High-rate codes with sublinear-time decoding

Swastik Kopparty, Institute for Advanced Study
and Shubhangi Saraf, Massachusetts Institute of Technology
and Sergey Yekhanin, Microsoft Research Silicon Valley

Locally decodable codes are error-correcting codes that admit efficient decoding algorithms; any bit of the original message can be recovered by looking at only a small number of locations of a corrupted codeword. The tradeoff between the rate of a code and the locality/efficiency of its decoding algorithms has been well studied, and it has widely been suspected that nontrivial locality must come at the price of low rate. A particular setting of potential interest in practice is codes of constant rate. For such codes, decoding algorithms with locality $O(k^\epsilon)$ were known only for codes of rate $\epsilon^{\Omega(1/\epsilon)}$, where k is the length of the message. Furthermore, for codes of rate $> 1/2$, no nontrivial locality had been achieved.

In this paper we construct a new family of locally decodable codes that have very efficient local decoding algorithms, and at the same time have rate approaching 1. We show that for every $\epsilon > 0$ and $\alpha > 0$, for infinitely many k, there exists a code C which encodes messages of length k with rate $1 - \alpha$, and is locally decodable from a constant fraction of errors using $O(k^\epsilon)$ queries and time.

These codes, which we call multiplicity codes, are based on evaluating multivariate polynomials and their derivatives. Multiplicity codes extend traditional multivariate polynomial codes; they inherit the local-decodability of these codes, and at the same time achieve better tradeoffs and flexibility in the rate and minimum distance.

Categories and Subject Descriptors: E.4 [**Coding and Information Theory**]

General Terms: Algorithms, Reliability, Theory

Additional Key Words and Phrases: multiplicity codes, sublinear-time algorithms, locally correctable codes, locally decodable codes, polynomials, derivatives

ACM Reference Format:
Swastik Kopparty, Shubhangi Saraf, Sergey Yekhanin, 2014. High-rate codes with sublinear-time decoding. *ACM Trans. Embedd. Comput. Syst.* 9, 4, Article 39 (March 2010), 21 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

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

Part of this work was done while the first two authors were interns at Microsoft Research Silicon Valley.
Author's addresses: S. Kopparty, Department of Mathematics & Department of Computer Science, Rutgers University, email: swastik.kopparty@rutgers.edu; S. Saraf, Department of Mathematics & Department of Computer Science, Rutgers University, email: shubhangi.saraf@rutgers.edu; Sergey Yekhanin, Microsoft Research Silicon Valley, yekhanin@microsoft.com.
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2010 ACM 1539-9087/2010/03-ART39 $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

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 coordinates. 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 [Yekhanin 2012] 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 paper, 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 2.4) 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.*

1.1. 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 [Reed 1954]. 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 [Beaver and Feigenbaum 1990; Lipton 1990; Lund et al. 1992; Shamir 1992; Babai et al. 1991b; Babai et al. 1991a; Arora and Safra 1998; Arora et al. 1998] as well as in other fundamental results in complexity theory [Babai et al. 1993; Impagliazzo and Wigderson 1997; Arora and Sudan 2003; Sudan et al. 1999; Shaltiel and Umans 2005].

Locally decodable codes were first formally defined in [Babai et al. 1991b; Katz and Trevisan 2000] (see also [Sudan et al. 1999]). 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 [Yekhanin 2008; Raghavendra 2007; Efremenko 2009; Dvir et al. 2010; Ben-Aroya et al. 2010; Itoh and Suzuki 2010; Chee et al. 2010] 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 [Katz and Trevisan 2000; Kerenidis and de Wolf 2004; Goldreich et al. 2002; Wehner and de Wolf 2005; Woodruff 2007; Deshpande et al. 2002; Obata 2002] on the problem of proving lower bounds on the length of locally decodable codes. In particular, it is known [Katz and Trevisan 2000] 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 [Dvir 2010] recently conjectured a lower bound on the length of some closely related objects called *locally correctable codes*. Precisely, the conjecture of [Dvir 2010] 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 correctable with query complexity $O(n^\epsilon)$ from a certain sub-constant fraction of errors. Dvir [Dvir 2010] 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 correctable from even a constant fraction of errors with $O(n^\epsilon)$ queries[1].

