $\mathbf{b} \in B$, Equation (3.3) now implies that with probability 0.9, for each $0 \leq e < s$, the vector $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ with $u_{\mathbf{i}} = P^{(\mathbf{i})}(\mathbf{a})$ will 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 B, the elements $\mathbf{y}_1, \ldots, \mathbf{y}_m$ will be linearly independent over $\mathbb{F}_q^m$ (since $q \geq 10m$). In this case, there is an $\mathbb{F}_q$-linear map which gives a bijection between S^m and B. Via this linear map, we get a degree preserving correspondence between polynomials evaluated on S^m and polynomials evaluated on B. Now by Lemma 3.2.3 (and recalling that $|S| = 5(s+1)$), there is no nonzero polynomial of degree $< s$ that vanishes on more that 1/5-fraction of the points of S^m. Hence no nonzero polynomial of degree $< s$ vanishes on more that 1/5-fraction of the points of B.

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

Overall, with probability at least 0.8, the algorithm will output $P^{(\mathbf{i})}(\mathbf{a})$, as desired.

This completes the proof of Theorem 3.2.6. ■

3.3.3 Running Time

In this section, we will see how the above local self-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 $Q_{\mathbf{b}}(T)$. 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.

Decoding univariate multiplicity codes. We deal with the first step first, that of decoding univariate multiplicity codes. Explicitly, let $\Sigma = \mathbb{F}_q^s$, we have a function $\ell : \mathbb{F}_q \to \Sigma$, and we want to find the univariate polynomial $Q(T) \in \mathbb{F}_q[T]$ of degree at most d such that $\Delta(\ell, \mathrm{Enc}_{s,d,1,q}(Q)) < (1 - d/sq)/2 = \delta/2$. 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 $0 \leq i < s$.

Univariate multiplicity codes are instances of "ideal error-correcting codes" [GSS00, Sud01]. Formally defining ideal error-correcting codes will take us too far afield; we will content ourselves with instantiating the known algorithm for decoding ideal error-correcting codes in this case, and explicitly writing out the resulting efficient algorithm for decoding univariate multiplicity codes. All these algorithms are extensions of the beautiful Berlekamp-Welch algorithm for decoding Reed-Solomon codes.

Let $Q(T)$ be a polynomial of degree at most d such that $\Delta(\ell, \mathrm{Enc}_{s,d,1,q}(Q)) < \delta/2$. Our underlying goal is to search for polynomials $E(T), N(T)$ such that $N(T) = E(T)Q(T)$ (and so we obtain $Q(T)$ as $N(T)/E(T)$). By the product rule for Hasse derivatives (which states that $(P_1 \cdot P_2)^{(i)}(T) = \sum_{j=0}^{i} P_1^{(j)}(T) P_2^{(i-j)}(T)$, see [HKT08]),

such polynomials $E(T)$, $N(T)$ will also satisfy the equalities

$$\begin{aligned} N^{(1)}(T) &= (E \cdot Q)^{(1)}(T) = E(T)Q^{(1)}(T) + E^{(1)}(T)Q(T), \\ N^{(2)}(T) &= (E \cdot Q)^{(2)}(T) = E(T)Q^{(2)}(T) + E^{(1)}(T)Q^{(1)}(T) + E^{(2)}(T)Q(T), \\ &\cdots \\ N^{(s-1)}(T) &= (E \cdot Q)^{(s-1)}(T) = \sum_{i=0}^{s-1} E^{(i)}(T)Q^{(s-1-i)}(T) \end{aligned} \tag{3.7}$$

This motivates the following algorithm.

- Search for nonzero polynomials $E(T)$, $N(T)$ 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:

$$\begin{aligned} N(x) &= E(x)\ell^{(0)}(x) \\ N^{(1)}(x) &= E(x)\ell^{(1)}(x) + E^{(1)}(x)\ell^{(0)}(x) \\ &\cdots \\ N^{(s-1)}(x) &= \sum_{i=0}^{s-1} E^{(i)}(x)\ell^{(s-1-i)}(x) \end{aligned}$$

  This is a collection of sq homogeneous linear equations in $(sq-d)/2+1+(sq+d)/2+1 > sq$ unknowns (the coefficients of E and N). Thus a nonzero solution $E(T), N(T)$ exists. Take any such nonzero solution.

- Given E, N as above, output N/E.

To see correctness, take any Q such that $\Delta(\ell, \mathrm{Enc}_{s,d,m,q}(Q)) < \delta/2$. Observe that for any x where $\ell(x) = \mathrm{Enc}_{s,d,m,q}(Q)(x)$, the system of equations (3.7) is satisfied at $T = x$, and hence the polynomial $N(T) - E(T)Q(T)$ has a root with multiplicity s at x. Thus $\sum_{x \in \mathbb{F}_q} \mathrm{mult}(N - EQ, x) > (1 - \delta/2)sq = (sq + d)/2$. Since $\deg(N - EQ) \leq (sq + d)/2$, we conclude that the polynomial $N - EQ$ must be identically 0, and hence $Q(T) = N(T)/E(T)$, as desired (here we used the fact that E, N are not both identically 0). In particular $E \mid N$, and N/E is a polynomial.

Solving the noisy system. We now show how to solve the noisy system of linear equations efficiently. For m and s constant, this is a system of constantly many $(O(s)^m)$ 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 $\exp(s^m) \cdot \operatorname{poly} \log q$.

This is somewhat unsatisfactory. We point out some special situations where solving these noisy linear equations can be done in (optimal) time $\operatorname{poly}(s^m, \log q)$. Observe that our task is of the form: "Given a function $r : B \to \mathbb{F}_q$, find all polynomials $R(X_1, \ldots, X_m) \in \mathbb{F}_q[X_1, \ldots, X_m]$ of degree $< s$ such that $R(\mathbf{b}) = r(\mathbf{b})$ for at least α-fraction of the $\mathbf{b} \in B$". Thus, this is a problem of noisy polynomial interpolation. As observed in Step 3 of the analysis of the main local self-correction algorithm, there is a linear map which puts S^m and B in bijection. This linear map gives rise to a degree-preserving correspondence between polynomials evaluated on S^m and polynomials evaluated on B. Thus, our task of doing noisy polynomial interpolation (for polynomials of degree $< s$) on B reduces to noisy polynomial interpolation (for polynomials of degree $< s$) on S^m. This brings us into familiar territory. For certain fields $\mathbb{F}_q$, and certain choices of the set S, there are known efficient algorithms for doing this. In particular, if q is a power of p, and S is an $\mathbb{F}_p$-subspace of $\mathbb{F}_q$, given a function $r : S^m \to \mathbb{F}_q$, one can recover the unique degree $< s$ polynomial R that agrees with r in at least $(1 + s/q)/2$-fraction of the points of S^m in time $\operatorname{poly}(|S|^m, \log q)$. If $q > 4s$, then this fraction is at most 3/5, and this suffices for application to the local self-correcting algorithm of the previous section.

3.4 Discussion

Multiplicity codes are a natural and basic family of codes. Their similarity to multivariate polynomial codes endows them with a useful geometric structure, and their better rate/distance tradeoff allows them to carry that usefulness into new combinatorial regimes.

There are a number of questions related to multiplicity codes that invite further exploration.

- Because of their high rate, multiplicity codes seem to have potential for use in practice. It will be very interesting to further explore the practicality of these codes. Below we give few examples of concrete (rounded) code parameters.

Rate	Length	Queries	Alphabet	Code parameters
0.75	$28,000$	500	2^{50}	$q = 167,\ m = 2,\ s = 3,\ d = 497$
0.75	$100,000$	1000	2^{50}	$q = 331,\ m = 2,\ s = 3,\ d = 989$
0.70	$20,000,000$	$10,000$	2^{1000}	$q = 277,\ m = 3,\ s = 8,\ d = 2207$
0.83	$400,000,000$	$100,000$	2^{200}	$q = 19,997,\ m = 2,\ s = 5,\ d = 99,979$

- For every $\epsilon > 0$ multiplicity codes yield positive-rate $O(n^{\epsilon})$-query LDCs tolerating a constant fraction of errors. It is very interesting to see if one can reduce the query complexity of positive rate LDCs even further. The only lower bound we currently have in this regime is $\Omega(\log n)$.

  If we are willing to relax the requirement of recovering from a constant fraction of errors, and consider a smaller (but nontrivial) fraction of errors, then one can get multiplicity codes of positive rate which can be locally self-corrected with $\exp\left(\sqrt{\log n \log\log n}\right)$ queries.[1]

- Finally, it would be interesting to see if multiplicity codes can be useful in the various other settings where multivariate polynomial codes have proven useful.

[1]Discrepancies between local decodability and smoothness have been observed in the past, e.g., in [IKOS04] it is noted that m-variate binary Reed Muller codes of degree $d > m/2$ are smooth codes with non-trivial locality and rate close to 1. These codes however have poor distance, and thus are not locally decodable from a constant fraction of errors.

Chapter 4

Kakeya Sets

4.1 A Lower Bound on the Size of Kakeya Sets

Let $\mathbb{F}_q$ denote the finite field of cardinality q. A set $K \subseteq \mathbb{F}_q^n$ is said to be a *Kakeya set* if it "contains a line in every direction". In other words, for every "direction" $\mathbf{b} \in \mathbb{F}_q^n$ there should exist an "offset" $\mathbf{a} \in \mathbb{F}_q^n$ such that the "line" through $\mathbf{a}$ in direction $\mathbf{b}$, i.e., the set $\{\mathbf{a} + t\mathbf{b} | t \in \mathbb{F}_q\}$, is contained in K. A question of interest in combinatorics/algebra/geometry, posed originally by Wolff [Wol99], is: "What is the size of the smallest Kakeya set, for a given choice of q and n?"

The trivial upper bound on the size of a Kakeya set is q^n and this can be improved to roughly $\frac{1}{2^{n-1}}q^n$ (precisely the bound is $\frac{1}{2^{n-1}}q^n + O(q^{n-1})$, see [SS08] for a proof of this bound due to Dvir). An almost trivial lower bound is $q^{n/2}$ (every Kakeya set "contains" at least q^n lines, but there are at most $|K|^2$ lines that intersect K at least twice). Till recently even the exponent of q was not known precisely (see [Dvi08] for details of work prior to 2008). This changed with the breakthrough result of [Dvi08] (combined with an observation of Alon and Tao) who showed that for every n, $|K| \geq c_n q^n$, for some constant c_n depending only on n.

Subsequently the work [SS08] explored the growth of the constant c_n as a function of n. The result of [Dvi08] shows that $c_n \geq 1/n!$, and [SS08] improve this bound to

show that $c_n \geq 1/(2.6)^n$. This still leaves a gap between the upper bound and the lower bound and we effectively close this gap.

Theorem 4.1.1 *If K is a Kakeya set in $\mathbb{F}_q^n$ then $|K| \geq \frac{1}{2^n} q^n$.*

Note that our bound is tight to within a $2 + o(1)$ multiplicative factor as long as $q = \omega(2^n)$ and in particular when $n = O(1)$ and $q \to \infty$.

4.2 A Nearly Optimal Lower Bound on Kakeya Sets

We now give a lower bound on the size of Kakeya sets in $\mathbb{F}_q^n$. We implement the plan described in Section 1.6. Specifically, in Proposition 4.2.1 we show that we can find a somewhat low degree non-zero polynomial that vanishes with high multiplicity on any given Kakeya set, where the degree of the polynomial grows with the size of the set. Next, in Claim 4.2.3 we show that the homogenous part of this polynomial vanishes with fairly high multiplicity everywhere in $\mathbb{F}_q^n$. Using the strengthened Schwartz-Zippel lemma, we conclude that the homogenous polynomial is identically zero if the Kakeya set is too small, leading to the desired contradiction. The resulting lower bound (slightly stronger than Theorem 4.1.1) is given in Theorem 4.2.2.

Proposition 4.2.1 *Given a set $K \subseteq \mathbb{F}^n$ and non-negative integers m, d such that*

$$\binom{m+n-1}{n} \cdot |K| < \binom{d+n}{n},$$

there exists a non-zero polynomial $P = P_{m,K} \in \mathbb{F}[\mathbf{X}]$ of total degree at most d such that $\mathrm{mult}(P, \mathbf{a}) \geq m$ *for every $\mathbf{a} \in K$.*

Proof The number of possible monomials in P is $\binom{d+n}{n}$. Hence there are $\binom{d+n}{n}$ degrees of freedom in the choice for the coefficients for these monomials. For a given

point $\mathbf{a}$, the condition that $\text{mult}(P, \mathbf{a}) \geq m$ imposes $\binom{m+n-1}{n}$ homogeneous linear constraints on the coefficients of P. Since the total number of (homogeneous) linear constraints is $\binom{m+n-1}{n} \cdot |K|$, which is strictly less than the number of unknowns, there is a nontrivial solution.

■

Theorem 4.2.2 *If $K \subseteq \mathbb{F}_q^n$ is a Kakeya set, then $|K| \geq \left(\frac{q}{2-1/q}\right)^n$.*

Proof Let ℓ be a large multiple of q and let

$$m = 2\ell - \ell/q$$

$$d = \ell q - 1.$$

These three parameters (ℓ, m and d) will be used as follows: d will be the bound on the degree of a polynomial P which vanishes on K, m will be the multiplicity of the zeros of P on K and ℓ will be the multiplicity of the zeros of the homogenous part of P which we will deduce by restricting P to lines passing through K.

Note that by the choices above we have $d < \ell q$ and $(m - \ell)q > d - \ell$. A particular form we will use later is that $(m - w)q > d - w$ for every $w \leq \ell$ (using $q \geq 1$). We prove below that

$$|K| \geq \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}} \geq \alpha^n$$

where $\alpha \to \frac{q}{2-1/q}$ as $\ell \to \infty$.

Assume for contradiction that $|K| < \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}}$. Then, by Proposition 4.2.1 there exists $d^* \leq d$ and a non-zero polynomial $P(\mathbf{X}) \in \mathbb{F}[\mathbf{X}]$ of total degree exactly d^* such that $\text{mult}(P, \mathbf{x}) \geq m$ for every $\mathbf{x} \in K$. Note that $d^* \geq \ell$ since $d^* \geq m$ (since P is nonzero and vanishes to multiplicity $\geq m$ at some point), and $m \geq \ell$ by our choice of m. Let $H_P(\mathbf{X})$ be the homogeneous part of $P(\mathbf{X})$ of degree d^*. Note that $H_P(\mathbf{X})$ is nonzero. The following claim shows that H_P vanishes to multiplicity ℓ at each point of $\mathbb{F}_q^n$.

Claim 4.2.3 *For each* $\mathbf{b} \in \mathbb{F}_q^n$.

$$\mathrm{mult}(H_P, \mathbf{b}) \geq \ell.$$

Proof Fix $\mathbf{i}$ with $\mathrm{wt}(\mathbf{i}) = w \leq \ell - 1$. Let $Q(\mathbf{X}) = P^{(\mathbf{i})}(\mathbf{X})$. Let d' be the degree of the polynomial $Q(\mathbf{X})$, and note that $d' \leq d^* - w$.

Let $\mathbf{a} = \mathbf{a}(\mathbf{b})$ be such that $\{\mathbf{a} + t\mathbf{b} | t \in \mathbb{F}_q\} \subset K$. Then for all $t \in \mathbb{F}_q$, by Lemma 2.1.4, $\mathrm{mult}(Q, \mathbf{a} + t\mathbf{b}) \geq m - w$. Since $w \leq \ell - 1$ and $(m - \ell) \cdot q > d^* - \ell$, we get that $(m - w) \cdot q > d^* - w \geq d'$.

Let $Q_{\mathbf{a},\mathbf{b}}(T)$ be the polynomial $Q(\mathbf{a} + T\mathbf{b}) \in \mathbb{F}_q[T]$. Then $Q_{\mathbf{a},\mathbf{b}}(T)$ is a univariate polynomial of degree at most d', and by Corollary 2.1.6, it vanishes at each point of $\mathbb{F}_q$ with multiplicity $m - w$. Since

$$(m - w) \cdot q > d^* - w \geq \deg(Q_{\mathbf{a},\mathbf{b}}(T)),$$

we conclude that $Q_{\mathbf{a},\mathbf{b}}(T) = 0$. Hence the coefficient of $T^{d'}$ in $Q_{\mathbf{a},\mathbf{b}}(T)$ is 0. Let H_Q be the homogenous component of Q of highest degree. Observe that the coefficient of $T^{d'}$ in $Q_{\mathbf{a},\mathbf{b}}(T)$ is $H_Q(\mathbf{b})$. Hence $H_Q(\mathbf{b}) = 0$.

Now, if $(H_P)^{(\mathbf{i})}(\mathbf{X}) = 0$, then $(H_P)^{(\mathbf{i})}(\mathbf{b}) = 0$. Else $H_Q(\mathbf{X}) = (H_P)^{(\mathbf{i})}(\mathbf{X})$ (by Item 3 of Proposition 2.1.3), and hence as before $(H_P)^{(\mathbf{i})}(\mathbf{b}) = H_Q(\mathbf{b}) = 0$. Since this is true for all $\mathbf{i}$ of weight at most $\ell - 1$, we have that $\mathrm{mult}(H_P, \mathbf{b}) \geq \ell$. ■

Applying Corollary 2.1.8, and noting that $\ell q^n > d^* q^{n-1}$, we conclude that $H_P(\mathbf{X}) = 0$. This contradicts the fact that $P(\mathbf{X})$ is a nonzero polynomial.

Hence,

$$|K| \geq \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}}$$

Now, by our choice of d and m,

$$\frac{\binom{d+n}{n}}{\binom{m+n-1}{n}} = \frac{\binom{\ell q-1+n}{n}}{\binom{2\ell-\ell/q+n-1}{n}} = \frac{\prod_{i=1}^{n}(\ell q - 1 + i)}{\prod_{i=1}^{n}(2\ell - \ell/q - 1 + i)}$$

Since this is true for all ℓ such that ℓ is a multiple of q, we get that

$$|K| \geq \lim_{\ell \to \infty} \prod_{i=1}^{n} \left(\frac{q - 1/l + i/l}{2 - 1/q - 1/l + i/l} \right) = \left(\frac{q}{2 - 1/q} \right)^n$$

■

4.3 An Upper Bound on the Size of Kakeya Sets

We include here Dvir's proof giving a non-trivial upper bound on the size of Kakeya sets in fields of odd characteristic. For the case of even characteristic we complement their results by using a variation of their construction.

Theorem 4.3.1 *For every $n \geq 2$, and field $\mathbb{F}$, there exists a Kakeya set in $\mathbb{F}^n$ of cardinality at most $2^{-(n-1)} \cdot q^n + O(q^{n-1})$.*

Proof We consider two cases depending on whether $\mathbb{F}$ is of odd or even characteristic.