1.2. 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 correctable codes" over "large alphabets", which we now define. We have a set Σ (the "alphabet"), and we want to construct

[1][Dvir 2010] 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.

a subset $\mathcal{C}$ (the "code") of Σ^n, of size $|\Sigma|^k$ (we call k the "message length"), with the following local 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 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 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 correct it with essentially the same query complexity. Finally, we mention how general multiplicity codes are defined and some of the ideas that go into locally 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 $\binom{d+2}{2}$ monomials, and so the message length $k = \binom{d+2}{2}$. Thus the rate of this code is $\binom{d+2}{2}/q^2 \approx (1-\delta)^2/2$. Notice that this code cannot have rate more than $1/2$.

Local Correction of Reed-Muller codes. Given a received word $r \in (\mathbb{F}_q)^{q^2}$ such that r is close in Hamming distance to the codeword corresponding to $P(X,Y)$, let us recall how the classical local 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 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 $\binom{d+2}{2}/3$. Thus the rate of this code is $(\binom{d+2}{2}/3)/q^2 \approx 2(1-\delta)^2/3$.

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 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 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[2] $Q(T) = P(\mathbf{a} + \mathbf{b}T)$. The important 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

[2]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.

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 4, we present our local 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).

Applications of derivatives and multiplicities. The notions of derivative and multiplicity have played an important role in several prior works in coding theory and theoretical computer science. The "method of multiplicities" is a powerful combinatorial/algorithmic technique which has been developed and used in a number of contexts in recent years [Guruswami and Sudan 1999; Parvaresh and Vardy 2005; Guruswami and Rudra 2008; Saraf and Sudan 2008; Dvir et al. 2009]. It is a method for analyzing subsets of $\mathbb{F}_q^m$ by interpolating a polynomial that vanishes at each point of that subset with high multiplicity; this often yields a strengthening of the "polynomial method", which would analyze such a subset by interpolating a polynomial that simply vanishes at each point of that subset. Xing [Xing 2003] 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 [Woodruff and Yekhanin 2005] 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[3].

Subsequent work. Since the publication of the conference version of this work, there have been several other constructions of locally correctable codes achieving rate and query complexity similar to what is achieved by multiplicity codes: Guo-Kopparty-Sudan [Guo et al. 2013] (based on lifted Reed-Solomon codes), Guo [Guo 2013] (based on more general lifts of codes) and Hemenway-Ostrovsky-Wooters [Hemenway et al. 2013] (based on expander codes). There have also been further results on multiplicity codes themselves: list-decoding of univariate multiplicity codes [Guruswami and Wang 2011; Kopparty 2012], list-decoding and local list-decoding of multivariate multiplicity codes [Kopparty 2012], and improved local correction for multivariate multiplicity codes [Kopparty 2014].

Organization of this paper:. In the next section, we state our main theorems on the existence of locally decodable/correctable codes. In Section 3, we formally define multi-

[3] Some restrictions apply.

plicity 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 4 we give our local correction algorithms for multiplicity codes. In Section 5 we consider some other interesting settings of the parameters of multiplicity codes. Section 6 contains some concluding remarks.

2. 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 with 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 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 2.1 (*Locally Correctable Code*). A code $\mathcal{C} \subseteq \Sigma^n$ is said to be locally correctable from δ'-fraction errors with t queries if there is a randomized procedure A, such that:

— **Local 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 2.2 (*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 procedure 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 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 correctable.

We now state the two main theorems, which assert the existence of locally 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 2.3 (LOCALLY CORRECTABLE CODES OVER LARGE ALPHABETS). *For all constants $0 < \epsilon, \alpha < 1$, 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 at least $\delta = \epsilon\alpha/2$, and is locally correctable from $\delta/10$-fraction errors with $O(n^{\epsilon})$ queries.*

The next theorem is the analogue of Theorem 2.3 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 2.4 (LOCALLY CORRECTABLE CODES OVER SMALL ALPHABETS). *Let p be a prime power. For all constants $\epsilon, \alpha > 0$, there exists $\delta > \frac{\epsilon\alpha^3}{256\log(1/\alpha)}$, 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 correctable from $\delta/20$-fraction errors with $O(n^{\epsilon})$ queries.*

Remark 2.5. The codes in both the above theorems are efficiently constructible. Furthermore, both the local correction algorithms can be made to run in time $n^{O(\epsilon)}$.

The proofs of the theorems above appear in Section 3.3.

3. 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. 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.