Odd characteristic: Let $D_n = \{\langle \alpha_1, \ldots, \alpha_{n-1}, \beta \rangle | \alpha_i, \beta \in \mathbb{F}, \alpha_i + \beta^2 \text{ is a square}\}$. Now let $K_n = D_n \cup (\mathbb{F}^{n-1} \times \{0\})$ where $\mathbb{F}^{n-1} \times \{0\}$ denotes the set $\{\langle \mathbf{a}, 0 \rangle | \mathbf{a} \in \mathbb{F}^{n-1}\}$. We claim that K_n is a Kakeya set of the appropriate size.

Consider a direction $\mathbf{b} = \langle b_1, \ldots, b_n \rangle$. If $b_n = 0$, for $\mathbf{a} = \langle 0, \ldots, 0 \rangle$ we have that $\mathbf{a} + t\mathbf{b} \in \mathbb{F}^{n-1} \times \{0\} \subseteq K_n$. The more interesting case is when $b_n \neq 0$. In this case let $\mathbf{a} = \langle (b_1/(2b_n))^2, \ldots, (b_{n-1}/(2b_n))^2, 0 \rangle$. The point $\mathbf{a} + t\mathbf{b}$ has coordinates $\langle \alpha_1, \ldots, \alpha_{n-1}, \beta \rangle$ where $\alpha_i = (b_i/(2b_n))^2 + tb_i$ and $\beta = tb_n$. We have $\alpha_i + \beta^2 = (b_i/(2b_n) + tb_n)^2$ which is a square for every i and so $\mathbf{a} + t\mathbf{b} \in D_n \subseteq K_n$. This proves that K_n is indeed a Kakeya set.

Finally we verify that the size of K_n is as claimed. First note that the size of D_n is exactly $q \cdot ((q+1)/2)^{n-1} = 2^{-(n-1)} q^n + O(q^{n-1})$ (q choices for β and $(q+1)/2$ choices

for each $\alpha_i + \beta^2$). The size of K_n is at most $|D_n| + q^{n-1} = 2^{-(n-1)}q^n + O(q^{n-1})$ as claimed.

Even characteristic: This case is handled similarly with minor variations in the definition of K_n. Specifically, we let $K_n = E_n = \{\langle \alpha_1, \ldots, \alpha_{n-1}, \beta\rangle | \alpha_i, \beta \in \mathbb{F}, \exists \gamma_i \in \mathbb{F}$ such that $\alpha_i = \gamma_i^2 + \gamma_i\beta\}$. (As we see below E_n contains $\mathbb{F}^{n-1} \times \{0\}$ and so there is no need to set $K_n = E_n \cup \mathbb{F}^{n-1} \times \{0\}$.)

Now consider direction $\mathbf{b} = \langle b_1, \ldots, b_n\rangle$. If $b_n = 0$, then let $\mathbf{a} = 0$. We note that $\mathbf{a} + t\mathbf{b} = \langle tb_1, \ldots, tb_{n-1}, 0\rangle = \langle \gamma_1^2 + \beta\gamma_1, \ldots \gamma_{n-1}^2 + \beta\gamma_{n-1}, \beta\rangle$ for $\beta = 0$ and $\gamma_i = \sqrt{tb_i} = (tb_i)^{q/2}$. We conclude that $\mathbf{a} + t\mathbf{b} \in E_n$ for every $t \in \mathbb{F}$ in this case. Now consider the case where $b_n \neq 0$. Let $\mathbf{a} = \langle (b_1/b_n)^2, \ldots, (b_{n-1}/b_n)^2, 0\rangle$. The point $\mathbf{a} + t\mathbf{b}$ has coordinates $\langle \alpha_1, \ldots, \alpha_{n-1}, \beta\rangle$ where $\alpha_i = (b_i/b_n)^2 + tb_i$ and $\beta = tb_n$. For $\gamma_i = (b_i/b_n)$, $\gamma_i^2 + \gamma_i\beta = (b_1/b_n)^2 + tb_i = \alpha_i$. Hence $\mathbf{a} + t\mathbf{b} \in E_n = K_n$.

It remains to compute the size of E_n. The number of points of the form $\langle \alpha_1, \ldots, \alpha_{n-1}, 0\rangle \in E_n$ is exactly q^{n-1}. We now determine the size of $\langle \alpha_1, \ldots, \alpha_{n-1}, \beta\} \in E_n$ for fixed $\beta \neq 0$. We first claim that the set $\{\gamma^2 + \beta\gamma | \gamma \in \mathbb{F}\}$ has size exactly $q/2$. This is so since for every $\gamma \in \mathbb{F}$, we have $\gamma^2 + \beta\gamma = \tau^2 + \beta\tau$ for $\tau = \gamma + \beta \neq \gamma$, and so the map $\gamma \mapsto \gamma^2 + \beta\gamma$ is a 2-to-1 map on its image. Thus, for $\beta \neq 0$, the number of points of the form $\langle \alpha_1, \ldots, \alpha_{n-1}, \beta\}$ in E_n is exactly $(q/2)^{n-1}$. We conclude that E_n has cardinality $(q-1) \cdot (q/2)^{n-1} + q^{n-1} = 2^{-(n-1)}q^n + O(q^{n-1})$. ■

We remark that for the case of odd characteristic, one can also use a recursive construction, replacing the set $\mathbb{F}^{n-1} \times \{0\}$ by $K_{n-1} \times \{0\}$. This would reduce the constant in the $O(q^{n-1})$ term, but not alter the leading term. Also we note that the construction used in the even case essentially also works in the odd characteristic case. Specifically the set $E_n \cup \mathbb{F}^{n-1} \times \{0\}$ is a Kakeya set also for odd characteristic. Its size can also be argued to be $2^{-(n-1)} \cdot q^n + O(q^{n-1})$.

Chapter 5

Mergers and Extractors with Sublinear Entropy Loss

5.1 Statistical Kakeya for Curves

Next we extend the results of the previous section to a form conducive to analyze the mergers of Dvir and Wigderson [DW08]. The extension changes two aspects of the consideration in Kakeya sets, that we refer to as "statistical" and "curves". We describe these terms below.

In the setting of Kakeya sets we were given a set K such that for *every* direction, there was a line in that direction such that *every* point on the line was contained in K. In the *statistical* setting we replace both occurrences of the "every" quantifier with a weaker "for many" quantifier. So we consider sets that satisfy the condition that for many directions, there exists a line in that direction intersecting K in many points.

A second change we make is that we now consider curves of higher degree and not just lines. We also do not consider curves in various *directions*, but rather curves passing through a given set of special points. We start with formalizing the terms "curves", "degree" and "passing through a given point".

A *curve of degree* Λ *in* $\mathbb{F}_q^n$ is a tuple of polynomials $C(X) = (C_1(X), \ldots, C_n(X)) \in \mathbb{F}_q[X]^n$ such that $\max_{i \in [n]} \deg(C_i(X)) = \Lambda$. A curve C naturally defines a map from $\mathbb{F}_q$ to $\mathbb{F}_q^n$. For $\mathbf{x} \in \mathbb{F}_q^n$, we say that a curve C *passes through* $\mathbf{x}$ if there is a $t \in \mathbb{F}_q$ such that $C(t) = \mathbf{x}$.

We now state and prove our statistical version of the Kakeya theorem for curves.

Theorem 5.1.1 (Statistical Kakeya for curves) *Let $\lambda > 0, \eta > 0$. Let $\Lambda > 0$ be an integer such that $\eta q > \Lambda$. Let $S \subseteq \mathbb{F}_q^n$ be an arbitrary set satisfying $|S| = \lambda q^n$. Let $K \subseteq \mathbb{F}_q^n$ be such that for each $\mathbf{x} \in S$, there exists a curve $C_{\mathbf{x}}$ of degree at most Λ that passes through $\mathbf{x}$, and intersects K in at least ηq points. Then,*

$$|K| \geq \left(\frac{\lambda q}{\Lambda \left(\frac{\lambda q - 1}{\eta q} \right) + 1} \right)^n .$$

In particular, if $\lambda \geq \eta$ we get that $|K| \geq \left(\frac{\eta q}{\Lambda + 1} \right)^n$.

Observe that when $\lambda = \eta = 1$, and $\Lambda = 1$, we get the same bound as that for Kakeya sets as obtained in Theorem 4.2.2.

Proof Let ℓ be a large integer and let

$$d = \lambda \ell q - 1$$

$$m = \Lambda \frac{\lambda \ell q - \ell}{\eta q} + \ell.$$

By our choice of m and d, we have $\eta q(m - (\ell - 1)) > \Lambda(d - (\ell - 1))$. Since $\eta q > \Lambda$, we have that for all w such that $0 \leq w \leq \ell - 1$, $\eta q(m - w) > \Lambda(d - w)$. Just as in the proof of Theorem 4.2.2, we will prove that

$$|K| \geq \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}} \geq \alpha^n$$

where $\alpha \to \frac{\lambda q}{\Lambda \left(\frac{\lambda q - 1}{\eta q} \right) + 1}$ as $\ell \to \infty$.

Assume for contradiction that $|K| < \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}}$. Then, as before, by Proposition 4.2.1 there exists a non-zero polynomial $P(\mathbf{X}) \in \mathbb{F}_q[\mathbf{X}]$ of total degree d^*, where $d^* \leq d$, such that $\text{mult}(P, \mathbf{a}) \geq m$ for every $\mathbf{a} \in K$. We will deduce that in fact P must vanish on all points in S with multiplicity ℓ. We will then get the desired contradiction from Corollary 2.1.8.

Claim 5.1.2 *For each* $\mathbf{x}_0 \in S$,

$$\text{mult}(P, \mathbf{x}_0) \geq \ell.$$

Proof Fix any $\mathbf{i}$ with $\text{wt}(\mathbf{i}) = w \leq \ell - 1$. Let $Q(\mathbf{X}) = P^{(\mathbf{i})}(\mathbf{X})$. Note that $Q(\mathbf{X})$ is a polynomial of degree at most $d^* - w$. By Lemma 2.1.4, for all points $\mathbf{a} \in K$, $\text{mult}(Q, \mathbf{a}) \geq m - w$.

Let $C_{\mathbf{x}_0}$ be the curve of degree Λ through $\mathbf{x}_0$, that intersects K in at least ηq points. Let $t_0 \in \mathbb{F}_q$ be such that $C_{\mathbf{x}_0}(t_0) = \mathbf{x}_0$. Let $Q_{\mathbf{x}_0}(T)$ be the polynomial $Q \circ C_{\mathbf{x}_0}(T) \in \mathbb{F}_q[T]$. Then $Q_{\mathbf{x}_0}(T)$ is a univariate polynomial of degree at most $\Lambda(d^* - w)$. By Corollary 2.1.6, for all points $t \in \mathbb{F}_q$ such that $C_{\mathbf{x}_0}(t) \in K$, $Q_{\mathbf{x}_0}(T)$ vanishes at t with multiplicity $m - w$. Since the number of such points t is at least ηq, we get that $Q_{\mathbf{x}_0}(T)$ has at least $\eta q(m - w)$ zeros (counted with multiplicity). However, by our choice of parameters, we know that

$$\eta q(m - w) > \Lambda(d - w) \geq \Lambda(d^* - w) \geq \deg(Q_{\mathbf{x}_0}(T)).$$

Since the degree of $Q_{\mathbf{x}_0}(T)$ is strictly less than the number of its zeros, $Q_{\mathbf{x}_0}(T)$ must be identically zero. Thus we get $Q_{\mathbf{x}_0}(t_0) = Q(C_{\mathbf{x}_0}(t_0)) = Q(\mathbf{x}_0) = 0$ Hence $P^{(\mathbf{i})}(\mathbf{x}_0) = 0$. Since this is true for all $\mathbf{i}$ with $\text{wt}(\mathbf{i}) \leq \ell - 1$, we conclude that $\text{mult}(P, \mathbf{x}_0) \geq \ell$. ■

Thus P vanishes at every point in S with multiplicity ℓ. As $P(\mathbf{X})$ is a non-zero polynomial, Corollary 2.1.8 implies that $\ell|S| \leq d^* q^{n-1}$. Hence $\ell \lambda q^n \leq d q^{n-1}$, which contradicts the choice of d.

Thus $|K| \geq \frac{\binom{d+n}{n}}{\binom{m+n-1}{n}}$. By choice of d and m,

$$|K| \geq \frac{\binom{\lambda\ell q-1+n}{n}}{\binom{\Lambda\frac{\lambda\ell q-1-(\ell-1)}{\eta q}+\ell+n-1}{n}}.$$

Picking ℓ arbitrarily large, we conclude that

$$|K| \geq \lim_{\ell\to\infty} \frac{\binom{\lambda\ell q-1+n}{n}}{\binom{\Lambda\frac{\lambda\ell q-1-(\ell-1)}{\eta q}+\ell+n-1}{n}} = \lim_{\ell\to\infty} \left(\frac{\ell\lambda q-1}{\ell\Lambda\left(\frac{\lambda q-1}{\eta q}\right)+\ell}\right)^n = \left(\frac{\lambda q}{\Lambda\left(\frac{\lambda q-1}{\eta q}\right)+1}\right)^n.$$

■

5.2 Improved Mergers

In this section we state and prove our main result on randomness mergers.

5.2.1 Definitions and Theorem Statement

We start by recalling some basic quantities associated with random variables. The statistical distance between two random variables X and Y taking values from a finite domain Ω is defined as

$$\max_{S\subseteq\Omega} |\mathbf{Pr}[\mathsf{X}\in S] - \mathbf{Pr}[\mathsf{Y}\in S]|.$$

We say that X is ϵ-close to Y if the statistical distance between X and Y is at most ϵ, otherwise we say that X and Y are ϵ-far. The min-entropy of a random variable X is defined as

$$H_\infty(\mathsf{X}) \triangleq \min_{x\in\mathrm{supp}(\mathsf{X})} \log_2\left(\frac{1}{\mathbf{Pr}[\mathsf{X}=x]}\right).$$

Note that the notion of having high min-entropy is closed under convex combinations: Specifically, if $0 \leq \alpha \leq 1$ and Y and $\mathbb{Z}$ are random variables supported on Ω with

min-entropy at least m and X is the random variable satisfying $\mathbf{Pr}[\mathsf{X} = x] = \alpha\mathbf{Pr}[\mathsf{Y} = x] + (1-\alpha)\mathbf{Pr}[\mathbb{Z} = x]$, then X also has min-entropy at least m.

We say that a random variable X is ϵ-close to having min-entropy m if there exists a random variable Y of min-entropy m such that X is ϵ-close to Y.

A "merger" of randomness takes a Λ-tuple of random variables and "merges" their randomness to produce a high-entropy random variable, provided the Λ-tuple is "somewhere-random" as defined below.

Definition 5.2.1 (Somewhere-random source) *For integers Λ and N a* simple (N, Λ)-somewhere-random source *is a random variable $\mathsf{A} = (\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda)$ taking values in S^Λ, where S is some finite set of cardinality 2^N, such that for some $i_0 \in [\Lambda]$, the distribution of A_{i_0} is uniform over S. A* (N, Λ)-somewhere-random source *is a convex combination of simple (N, Λ)-somewhere-random sources. (When N and Λ are clear from context we refer to the source as simply a "somewhere-random source".)*

We are now ready to define a merger.

Definition 5.2.2 (Merger) *For positive integer Λ and set S of size 2^N, a function $f : S^\Lambda \times \{0,1\}^d \to S$ is called an* (m, ϵ)-merger *(of (N, Λ)-somewhere-random sources), if for every (N, Λ) somewhere-random source $\mathsf{A} = (\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda)$ taking values in S^Λ, and for B being uniformly distributed over $\{0,1\}^d$, the distribution of $f((\mathsf{A}_1, \ldots, \mathsf{A}_\Lambda), \mathsf{B})$ is ϵ-close to having min-entropy m.*

A merger thus has five parameters associated with it: N, Λ, m, ϵ and d. The general goal is to give explicit constructions of mergers of (N, Λ)-somewhere-random sources for every choice of N and Λ, for as large an m as possible, and with ϵ and d being as small as possible. Known mergers attain $m = (1-\delta) \cdot N$ for arbitrarily small δ and our goal will be to achieve $\delta = o(1)$ as a function of N, while ϵ is an arbitrarily small positive real number. Thus our main concern is the growth of d as a function of N and Λ. Prior to this work, the best known bounds required either $d = \Omega(\log N + \log \Lambda)$ [DW08] or $d = \Omega(\Lambda)$ [LRVW03]. We only require $d = \Omega(\log \Lambda)$.

Theorem 5.2.3 *For every $\epsilon, \delta > 0$ and integers N, Λ, there exists a $((1-\delta)\cdot N, \epsilon)$-merger of (N, Λ)-somewhere-random sources, computable in polynomial time, with seed length*

$$d = \frac{1}{\delta} \cdot \log_2 \left(\frac{2\Lambda}{\epsilon} \right).$$

5.2.2 The Curve Merger of [DW08] and its Analysis

The merger that we consider is a very simple one proposed by Dvir and Wigderson [DW08], and we improve their analysis using our extended method of multiplicities. We note that they used the polynomial method in their analysis; and the basic method of multiplicities doesn't seem to improve their analysis.

The curve merger of [DW08], denoted f_{DW}, is obtained as follows. Let $q \geq \Lambda$ be a prime power, and let n be any integer. Fix some Λ distinct elements of $\mathbb{F}_q$ and let these be denoted $\gamma_1, \ldots, \gamma_\Lambda$. Let $c_i(T) \in \mathbb{F}_q[T]$ be the unique degree $\Lambda - 1$ polynomial with $c_i(\gamma_i) = 1$ and for all $j \neq i$, $c_i(\gamma_j) = 0$. Then the curve merger f_{DW} maps $(\mathbb{F}_q^n)^\Lambda \times \mathbb{F}_q$ to $\mathbb{F}_q^n$ as follows:

$$f_{\mathrm{DW}}((\mathbf{x}_1, \ldots, \mathbf{x}_\Lambda), u) = \sum_{i=1}^{\Lambda} c_i(u)\mathbf{x}_i,$$

where $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_\Lambda) \in (\mathbb{F}_q^n)^\Lambda$ and $u \in \mathbb{F}_q$. In other words, $f_{\mathrm{DW}}((\mathbf{x}_1, \ldots, \mathbf{x}_\Lambda), u)$ picks the (canonical) curve passing through $\mathbf{x}_1, \ldots, \mathbf{x}_\Lambda$ and outputs the uth point on the curve..

Theorem 5.2.4 *Let $q \geq \Lambda$ and A be a somewhere-random source taking values in $(\mathbb{F}_q^n)^\Lambda$. Let B be distributed uniformly over $\mathbb{F}_q$, with A, B independent. Let $\mathsf{C} = f_{\mathrm{DW}}(\mathsf{A}, \mathsf{B})$. Then for*

$$q \geq \left(\frac{2\Lambda}{\epsilon} \right)^{\frac{1}{\delta}},$$

C is ϵ-close to having min-entropy $(1-\delta) \cdot n \cdot \log_2 q$.