3.1. 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 3.1 (*(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,

$$P(\mathbf{X}+\mathbf{Z}) = \sum_{\mathbf{i}} P^{(\mathbf{i})}(\mathbf{X})\mathbf{Z}^{\mathbf{i}}. \tag{1}$$

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

$$(\lambda P)^{(\mathbf{i})}(\mathbf{X}) = \lambda P^{(\mathbf{i})}(\mathbf{X}) \quad \text{and} \quad \mathrm{P}^{(\mathbf{i})}(\mathbf{X}) + \mathrm{Q}^{(\mathbf{i})}(\mathbf{X}) = (\mathrm{P}+\mathrm{Q})^{(\mathbf{i})}(\mathbf{X}). \tag{2}$$

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

Definition 3.2 (*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 main technical fact we will need about derivatives and multiplicities is a bound on the number of points that a low-degree polynomial can vanish on with high multiplicity. We state this lemma below. For an elementary proof in our notation, see [Dvir et al. 2009].

LEMMA 3.3. *Let $P \in \mathbb{F}[\mathbf{X}]$ be a nonzero polynomial of total degree at most d. Then for any finite $S \subseteq \mathbb{F}$,*

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

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

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

3.2. The definition of multiplicity codes

We now come to the definition of multiplicity codes.

Definition 3.4 (*Multiplicity code*). Let s, d, m be nonnegative integers and let q be a prime power. Let $\Sigma = \mathbb{F}_q^{\binom{m+s-1}{m}} = \mathbb{F}_q^{\{\mathbf{i}:\mathrm{wt}(\mathbf{i})<s\}}$. 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 $\langle P^{(\mathbf{i})}(\mathbf{a})\rangle_{\mathrm{wt}(\mathbf{i})<s} \in \Sigma$.

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:

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

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 3.5 (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 at least $\delta = 1 - \frac{d}{sq}$ 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 block-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 3.3, 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 paper, shows that multiplicity codes are locally correctable.

THEOREM 3.6 (MULTIPLICITY CODES ARE LOCALLY 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 a lower bound for the distance of $\mathcal{C}$. Suppose $q \geq \max\{10m, \frac{d+6s}{s}, 12(s+1)\}$. Then $\mathcal{C}$ is locally correctable from $\frac{\delta}{10}$-fraction errors with $(O(s)^m \cdot q)$-queries.*

The proof of this theorem appears in Section 4.2. In Section 4.3, we will show that the local 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.

3.3. Proof of the main theorems

We now show how to instantiate multiplicity codes to prove our main theorems on the existence of locally correctable codes with improved rate and query-complexity (assuming Theorem 3.6).

PROOF OF THEOREM 2.3. 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 block-length n, rate $\geq 1 - \alpha$, distance at least $\delta = \epsilon\alpha/2$, and locally correctable with $O(n^\epsilon)$ queries from $\delta/10$-fraction errors. Peeking ahead, we will be taking a multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$, with $m = \Theta(\frac{1}{\epsilon}), s = \Omega(\frac{m^2}{\alpha})$ (both are constants) and $d = (1-\delta) \cdot s \cdot q$ with $q \to \infty$.

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 $s = O_{\epsilon,\alpha}(m^2)$). Observe that m and s are constants. Let $d = \lfloor (1-\delta) \cdot s \cdot q \rfloor$. 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 block-length n and is over an alphabet of size $q^{\binom{m+s-1}{m}} = n^{O_{\epsilon,\alpha}(1)}$. By Lemma 3.5, $\mathcal{C}$ has distance at least δ and rate at least $(1 - \frac{m^2}{s}) \cdot (1-\delta)^m > 1 - \alpha$. By Theorem 3.6, $\mathcal{C}$ can be locally corrected from $\delta/10$-fraction errors using $O(n^{1/m}) = O(n^\epsilon)$ queries. This completes the proof of Theorem 2.3. □

Finally, we complete the proof of Theorem 2.4, by concatenating suitable multiplicity codes with good linear codes over small alphabets.

PROOF OF THEOREM 2.4. Set $\alpha_1 = \alpha/2$ and $\epsilon_1 = \epsilon/2$. As in the proof of Theorem 2.3, 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 correctable from $\delta_1/10$ fraction errors with $O(n_1^{\epsilon_1})$ queries. We will take such codes $\mathcal{C}_1$ where $q = p^t$ for integers $t > 0$.

Set $\alpha_2 = \alpha/2$. We now pick another code $\mathcal{C}_2$ of message length $n_2 = \binom{m+s-1}{m} \cdot t$ that is $\mathbb{F}_p$-linear and has rate $1 - \alpha_2$ and use it to encode the symbols of $\mathcal{C}_1$. One can choose such a $\mathcal{C}_2$ with [MacWilliams and Sloane 1977]: (1) minimum distance $\delta_2 > \alpha_2^2/4\log(1/\alpha_2)$, (2) $\mathcal{C}_2$ being encodable in time $\mathrm{poly}(n_2)$, and (3) $\mathcal{C}_2$ having $\mathrm{poly}(n_2)$-time algorithms for decoding it upto half its minimum distance.

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 (2)), and has distance δ and rate R that are at least the products of the corresponding parameters of $\mathcal{C}_1$ and $\mathcal{C}_2$. 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 $\delta > \epsilon\alpha^3/256\log(1/\alpha)$.

We now argue that the code $\mathcal{C}$ is locally correctable. To locally correct some coordinate of a codeword of $\mathcal{C}$ given access to a corrupted codeword of $\mathcal{C}$, we first run the local corrector for $\mathcal{C}_1$ to decode the coordinate of $\mathcal{C}_1$ that contains that coordinate of $\mathcal{C}$. Whenever this local 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 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 corrector for $\mathcal{C}_1$ will run correctly, and this yields the desired corrected coordinate of $\mathcal{C}$. $\square$

4. LOCAL CORRECTION OF MULTIPLICITY CODES

In this section, we prove that multiplicity codes are locally 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 $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^m$, 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:

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} P^{(\mathbf{i})}(\mathbf{a})\mathbf{b}^{\mathbf{i}} = \text{coefficient of } T^e \text{ in } Q(T). \tag{3}$$

— **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,

$$Q^{(j)}(t) = \sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=j} P^{(\mathbf{i})}(\mathbf{a} + \mathbf{b}t)\mathbf{b}^{\mathbf{i}}. \tag{4}$$

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 correction algorithm for correcting from $\Omega(\delta)$-fraction errors (recall that δ is the minimum distance of the multiplicity code being considered), 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 4.3, we show how the various steps of the algorithms can be made to run in a time-efficient manner as well.

4.1. Simplified error-correction from few errors

Simplified Local 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 $P^{(<s)}(\mathbf{a})$:** 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 $\mathbf{b} \in B$):

$$\sum_{\mathbf{i}|\mathrm{wt}(\mathbf{i})=e} \mathbf{b}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coefficient of } T^e \text{ in } Q_{\mathbf{b}}(T). \tag{5}$$

Find all $\langle u_{\mathbf{i}} \rangle_{\mathrm{wt}(\mathbf{i})=e}$ which satisfy at all these equations (over all $e < s$ and $b \in B$). 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 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 (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 $\Delta(\mathrm{Enc}_{s,d,1,q}(Q),\ell_{\mathbf{b}}) < \delta < 2$. This is because for distinct $Q(T), Q'(T)$ of degree at most d, Lemma 3.5 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) 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 (5). Finally, we show that this solution $u_{\mathbf{i}}$ is unique. With probability at least 0.9 over the choice of B, the elements of B will form an interpolating set[4] for polynomials of degree $< s$ (this holds as long as q is large enough in terms of m and s; we omit the proof since this is not relevant to our final improved local correction algorithm). In particular, there is no nonzero polynomial of degree $< s$ that vanishes on all the points of B.

Hence for each e, the vector $\langle u_i \rangle_{\mathrm{wt}(\mathbf{i})=e}$ that satisfies all the equations in the system (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.

4.2. Error-correction from $\Omega(\delta)$-fraction errors

We now come to the main local correcting algorithm and the proof of Theorem 3.6. 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 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 grid-like set. 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 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 $r(\mathbf{a})$.

[4]An interpolating set for m-variate polynomials over $\mathbb{F}_q$ of degree $< s$ is a set of $\binom{m+s-1}{m}$ points in $\mathbb{F}_q^m$ such that every nonzero polynomial of degree at most s is nonzero on some point of the set.

(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 $P^{(<s)}(\mathbf{a})$:** 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 $\mathbf{b} \in B$):

$$\sum_{\mathbf{i} | \mathrm{wt}(\mathbf{i}) = e} \mathbf{b}^{\mathbf{i}} u_{\mathbf{i}} = \text{ coeff of } T^e \text{ in } Q_{\mathbf{b}}(T). \tag{6}$$

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 3.6).