Theorem 5.2.3 easily follows from the above. We note that [DW08] proved a similar

theorem assuming $q \geq \text{poly}(n, \Lambda)$, forcing their seed length to grow logarithmically with n as well. We should also note at this point that a representation of a finite field $\mathbb{F}_q$ with q a power of 2 can be found efficiently (in time polynomial in $\log(q)$) by the results in [Sho88]. This makes our construction explicit even for small ϵ and δ.

Proof of Theorem 5.2.3: Let $q = 2^d$, so that $q \geq \left(\frac{2\Lambda}{\epsilon}\right)^{\frac{1}{\delta}}$, and let $n = N/d$. Then we may identify $\mathbb{F}_q$ with $\{0,1\}^d$ and $\mathbb{F}_q^n$ with $\{0,1\}^N$. Take f to be the function f_{DW} given earlier. Clearly f is computable in the claimed time. Theorem 5.2.4 shows that f has the required merger property. ■

We now prove Theorem 5.2.4.

Proof of Theorem 5.2.4: We first note that it suffices to prove the theorem for the case that A is a simple somewhere-random source. Once this is done, to handle the general case when A is a convex combination of simple somewhere-random sources, we can simply use the fact that $f_{\text{DW}}(\mathsf{A}, \mathbf{B})$ will be a convex combination of random variables that are ϵ-close to having high min-entropy and this notion is closed under convex combinations.

Let A be a simple somewhere-random source. Let $m = (1-\delta) \cdot n \cdot \log_2 q$. We wish to show that $f_{\text{DW}}(\mathsf{A}, \mathsf{B})$ is ϵ-close to having min-entropy m.

Suppose not. Then there is a set $K \subseteq \mathbb{F}_q^n$ with $|K| \leq 2^m = q^{(1-\delta)\cdot n} \leq \left(\frac{\epsilon q}{2\Lambda}\right)^n$ such that

$$\Pr_{\mathsf{A},\mathsf{B}}[f(\mathsf{A}, \mathsf{B}) \in K] \geq \epsilon.$$

To see why, consider the set K of the 2^m most 'popular' values in the distribution (i.e. those that have the highest probabilities). If this set is 'hit' w.p at most ϵ then the distribution is clearly ϵ close to having min-entropy at least m.

Suppose A_{i_0} is uniformly distributed over $\mathbb{F}_q^n$. Let A_{-i_0} denote the random variable

$$(\mathsf{A}_1, \ldots, \mathsf{A}_{i_0-1}, \mathsf{A}_{i_0+1}, \ldots, \mathsf{A}_\Lambda).$$

By an averaging argument, with probability at least $\lambda = \epsilon/2$ over the choice of A_{i_0},

we have

$$\Pr_{\mathsf{A}_{-i_0},\mathsf{B}}[f(\mathsf{A},\mathsf{B}) \in K] \geq \eta,$$

where $\eta = \epsilon/2$. Since A_{i_0} is uniformly distributed over $\mathbb{F}_q^n$, we conclude that there is a set S of cardinality at least λq^n such that for any $\mathbf{x} \in S$,

$$\Pr_{\mathsf{A},\mathsf{B}}[f(\mathsf{A},\mathsf{B}) \in K \mid \mathsf{A}_{i_0} = \mathbf{x}] \geq \eta.$$

By fixing the values of A_{-i_0} to preserve the above probability, we conclude that for each $\mathbf{x} \in S$, there is a $\mathbf{y} = \mathbf{y}(\mathbf{x}) = (\mathbf{y}_1, \ldots, \mathbf{y}_\Lambda)$ with $\mathbf{y}_{i_0} = x$ such that $\Pr_{\mathsf{B}}[f(\mathbf{y},\mathsf{B}) \in K] \geq \eta$. Define the degree $\Lambda - 1$ curve $C_{\mathbf{x}}(T) = f(\mathbf{y}(\mathbf{x}), T) = \sum_{j=1}^{\Lambda} \mathbf{y}_j c_j(T)$. Recall that the c_j's come from the definition of f_{DW}. Then $C_{\mathbf{x}}$ passes through $\mathbf{x}$, since $C_{\mathbf{x}}(\gamma_{i_0}) = \sum_{j=1}^{\Lambda} \mathbf{y}_j c_j(\gamma_{i_0}) = \mathbf{y}_{i_0} = \mathbf{x}$, and $\Pr_{\mathsf{B}\in\mathbb{F}_q}[C_{\mathbf{x}}(\mathsf{B}) \in K] \geq \eta$ by definition of $C_{\mathbf{x}}$.

Thus S and K satisfy the hypothesis of Theorem 5.1.1. We now conclude that

$$|K| \geq \left(\frac{\lambda q}{(\Lambda - 1)\left(\frac{\lambda q - 1}{\eta q}\right) + 1} \right)^n = \left(\frac{\epsilon q/2}{\Lambda - (\Lambda - 1)/\eta q} \right)^n > \left(\frac{\epsilon q}{2\Lambda} \right)^n.$$

This is a contradiction, and the proof of the theorem is complete. ■

The Somewhere-High-Entropy case: It is possible to extend the merger analysis given above also to the case of *somewhere-high-entropy* sources. In this scenario the source is comprised of blocks, one of which has min entropy at least r. One can then prove an analog of Theorem 5.2.4 saying that the output of f_{DW} will be close to having min entropy $(1-\delta)\cdot r$ under essentially the same conditions on q. The proof is done by hashing the source using a random linear function into a smaller dimensional space and then applying Theorem 5.2.4 (in a black box manner). The reason why this works is that the merger commutes with the linear map (for details see [DW08]). We do not give the details here since they are exactly the same as in [DW08]. We will not make use of this case in any of our other results.

5.3 Extractors with Sub-linear Entropy Loss

In this section we use our improved analysis of the Curve Merger to show the existence of an explicit extractor with logarithmic seed and sub linear entropy loss.

We will call a random variable X distributed over $\{0,1\}^n$ with min-entropy k an (n,k)-source.

Definition 5.3.1 (Extractor) *A function $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ is a (k,ϵ)-*extractor *if for every (n,k)-source X, the distribution of $E(\mathsf{X}, \mathsf{U}_d)$ is ϵ-close to uniform, where U_d is a random variable distributed uniformly over $\{0,1\}^d$, and X, U_d are independent. An extractor is called* explicit *if it can be computed in polynomial time.*

It is common to refer to the quantity $k - m$ in the above definition as the *entropy loss* of the extractor. The next theorem asserts the existence of an explicit extractor with logarithmic seed and sub-linear entropy loss.

Theorem 5.3.2 (Basic extractor with sub-linear entropy loss) *For every $c_1 \geq 1$, for all positive integers $k < n$ with $k \geq \log^2(n)$, there exists an explicit (k,ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ with*

$$d = O(c_1 \cdot \log(n)),$$

$$k - m = O\left(\frac{k \cdot \log\log(n)}{\log(n)}\right),$$

$$\epsilon = O\left(\frac{1}{\log^{c_1}(n)}\right).$$

The extractor of this theorem is constructed by composing several known explicit constructions of pseudorandom objects with the merger of Theorem 5.2.3. In Section 5.3.1 we describe the construction of our basic extractor. We then show, in Section 5.3.2 how to use the 'repeated extraction' technique of Wigderson and Zuckerman [WZ99] to boost this extractor and reduce the entropy loss to $k - m = O(k/\log^c n)$ for

any constant c (while keeping the seed logarithmic). The end result is the following theorem:

Theorem 5.3.3 (Final extractor with sub-linear entropy loss) *For every $c_1, c_2 \geq 1$, for all positive integers $k < n$, there exists an explicit (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ with*

$$d = O(c_1 c_2 \cdot \log(n)),$$

$$k - m = O\left(\frac{k}{\log^{c_2}(n)}\right),$$

$$\epsilon = O\left(\frac{1}{\log^{c_1}(n)}\right).$$

5.3.1 Proof of Theorem 5.3.2

Note that we may equivalently view an extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ as a randomized algorithm $E : \{0,1\}^n \to \{0,1\}^m$ which is allowed to use d uniformly random bits. We will present the extractor E as such an algorithm which takes 5 major steps.

Before giving the formal proof we give a high level description of our extractor. Our first step is to apply the lossless condenser of [GUV07] to output a string of length $2k$ with min entropy k (thus reducing our problem to the case $k = \Omega(n)$). The construction continues along the lines of [DW08]. In the second step, we partition our source (now of length $n' = 2k$) into $\Lambda = \log(n)$ consecutive blocks $X_1, \ldots, X_\Lambda \in \{0,1\}^{n'/\Lambda}$ of equal length. We then consider the Λ possible ways of partitioning the source into a prefix of j blocks and suffix of $\Lambda - j$ blocks for j between 1 and Λ. By a result of Ta-Shma [TS96b], after passing to a convex combination, one of these partitions is a (k', k_2) block source with k' being at least $k - O(k/\Lambda)$ and k_2 being at least poly-logarithmic in k. In the third step we use a block source extractor (from [RRS00]) on each one of the possible Λ partitions (using the same seed for each partition) to obtain a somewhere random source with block length k'. The fourth

step is to merge this somewhere random source into a single block of length k' and entropy $k' \cdot (1-\delta)$ with δ sub-constant. In view of our new merger parameters, and the fact that Λ (the number of blocks) is small enough, we can get away with choosing $\delta = \log\log(n)/\log(n)$ and keeping the seed logarithmic and the error poly-logarithmic. To finish the construction (the fifth step) we need to extract almost all the entropy from a source of length k' and entropy $k' \cdot (1-\delta)$. This can be done (using techniques from [RRS00]) with logarithmic seed and an additional entropy loss of $O(\delta \cdot k')$.