PROOF OF THEOREM 3.6. For $m = 1$ the theorem is trivial, and so we assume $m \ge 2$. Recall that we have $q \ge 10m$, $q \ge \frac{d+6s}{s}$ (so that $q \ge \frac{6}{\delta}$) and that $q \ge 12(s+1)$.

We will show that the above algorithm is a local 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 4.1. *Let m be a positive integer, and let $S \subset \mathbb{F}_q$ be any set such that $|S|^m \ge 500$. 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 $m \ge 2$, and so $(12(s+1))^m > 500$), 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 (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 3.5 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) 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 (6).

Finally, we show that this solution $u_{\mathbf{i}}$ is unique with probability at least 0.9. 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 $\mathbb{F}_q$-affine map which gives a bijection between S^m and B. Via this affine map, we get a degree preserving correspondence between polynomials evaluated on S^m and polynomials evaluated on B. Now by Lemma 3.3 (and recalling that $|S| \geq 12(s+1)$), 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 (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 (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 3.6.

4.3. Running time

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

There are two key steps which we need to elaborate on. The first step is the search for the univariate polynomial $Q_{\mathbf{b}}(T)$. 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" [Guruswami et al. 2000; Sudan 2001]. 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 [Welch and Berlekamp 1986] 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 [Hirschfeld et al. 2008]), such polynomials $E(T)$, $N(T)$ will also satisfy the equalities

$$\begin{aligned} N^{(1)}(T) &= (E \cdot Q)^{(1)}(T) = E(T)Q^{(1)}(T) + E^{(1)}(T)Q(T), \\ N^{(2)}(T) &= (E \cdot Q)^{(2)}(T) = E(T)Q^{(2)}(T) + E^{(1)}(T)Q^{(1)}(T) + E^{(2)}(T)Q(T), \\ &\cdots \\ N^{(s-1)}(T) &= (E \cdot Q)^{(s-1)}(T) = \sum_{i=0}^{s-1} E^{(i)}(T)Q^{(s-1-i)}(T) \end{aligned} \tag{7}$$

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 $\Delta(\ell, \mathrm{Enc}_{s,d,1,q}(Q)) < \delta/2$. Observe that for any x where $\ell(x) = \mathrm{Enc}_{s,d,1,q}(Q)(x)$, the system of equations (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 $\deg(N - EQ) \le (sq+d)/2$, 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 $\exp(s^m) \cdot \mathrm{poly}\log q$.

This is unsatisfactory (even though it is optimal if we treat m, s as constants). We now point out a special situation where solving these noisy linear equations can be done in (optimal) time $\mathrm{poly}(s^m, \log q)$. 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 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 $r : S^m \to \mathbb{F}_q$, the standard Reed-Muller code decoding algorithms [Kasami et al. 1968] can recover the unique degree $< s$ polynomial R that agrees with r in at least $(1 + \frac{s}{|S|})/2$-fraction of the points of S^m in time $\mathrm{poly}(|S|^m, \log q)$. If $|S| > 4s$, then this fraction is at most $3/5$, and this suffices for application to the local correcting algorithm of the previous section.

5. MULTIPLICITY CODES IN COMPARISON WITH REED-MULLER CODES

Multiplicity codes are a natural extension of Reed-Muller codes. The codewords of Reed-Muller codes correspond to evaluations of multivariate polynomials on a set of points, whereas the codewords of multiplicity codes correspond to evaluations of multivariate polynomials as well as evaluations of their derivatives on a set of points. The ability to evaluate derivatives yields improvements in the tradoff between rate and local decodability in a wide range of parameters. One of the most striking range of parameters is what we have already discussed in the very high rate regime. There are other settings where the improvement is significant and we mention some of these below.

— **Query complexity at least** $\exp(\sqrt{\log n})$

Fix the codeword length to be n, the field size to be $q = 2^{\sqrt{\log n}}$ and $m = \sqrt{\log n}$. (We can choose n so that all these quantitites are integers). Let $s = m$. Let $\delta > 0$ be a lower bound on the distance, and let the degree of the polynomials be $d = \lfloor sq(1-\delta) \rfloor$. Consider the Multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$. Then the query complexity is $2^{O(\sqrt{\log n})}$. The message length is

$$\frac{\binom{d+m}{m}}{\binom{s+m}{m}} \approx \frac{(sq(1-\delta))^m}{m! 4^m} \approx \frac{q^m}{2^{O(m)}}.$$

Thus the rate is

$$\frac{1}{2^{O(m)}} = \frac{1}{2^{O(\sqrt{\log n})}}.$$

We now compare this code to a Reed-Muller code of the same codeword length n and the same query complexity $2^{O(\sqrt{\log n})}$. For Reed-Muller codes, the query complexity equals the field size, and hence $q = 2^{O(\sqrt{\log n})}$. Thus $m = \Omega(\sqrt{\log n})$. The message length is thus,

$$\binom{d+m}{m} \approx \frac{q^m}{m!}$$

making the rate of the code

$$\frac{1}{2^{\omega(\sqrt{\log n})}}.$$

By varying m, it is not hard to show that for any query complexity that is at least $2^{O(\sqrt{\log n})}$, Multiplicity codes have an asymptotically better rate than Reed-Muller codes with the same query complexity.

— **Constant rate, $n^{o(1)}$ query complexity and correcting $o(1)$ fraction errors**
Fix the codeword length to be n, the field size to be $q = n^{1/\log\log n}$ and $m = \log\log n$. Let $s = m^2$. Let $\delta = \frac{1}{\log n}$ be a lower bound on the distance, and let the degree of the polynomials be $d = \lfloor sq(1-\delta) \rfloor$. Consider the Multiplicity code of order s evaluations of degree d polynomials in m variables over $\mathbb{F}_q$.
Then the query complexity is is $n^{O(1/\log\log n)} = n^{o(1)}$. The message length is

$$\frac{\binom{d+m}{m}}{\binom{s+m}{m}} \approx \frac{(sq(1-\delta))^m}{s^m} \approx (1-o(1))q^m.$$

Thus the rate is $1 - o(1)$. In summary, we get that for Multiplicity codes we can obtain query complexity $n^{o(1)}$ and rate arbitrarily close to 1, but at the price of the relative distance being $o(1)$. We do not know how to obtain these parameters for any other family of codes. In contrast, even for very tiny relative distance ($\approx 1/n$), If we want query complexity $n^{o(1)}$ then the rate of Reed-Muller must be $o(1)$.

6. 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.
Here is an example setting with concrete numbers. Suppose we have a storage medium which stores data in 225 bit blocks. Then, 1 billion blocks of data can be encoded at rate 5/6 (to become a 1.2 billion block codeword), such that even if an arbitrary 1 million blocks of the codeword get corrupted, then one can recover with high probability any desired block of the original data by just looking at 100,000 blocks of the corrupted codeword. Earlier, codes of this rate would have required one to decode the entire 1.2 billion block codeword to recover 1 block of original data.

— 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 corrected with $\exp\left(\sqrt{\log n \log\log n}\right)$ queries.

— 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.

REFERENCES

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

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

Sanjeev Arora and Madhu Sudan. 2003. Improved low-degree testing and its applications. *Combinatorica* 23 (2003), 365–426.

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

László Babai, Lance Fortnow, and Carsten Lund. 1991a. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity* 1, 1 (1991), 3–40.

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

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

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

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

A. Deshpande, R. Jain, T. Kavitha, S. Lokam, and J. Radhakrishnan. 2002. Better lower bounds for locally decodable codes. In *20th IEEE Computational Complexity Conference (CCC)*. 184–193.

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

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

Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. 2009. Extensions to the Method of Multiplicities, with Applications to Kakeya Sets and Mergers. In *50th IEEE Symposium on Foundations of Computer Science (FOCS)*. 181–190.

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

Oded Goldreich, Howard Karloff, Leonard Schulman, and Luca Trevisan. 2002. Lower bounds for locally decodable codes and private information retrieval. In *17th IEEE Computational Complexity Conference (CCC)*. 175–183.

Alan Guo. 2013. High rate locally correctable codes via lifting. *Electronic Colloquium on Computational Complexity (ECCC)* 20 (2013), 53.

Alan Guo, Swastik Kopparty, and Madhu Sudan. 2013. New affine-invariant codes from lifting. In *ITCS*. 529–540.

Venkatesan Guruswami and Atri Rudra. 2008. Explicit Codes Achieving List Decoding Capacity: Error-Correction With Optimal Redundancy. *IEEE Transactions on Information Theory* 54, 1 (2008), 135–150.

Venkatesan Guruswami, Amit Sahai, and Madhu Sudan. 2000. Soft-decision decoding of Chinese Remainder Codes. In *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*. Redondo Beach, California, 159–168.