Proof of [: Theorem 5.3.2] We now formally prove Theorem 5.3.2. It would be convenient for us to assume during the proof that k is not too small. This is accomplished by noticing that for small k, there already exist very good extractors. Formally, using Theorem 5.3.14 below, we can assume that

$$k > 2^{\sqrt{\log n}}.$$

We begin by reducing to the case where $n = O(k)$ using the lossless condensers of [GUV07].

Theorem 5.3.4 (Lossless condenser [GUV07]) *For all positive integers $k < n$ with $k = \omega(\log(n))$, there exists an explicit function $C_{\mathrm{GUV}} : \{0,1\}^n \times \{0,1\}^{d'} \mapsto \{0,1\}^{n'}$ with $n' = 2k$, $d' = O(\log(n))$, such that for every (n,k)-source X, $C(\mathsf{X}, \mathsf{U}_{d'})$ is $(1/n)$-close to an (n',k)-source, where $\mathsf{U}_{d'}$ is distributed uniformly over $\{0,1\}^{d'}$, and $\mathsf{X}, \mathsf{U}_{d'}$ are independent.*

Step 1: Pick $\mathsf{U}_{d'}$ uniformly from $\{0,1\}^{d'}$. Compute $\mathsf{X}' = C_{\mathrm{GUV}}(\mathsf{X}, \mathsf{U}_{d'})$.

By the above theorem, X' is $(1/n)$-close to an (n',k)-source, where $n' = 2k$. Our next goal is to produce a *somewhere-block source*. We now define these formally.

Definition 5.3.5 (Block Source) *Let $\mathsf{X} = (\mathsf{X}_1, \mathsf{X}_2)$ be a random source over $\{0,1\}^{n_1} \times \{0,1\}^{n_2}$. We say that X is a (k_1,k_2)-*block source *if X_1 is an (n_1,k_1)-source*

and for each $x_1 \in \{0,1\}^{n_1}$ *the conditional random variable* $\mathsf{X}_2|\mathsf{X}_1 = x_1$ *is an* (n_2, k_2)*-source.*

Definition 5.3.6 (Somewhere-block source) *Let* $\mathsf{X} = (\mathsf{X}_1, \ldots, \mathsf{X}_\Lambda)$ *be a random variable such that each* X_i *is distributed over* $\{0,1\}^{n_{i,1}} \times \{0,1\}^{n_{i,2}}$*. We say that* X *is a* simple somewhere-(k_1, k_2)-block source *if there exists* $i \in [\Lambda]$ *such that* X_i *is a* (k_1, k_2)*-block source. We say that* X *is a* somewhere-(k_1, k_2)-block source *if* X *is a convex combination of simple somewhere random sources.*

We now state a result of Ta-Shma [TS96b] which converts an arbitrary source into a somewhere-block source. This is the first step in the proof of Theorem 1 on Page 44 of [TS96b] (Theorem 1 shows how convert any arbitrary source to a somewhere-block source, and then does more by showing how one could extract from such a source).

Let Λ be an integer and assume for simplicity of notation that n' is divisible by Λ. Let

$$\mathsf{X}' = (\mathsf{X}'_1, \ldots, \mathsf{X}'_\Lambda) \in \left(\{0,1\}^{n'/\Lambda}\right)^\Lambda$$

denote the partition of X' into Λ blocks. For every $1 \leq j < \Lambda$ we denote

$$\mathsf{Y}_j = (\mathsf{X}'_1, \ldots, \mathsf{X}'_j),$$

$$\mathsf{Z}_j = (\mathsf{X}'_{j+1}, \ldots, \mathsf{X}'_\Lambda),$$

Consider the function $B^\Lambda_{\mathrm{TS}} : \{0,1\}^{n'} \to (\{0,1\}^{n'})^{\Lambda-1}$, where

$$B^\Lambda_{\mathrm{TS}}(X') = ((\mathsf{Y}_1, \mathsf{Z}_1), (\mathsf{Y}_2, \mathsf{Z}_2), \ldots, (\mathsf{Y}_{\Lambda-1}, \mathsf{Z}_{\Lambda-1})).$$

The next theorem shows that the source $((\mathsf{Y}_j, \mathsf{Z}_j))_{j \in [\Lambda-1]}$ is close to a somewhere-block source.

Theorem 5.3.7 ([TS96b]) *Let* Λ *be an integer. Let* $k = k_1 + k_2 + s$*. Then the function* $B^\Lambda_{\mathrm{TS}} : \{0,1\}^{n'} \to (\{0,1\}^{n'})^{\Lambda-1}$ *is such that for any* (n', k)*-source* X'*, letting*

$\mathsf{X}'' = B^{\Lambda}_{\mathrm{TS}}(\mathsf{X}')$*, we have that* X'' *is* $O(n' \cdot 2^{-s})$*-close to a somewhere-*$(k_1 - O(n'/\Lambda), k_2)$*-block source.*

Step 2: Set $\Lambda = \log(n)$. Compute $X'' = (\mathsf{X}''_1, \mathsf{X}''_2, \ldots, \mathsf{X}''_\Lambda) = B^{\Lambda}_{\mathrm{TS}}(\mathsf{X}')$.

Plugging $k_2 = O(\log^4(n')) = O(\log^4(k))$, $s = O(\log n)$ and $k_1 = k - k_2 - s$ in the above theorem, we conclude that X'' is $n'^{-\Omega(1)}$-close to a somewhere-(k', k_2)-block source, where

$$k' = k_1 - O(n'/\log(n)) = k - k_2 - s - O(k/\log(n)) = k - O(k/\log(n)),$$

where for the last inequality we use the fact that $k > \log^2(n)$ and so both s and k_2 are bounded by $O(k/\log(n))$.

We next use the block source extractor from [RRS00] to convert the above somewhere-block source to a somewhere-random source.

Theorem 5.3.8 ([RRS00]) *Let* $n' = n_1 + n_2$ *and let* k', k_2 *be such that* $k_2 > \log^4(n_1)$*. Then there exists an explicit function* $E_{\mathrm{RSW}} : \{0,1\}^{n_1} \times \{0,1\}^{n_2} \times \{0,1\}^{d''} \mapsto \{0,1\}^{m''}$ *with* $m'' = k'$*,* $d'' = O(\log(n'))$*, such that for any* (k', k_2)*-block source* X*,* $E_{\mathrm{RSW}}(\mathsf{X}, \mathsf{U}_{d''})$ *is* $(n_1)^{-\Omega(1)}$*-close to the uniform distribution over* $\{0,1\}^{m''}$*, where* $\mathsf{U}_{d''}$ *is distributed uniformly over* $\{0,1\}^{d''}$*, and* $\mathsf{X}, \mathsf{U}_{d''}$ *are independent.*

Set $d'' = O(\log(n'))$ as in Theorem 5.3.8.

Step 3: Pick $\mathsf{U}_{d''}$ uniformly from $\{0,1\}^{d''}$. For each $j \in [\Lambda]$, compute $\mathsf{X}'''_j = E_{\mathrm{RSW}}(\mathsf{X}''_j, U_{d''})$.

By the above theorem, X''' is $n'^{-\Omega(1)}$-close to a somewhere-random source. We are now ready to use the merger M from Theorem 5.2.3. We invoke that theorem with entropy-loss $\delta = \log\log(n)/\log(n)$ and error $\epsilon = \frac{1}{\log^{c_1}(n)}$, and hence M has a seed

length of

$$d''' = O\left(\frac{1}{\delta}\log\frac{\Lambda}{\epsilon}\right) = O(c_1\log(n)).$$

Step 4: Pick $\mathsf{U}_{d''}$ uniformly from $\{0,1\}^{d'''}$. Compute $\mathsf{X}'''' = M(\mathsf{X}''', \mathsf{U}_{d'''})$.

By Theorem 5.2.3, X'''' is $O(\frac{1}{\log^{c_1}(n)})$-close to a $(k', (1-\delta)k')$-source, where $k' = k - O(k/\log k)$. Note that $\delta = o(1)$, and thus X'''' has nearly full entropy. We now apply an extractor for sources with extremely-high entropy rate, given by the following lemma.

Lemma 5.3.9 *For any k' and $\delta > 0$, there exists an explicit $(k'(1-\delta), k'^{-\Omega(1)})$-extractor $E_{\text{HIGH}} : \{0,1\}^{k'} \times \{0,1\}^{d''''} \mapsto \{0,1\}^{(1-3\delta)k'}$ with $d'''' = O(\log(k'))$.*

Proof The proof of this lemma will follow from Theorem 5.3.8. Initially, the input is partitioned into two blocks of length $k'(1-\delta) - 2\log^4 k'$ and $\delta k' + 2\log^4 k'$. Intuitively, this partition should be a block source, since fixing the first block cannot 'kill' all of the entropy of the source. Formally, using Lemma 6.2 from [RVW00], one knows that this partition is $1/k'$-close to a $(k'(1-2\delta) - 2\log^4 k', \log^4 k')$-block source. This block source is then passed through the block-source extractor of Theorem 5.3.8 which can extract $k'(1-2\delta) - 2\log^4 k' \geq k'(1-3\delta)$ bits with polynomially small error. ■

Step 5: Pick $\mathsf{U}_{d''''}$ uniformly from $\{0,1\}^{d''''}$. Compute $\mathsf{X}''''' = E_{\text{HIGH}}(\mathsf{X}'''', \mathsf{U}_{d''''})$. *Output* X'''''.