Venkatesan Guruswami and Madhu Sudan. 1999. Improved decoding of Reed-Solomon and Algebraic-geometric codes. *IEEE Transactions on Information Theory* 45 (1999), 1757–1767.

Venkatesan Guruswami and Carol Wang. 2011. Optimal Rate List Decoding via Derivative Codes. In *APPROX-RANDOM*. 593–604.

Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. 2013. Local Correctability of Expander Codes. In *ICALP (1)*. 540–551.

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

Russell Impagliazzo and Avi Wigderson. 1997. $\mathrm{P} = \mathrm{BPP}$ if E requires exponential circuits: Derandomizing the XOR Lemma. *Proceedings of the 29th Annual ACM Symposium on Theory of Computing* (May 1997), 220–229.

Toshiya Itoh and Yasuhiro Suzuki. 2010. New constructions for query-efficient locally decodable codes of subexponential length. *IEICE Transactions on Information and Systems* E93-D (2010), 263–270.

Tadao Kasami, Shu Lin, and W Peterson. 1968. New generalizations of the Reed-Muller codes–I: Primitive codes. *Information Theory, IEEE Transactions on* 14, 2 (1968), 189–199.

Jonathan Katz and Luca Trevisan. 2000. On the efficiency of local decoding procedures for error-correcting codes. In *32nd ACM Symposium on Theory of Computing (STOC)*. 80–86.

Iordanis Kerenidis and Ronald de Wolf. 2004. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *J. Comput. System Sci.* 69 (2004), 395–420.

Swastik Kopparty. 2012. List-decoding Multiplicity Codes. In *Electronic Colloquium on Computational Complexity (ECCC) (TR12-044)*.

Swastik Kopparty. to appear, 2014. Some remarks on multiplicity codes. In *Proceedings of the AMS Special Session on Discrete Geometry and Algebraic Combinatorics (Contemporary Mathematics)*.

Richard Lipton. 1990. Efficient checking of computations. In *7th International Symposium on Theoretical Aspects of Computer Science (STACS)*, Vol. 415 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 207–215.

Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. 1992. Algebraic Methods for Interactive Proof Systems. *J. ACM* 39, 4 (October 1992), 859–868.

F. J. MacWilliams and N. J. A. Sloane. 1977. *The Theory of Error Correcting Codes*. North Holland, Amsterdam, New York.

Kenji Obata. 2002. Optimal lower bounds for 2-query locally decodable linear codes. In *6th International Workshop on Randomization and Computation (RANDOM)*, Vol. 2483 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 39–50.

Farzad Parvaresh and Alexander Vardy. 2005. Correcting Errors Beyond the Guruswami-Sudan Radius in Polynomial Time. In *46th IEEE Symposium on Foundations of Computer Science (FOCS)*. 285–294.

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

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

Shubhangi Saraf and Madhu Sudan. 2008. Improved lower bound on the size of Kakeya sets over finite fields. *Analysis and PDE* (2008).

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

Adi Shamir. 1992. IP = PSPACE. *J. ACM* 39, 4 (October 1992), 869–877.

Madhu Sudan. 2001. Ideal Error-Correcting Codes: Unifying Algebraic and Number-Theoretic Algorithms. In *AAECC*. 36–45.

Madhu Sudan, Luca Trevisan, and Salil Vadhan. 1999. Pseudorandom generators without the XOR lemma. In *39th ACM Symposium on Theory of Computing (STOC)*. 537–546.

Stephanie Wehner and Ronald de Wolf. 2005. Improved lower bounds for locally decodable codes and private information retrieval. In *32nd International Colloquium on Automata, Languages and Programming (ICALP)*, Vol. 3580 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 1424–1436.

Lloyd R Welch and Elwyn R Berlekamp. 1986. Error correction for algebraic block codes. (1986). US Patent 4,633,470.

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

David Woodruff and Sergey Yekhanin. 2005. A geometric approach to information theoretic private information retrieval. In *20th IEEE Computational Complexity Conference (CCC)*. 275–284.

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

Sergey Yekhanin. 2008. Towards 3-query locally decodable codes of subexponential length. *J. ACM* 55 (2008), 1–16.

Sergey Yekhanin. 2012. Locally Decodable Codes. *Foundations and trends in theoretical computer science* 6 (2012), 139–255. Issue 3.

Received April 2013; revised August 2013; accepted February 2014