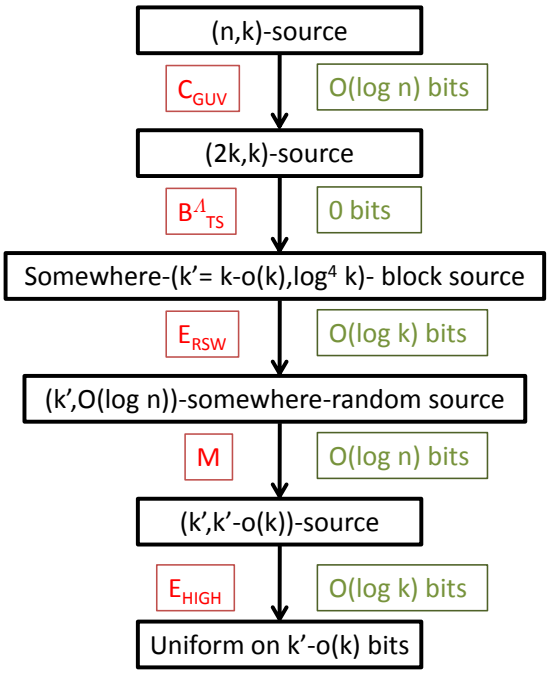
This completes the description of the extractor E. It remains to note that d, the total number of random bits used, is at most $d' + d'' + d''' + d'''' = O(c_1 \log n)$, The output X''''' is ϵ-close to uniformly distributed over

$$\{0,1\}^{(1-3\delta)k'} = \{0,1\}^{k - O(k\cdot\frac{\log\log n}{\log n})},$$

where ϵ can be bounded by the *sum* of all the errors in all of the above steps. The errors

introduced in all steps, other than the merger step (step 3), were polynomially small in k. Since $k > 2^{\sqrt{\log n}}$, this is negligible compared the $(\log^{c_1}(n))^{-1}$ error introduced in the merger step. Thus, the final error of the construction is dominated by this last quantity and so the error is as required. This completes the proof of Theorem 5.3.2. ■

We summarize the transformations in the following diagram:



5.3.2 Improving the Output Length by Repeated Extraction

We now use some ideas from [RRS00] and [WZ99] to extract an even larger fraction of the min-entropy out of the source. This will prove Theorem 5.3.3. We first prove a variant of the theorem with a restriction on k. This restriction will be later removed using known constructions of extractors for low min-entropy.

Theorem 5.3.10 (Explicit extractor with improved sub-linear entropy loss) *For every $c_1, c_2 \geq 1$, for all positive integers $k < n$ with $k = \log^{\omega(1)}(n)$, there exists an explicit (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ with*

$$d = O(c_1 c_2 \cdot \log(n)),$$

$$k - m = O\left(\frac{k}{\log^{c_2}(n)}\right),$$

$$\epsilon = O\left(\frac{1}{\log^{c_1}(n)}\right).$$

We first transform the extractor given in Theorem 5.3.2 into a *strong* extractor (defined below) via [RRS00, Theorem 8.2] (which gives a generic way of getting a strong extractor from any extractor). We then use a trick from [WZ99] that repeatedy uses the same extractor with independent seeds to extract the 'remaining entropy' from the source, thus improving the entropy loss.

Definition 5.3.11 *A* (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ *is strong if for every* (n, k)*-source* X*, the distribution of* $(E(\mathsf{X}, \mathsf{U}_d), \mathsf{U}_d)$ *is* ϵ*-close to the uniform distribution over* $\{0,1\}^{m+d}$*, where* U_d *is distributed uniformly over* $\{0,1\}^d$*, and* X, U_d *are independent.*

Theorem 5.3.12 ([RRS00, Theorem 8.2]) *Any explicit* (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ *can be transformed into an explicit strong* $(k, O(\sqrt{\epsilon}))$-extractor $E' : \{0,1\}^n \times \{0,1\}^{O(d)} \mapsto \{0,1\}^{m-d-2\log(1/\epsilon)-O(1)}$.

Theorem 5.3.13 ([WZ99, Lemma 2.4]) *Let* $E_1 : \{0,1\}^n \times \{0,1\}^{d_1} \mapsto \{0,1\}^{m_1}$ *be an explicit strong* (k, ϵ_1)-extractor*, and let* $E_2 : \{0,1\}^n \times \{0,1\}^{d_2} \mapsto \{0,1\}^{m_2}$ *be an explicit strong* $(k - (m_1 + r), \epsilon_2)$-extractor*. Then the function*

$$E_3 : \{0,1\}^n \times \left(\{0,1\}^{d_1} \times \{0,1\}^{d_2}\right) \mapsto \{0,1\}^{m_1+m_2}$$

defined by

$$E_3(x, y_1, y_2) = E_1(x, y_1) \circ E_2(x, y_2)$$

is a strong $(k, \epsilon_1 + \epsilon_2 + 2^{-r})$-extractor.

Proof [Theorem 5.3.10] Let E be the (k, ϵ)-extractor with seed $O(c_1 \log n)$ of Theorem 5.3.2. By Theorem 5.3.12, we get an explicit strong $(k, \sqrt{\epsilon})$-extractor E' with entropy loss $O(k\frac{\log\log n}{\log n})$. We now iteratively apply Theorem 5.3.13 as follows. Let $E^{(0)} = E'$. For each $1 < i \leq O(c_2)$, let $E^{(i)} : \{0,1\}^n \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}$ be the

strong (k, ϵ_i)-extractor produced by Theorem 5.3.13 when we take $E_1 = E^{(i-1)}$ and E_2 to be the strong $(k - m_{i-1} - c_1 \log n, 1/\log^{c_1}(n))$-extractor with seed length $O(c_1 \log n)$ given by Theorem 5.3.2 and Theorem 5.3.12. Thus,

$$d_i = O(ic_1 \log n).$$

$$\epsilon_i = O\left(\frac{i}{\log^{c_1}(n)}\right).$$

$$m_i = m_{i-1} + (k - m_{i-1} - c_1 \log n)\left(1 - O\left(\frac{\log\log n}{\log n}\right)\right).$$

Thus the entropy loss of $E^{(i)}$ is given by:

$$k - m_i = (k - m_{i-1})\left(1 - \left(1 - O\left(\frac{\log\log n}{\log n}\right)\right)\right) + O(c_1 \log n) = O\left(\frac{k \cdot \log\log(n)^i}{\log^i(n)}\right).$$

$E^{(O(c_2))}$ is the desired extractor. ■

Remark In fact [GUV07] and [RRV99] show how to extract *a*ll the min-entropy with polylogarithmic seed length. Combined with the lossless condenser of [GUV07] this gives an extractor that uses logarithmic seed to extract all the min-entropy from sources that have min-entropy rate at most $2^{O(\sqrt{\log n})}$.

Theorem 5.3.14 (Corollary of [GUV07, Theorem 4.21]) *For all positive integers $n \geq k$ such that $k = 2^{O(\sqrt{\log n})}$, and for all $\epsilon > 0$ there exists an explicit (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \mapsto \{0,1\}^m$ with $d = O(\log(n))$ and $m = k + d - 2\log(1/\epsilon) - O(1)$.*

This result combined with Theorem 5.3.10 gives an extractor with improved sublinear entropy loss that works for sources of all entropy rates, thus completing the proof of Theorem 5.3.3.

Chapter 6

Reed-Solomon Decoding

6.1 Bounds on the List Size for List-Decoding Reed-Solomon Codes

In this section, we give a simple algebraic proof of an upper bound on the list size for list-decoding Reed-Solomon codes within the Johnson radius.

Before stating and proving the theorem, we need some definitions. For a bivariate polynomial $P(X,Y) \in \mathbb{F}[X,Y]$, we define its (a,b)-degree to be the maximum of $ai+bj$ over all (i,j) such that the monomial X^iY^j appears in $P(X,Y)$ with a nonzero coefficient. Let $N(k,d,\theta)$ be the number of monomials X^iY^j which have $(1,k)$-degree at most d and $j \leq \theta d/k$. We have the following simple fact.

Fact 6.1.1 *For any $k < d$ and $\theta \in [0,1]$, $N(k,d,\theta) > \theta \cdot (2-\theta) \cdot \frac{d^2}{2k}$.*

Proof Letting $t = \lfloor \theta d/k \rfloor$, note that

$$N(k,d,\theta) = \sum_{j=0}^{t} (d+1-kj) = (t+1)(d+1) - k\binom{t+1}{2} = (t+1)(d+1-kt/2).$$

Using $t \leq \theta d/k \leq t+1$, we get $N(k,d,\theta) \geq (\theta d/k)(d+1-\theta d/2) > \theta \cdot (2-\theta) \cdot \frac{d^2}{2k}$. ■

Now we prove the main theorem of this section. The proof is an enhancement of the original analysis of the Guruswami-Sudan algorithm using the extended method of multiplicities.

Theorem 6.1.2 (List size bound for Reed-Solomon codes) *Let*

$$(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n) \in \mathbb{F}^2.$$

Let $R, \gamma \in [0,1]$ *with* $\gamma^2 > R$. *Let* $k = Rn$. *Let* $f_1(X), \ldots, f_L(X) \in \mathbb{F}[X]$ *be polynomials of degree at most* k, *such that for each* $j \in [L]$ *we have* $|\{i \in [n] : f_j(\alpha_i) = \beta_i\}| > \gamma n$. *Then* $L \leq \frac{2\gamma}{\gamma^2 - R}$.

Proof Let $\epsilon > 0$ be a parameter. Let $\theta = \frac{2}{\left(1+\frac{\gamma^2}{R}\right)}$. Let m be a large integer (to be chosen later), and let $d = (1+\epsilon) \cdot m \cdot \sqrt{\frac{nk}{\theta \cdot (2-\theta)}}$. We first interpolate a nonzero polynomial $P(X,Y) \in \mathbb{F}[X,Y]$ of $(1,k)$-degree at most d and Y-degree at most $\theta d/k$, that vanishes with multiplicity at least m at each of the points (α_i, β_i). Such a polynomial exists if $N(k,d,\theta)$, the number of monomials available, is larger than the number of homogeneous linear constraints imposed by the vanishing conditions:

$$\frac{m(m+1)}{2} \cdot n < N(k,d,\theta). \tag{6.1}$$

This can be made to hold by picking m sufficiently large, since by Fact 6.1.1,

$$N(k,d,\theta) > \theta \cdot (2-\theta)\frac{d^2}{2k} = \frac{(1+\epsilon)^2 m^2}{2} \cdot n.$$

Having obtained the polynomial $P(X,Y)$, we also view it as a univariate polynomial $Q(Y) \in \mathbb{F}(X)[Y]$ with coefficients in $\mathbb{F}(X)$, the field of rational functions in X.

Now let $f(X)$ be any polynomial of degree at most k such that, letting $I = \{i \in [n] : f(\alpha_i) = \beta_i\}$, $|I| \geq A$. We claim that the polynomial $Q(Y)$ vanishes at $f(X)$ with multiplicity at least $m - d/A$. Indeed, fix an integer $j < m - d/A$, and let $R_j(X) = Q^{(j)}(f(X)) = P^{(0,j)}(X, f(X))$. Notice that the degree of $R_j(X)$ is at most

d. By Proposition 2.1.5 and Lemma 2.1.4,

$$\mathrm{mult}(R_j, \alpha_i) \geq \mathrm{mult}(P^{(0,j)}, (\alpha_i, \beta_i)) \geq \mathrm{mult}(P, (\alpha_i, \beta_i)) - j.$$

Thus

$$\sum_{i \in I} \mathrm{mult}(R_j, \alpha_i) \geq |I| \cdot (m - j) \geq A \cdot (m - j) > d.$$

By Lemma 2.1.7, we conclude that $R_j(X) = 0$. Since this holds for every $j < m - d/A$, we conclude that $\mathrm{mult}(Q, f(X)) \geq m - d/A$.

We now complete the proof of the theorem. By the above discussion, for each $j \in [L]$, we know that $\mathrm{mult}(Q, f_j(X)) \geq m - \frac{d}{\gamma n}$. Thus, by Lemma 2.1.7 (applied to the nonzero polynomial $Q(Y) \in \mathbb{F}(X)[Y]$ and the set of evaluation points $S = \{f_j(X) : j \in [L]\}$)

$$\deg(Q) \geq \sum_{j \in [L]} \mathrm{mult}(Q, f(X)) \geq \left(m - \frac{d}{\gamma n}\right) \cdot L.$$

Since $\deg(Q) \leq \theta d/k$, we get,

$$\theta d/k \geq \left(m - \frac{d}{\gamma n}\right) \cdot L.$$

Using $d = (1+\epsilon) \cdot m \cdot \sqrt{\frac{nk}{\theta \cdot (2-\theta)}}$ and $\theta = \frac{2}{1+\frac{\gamma^2}{R}}$, we get,

$$\begin{aligned} L &\leq \frac{\theta}{k \cdot \frac{m}{d} - \frac{k}{\gamma n}} \\ &= \frac{\theta}{\frac{1}{1+\epsilon}\sqrt{\frac{k}{n} \cdot \theta \cdot (2-\theta)} - \frac{k}{\gamma n}} \\ &= \frac{1}{\frac{1}{1+\epsilon}\sqrt{R\left(\frac{2}{\theta} - 1\right)} - \frac{R}{\theta\gamma}} = \frac{1}{\frac{\gamma}{1+\epsilon} - \left(\frac{\gamma}{2} + \frac{R}{2\gamma}\right)}. \end{aligned}$$

Letting $\epsilon \to 0$, we get $L \leq \frac{2\gamma}{\gamma^2 - R}$, as desired. ■

Bibliography

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, January 1998.

[AS03] Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23:365–426, 2003.

[BET10] 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)*, 2010.

[BF90] 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.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

[BFLS91] 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.

[BFNW93] Laszlo Babai, Lance Fortnow, Naom Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[CB04] Yuval Cassuto and Jehoshua Bruck. A combinatorial bound on the list size. *Paradise Laboratory Technical report, California Institute of Technology*, 2004.

[CFL+10] Y. Meng Chee, T. Feng, S. Ling, H. Wang, and L. F. Zhang. Query-efficient locally decodable codes of subexponential length. Arxiv 1008.1617, 2010.

[DGY10] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. In *51st IEEE Symposium on Foundations of Computer Science (FOCS)*, 2010.

[DJK+02] 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.

[DKSS09a] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *FOCS*, pages 181–190, 2009.

[DKSS09b] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. *Electronic Colloquium on Computational Complexity (ECCC)*, 16:4, 2009.

[DS07] Z. Dvir and A. Shpilka. An improved analysis of linear mergers. *Comput. Complex.*, 16(1):34–59, 2007. (Extended abstract appeared in RANDOM 2005).

[Dvi08] Z. Dvir. On the size of Kakeya sets in finite fields. *J. AMS (to appear)*, 2008.

[Dvi10] Zeev Dvir. On matrix rigidity and locally self-correctable codes. In *26th IEEE Computational Complexity Conference (CCC)*, pages 102–113, 2010.

[DW08] Zeev Dvir and Avi Wigderson. Kakeya sets, new mergers and old extractors. In *FOCS*, pages 625–633. IEEE Computer Society, 2008.

[Efr09] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *41st ACM Symposium on Theory of Computing (STOC)*, pages 39–44, 2009.

[GKST02] 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.

[GR08] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.

[GSS00] 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.

[GUV07] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh-vardy codes. In *IEEE Conference on Computational Complexity*, pages 96–108. IEEE Computer Society, 2007.

[HKT08] J. W. P. Hirschfeld, G. Korchmaros, and F. Torres. *Algebraic Curves over a Finite Field (Princeton Series in Applied Mathematics)*. Princeton University Press, 2008.

[IKOS04] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *36th ACM Symposium on Theory of Computing (STOC)*, pages 262–271, 2004.

[IS10] 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.

[IW97] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR Lemma. *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 220–229, May 1997.

[KdW04] 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.

[KSY10] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:148, 2010.

[KSY11] *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*. ACM, 2011.

[KT00] 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.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, October 1992.

[Lip90] 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.

[LRVW03] Chi-Jen Lu, Omer Reingold, Salil Vadhan, and Avi Wigderson. Extractors: Optimal up to constant factors. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing"*, 2003.

[Oba02] 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.

[PV05] 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.

[Rag07] Prasad Raghavendra. A note on Yekhanin's locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR07-016, 2007.

[Ree54] Irving S. Reed. A class of multiple-error-correcting codes and the decoding scheme. *IEEE Transactions on Information Theory*, 4:38–49, 1954.

[RRS00] O. Reingold and and A. Wigderson R. Shaltiel. Extracting randomness via repeated condensing. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, 2000.

[RRV99] Ran Raz, Omer Reingold, and Salil P. Vadhan. Extracting all the randomness and reducing the error in trevisan's extractors. In *STOC*, pages 149–158, 1999.

[RT97] Michael Rozenbloom and Michael Tsfasman. Codes for the m-metric. *Problems Inform. Transmission*, 33:45–52, 1997.

[RVW00] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pages 3–13, November 2000.

[Sha92] Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, October 1992.

[Sha02] R. Shaltiel. Recent developments in extractors. *Bulletin of the European Association for Theoretical Computer Science*, 77:67–95, June 2002.

[Sho88] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 283–290, Washington, DC, USA, 1988. IEEE Computer Society.

[SS08] Shubhangi Saraf and Madhu Sudan. Improved lower bound on the size of Kakeya sets over finite fields. *Analysis and PDE*, 2008.

[STV99] 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.

[SU05] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

[Sud97] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

[Sud01] Madhu Sudan. Ideal error-correcting codes: Unifying algebraic and number-theoretic algorithms. In *AAECC*, pages 36–45, 2001.

[TS96a] A. Ta-Shma. On extracting randomness from weak random sources. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA*, pages 276–285, 1996.

[TS96b] A. Ta-Shma. *Refining Randomness*. PhD thesis, The Hebrew University, Jerusalem, Israel, 1996.

[WdW05] 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.

[Wol99] T. Wolff. Recent work connected with the Kakeya problem. In *Prospects in Mathematics*, pages 129–162, 1999.

[Woo07] David Woodruff. New lower bounds for general locally decodable codes. In *Electronic Colloquium on Computational Complexity (ECCC)*, TR07-006, 2007.

[WY05] 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.

[WZ99] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: explicit construction and applications. In *Combinatorica*, volume 19, pages 125–138, 1999.

[Xin03] Chaoping Xing. Nonlinear codes from algebraic curves improving the Tsfasman-Vladut-Zink bound. *IEEE Transactions on Information Theory*, 49(7):1653–1657, 2003.

[Yek08] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM*, 55:1–16, 2008.

[Yek10] Sergey Yekhanin. Locally decodable codes. *Foundations and trends in theoretical computer science*, 2010. to appear.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007.