Correcting Errors Beyond the Guruswami-Sudan Radius in Polynomial Time

Farzad Parvaresh and *Alexander Vardy*

University of California San Diego
9500 Gilman Drive, La Jolla, CA 92093
{*fparvaresh, avardy*}*@ucsd.edu*

Abstract

We introduce a new family of error-correcting codes that have a polynomial-time encoder and a polynomial-time list-decoder, correcting a fraction of adversarial errors up to

$$\tau_M = 1 - \sqrt[M+1]{M^M R^M}$$

where R is the rate of the code and $M \geqslant 1$ is an arbitrary integer parameter. This makes it possible to decode beyond the Guruswami-Sudan radius of $1 - \sqrt{R}$ for all rates less than $1/16$. Stated another way, for any $\varepsilon > 0$, we can list-decode in polynomial time a fraction of errors up to $1 - \varepsilon$ with a code of length n and rate $\Omega(\varepsilon/\log(1/\varepsilon))$, defined over an alphabet of size $n^M = n^{O(\log(1/\varepsilon))}$. Notably, this error-correction is achieved in the worst-case against adversarial errors: a probabilistic model for the error distribution is neither needed nor assumed. The best results so far for polynomial-time list-decoding of adversarial errors required a rate of $O(\varepsilon^2)$ to achieve the correction radius of $1-\varepsilon$.

Our codes and list-decoders are based on two key ideas. The first is the transition from bivariate polynomial interpolation, pioneered by Sudan and Guruswami-Sudan [12, 22], to multivariate interpolation decoding. The second idea is to part ways with Reed-Solomon codes, for which numerous prior attempts [2, 3, 12, 18] at breaking the $O(\varepsilon^2)$ rate barrier in the worst-case were unsuccessful. Rather than devising a better list-decoder for Reed-Solomon codes, we devise better codes. Standard Reed-Solomon encoders view a message as a polynomial $f(X)$ over a field $\mathbb{F}_q$, and produce the corresponding codeword by evaluating $f(X)$ at n distinct elements of $\mathbb{F}_q$. Herein, given $f(X)$, we first compute one or more related polynomials $g_1(X), g_2(X), \ldots, g_{M-1}(X)$ and produce the corresponding codeword by evaluating all these polynomials. Correlation between $f(X)$ and $g_i(X)$, carefully designed into our encoder, then provides the additional information we need to recover the encoded message from the output of the multivariate interpolation process.

* This work was supported in part by a Fellowship from the David and Lucile Packard Foundation and by the National Science Foundation.

1. Introduction

Let q be a power of a prime, let $\mathbb{F}_q$ denote the finite field of order q, and let $\mathbb{F}_q^n$ be the vector space of n-tuples over $\mathbb{F}_q$, endowed with the Hamming metric. An *error-correcting code* $\mathbb{C}$ of length n is simply a subset of $\mathbb{F}_q^n$, whose elements are called *codewords*. The *minimum distance* d of $\mathbb{C}$ is the minimum Hamming distance between distinct codewords, while its *rate* is defined as $R = \log_q|\mathbb{C}|/n$. Thus an *encoder* for a code $\mathbb{C} \subseteq \mathbb{F}_q^n$ of rate R is an injective function that maps $\lfloor Rn \rfloor$ arbitrary message symbols over an alphabet of size q into n coded symbols over $\mathbb{F}_q$ (a codeword of $\mathbb{C}$).

One of the central questions in coding theory can be stated as follows:

$$\text{What is the largest possible fraction of errors that a code } \mathbb{C} \text{ of rate } R \text{ can correct?} \qquad (\star)$$

To make sense of this question, we first need to make it precise. What are "errors" and what does it mean to "correct" them? Throughout this paper, we consider the case of *adversarial errors.* We say that a code $\mathbb{C}$ corrects up to t errors if the "correction" is *guaranteed* whenever we are presented with a vector $\underline{v} = \underline{c} + \underline{e} = (v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n$ that differs from the original codeword $\underline{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{C}$ in at most t positions. An alternative approach would be to assume a probability distribution on the error values (the differences $e_j = v_j - c_j$) and then require "correction" only with a certain (high) probability. For more on this, see for example [20]. Herein, a probabilistic model for the distribution of error values is *not* assumed. The "correction" itself can be also understood in a number of ways. If correction is defined to be "given $\underline{v}$, output the original message" and no limit is imposed on the time it might take to produce such output, then any code $\mathbb{C}$ can correct precisely $\lfloor (d-1)/2 \rfloor$ adversarial errors, where d is the minimum distance of $\mathbb{C}$. Thus answering the question in $(\star)$ becomes equivalent to maximizing the relative distance d/n for a given rate R. For much more on this classical problem, see e.g. [16]. However, it makes sense to modify the definition of "correction" in two important ways. First, instead of requiring the decoder for $\mathbb{C}$ to produce *only* the original message, one could define correction to be "given $\underline{v}$, output a small list of messages that is guaranteed to include the original message." This is known as the *list-decoding problem*, dating back to the work of [5, 24] in the 1950s. Second, it makes sense to require that decoding — namely, producing this list of messages — is accomplished in time that is *polynomial* in the code length n. This, in particular, guarantees that the size of the list produced by the decoder is at most polynomial in n.

For adversarial errors, with error-correction defined as above, this work gives the best known answer to the question in ($\star$), at least for low rates. We present a new class of algebraic error-correcting codes that correct a fraction of

$$\tau = 1 - O\big(R\log(1/R)\big) \tag{1}$$

adversarial errors in polynomial time. This exceeds the best previously known bound $\tau_{\mathrm{GS}} = 1 - \sqrt{R}$, due to Guruswami and Sudan [12], for all rates $R < 1/16$.

1.1. Prior work on the list-decoding problem

Formally, a code $\mathbb{C} \subseteq \mathbb{F}_q^n$ is said to be (τ, L) *list-decodable* if for all $\underline{v} \in \mathbb{F}_q^n$, the number of codewords at Hamming distance τn or less from $\underline{v}$ is at most L. Most of the applications of list-decodable codes in theoretical computer science correspond to the "high noise" situation where τ is close to 1, and we henceforth emphasize this fact by writing τ as $1 - \varepsilon$. For an excellent overview of the list-decoding problem and its applications, read Guruswami [7].

In particular, it is observed in [7] that $\big(1 - \varepsilon, O(1/\varepsilon)\big)$ list-decodable codes of rate $\Omega(\varepsilon)$ *exist*, and that $\Omega(\varepsilon)$ is the *best rate possible* for such codes. However, this existence result is based on random coding arguments — we do not know how to explicitly *construct* such codes in polynomial time, let alone *decode* them in polynomial time.

The first real breakthrough on the list-decoding problem was achieved by Sudan [22] and by Guruswami-Sudan [12] in the late 1990s. The groundbreaking work of [12] shows that using Reed-Solomon (and other algebraic) codes, it is possible to list-decode in polynomial time a fraction of $1-\varepsilon$ errors with a code of rate ε^2. Since then, the rate bound of $O(\varepsilon^2)$ has become a seemingly unsurmountable "barrier" for list-correcting a fraction of $1 - \varepsilon$ adversarial errors in polynomial time. To quote from [8]:

> *Despite much progress in the general area of list decoding, the $O(\varepsilon^2)$ bound has been a "barrier" for the rate for codes list-decodable up to a fraction $(1 - \varepsilon)$ of errors.* [...] *The question of improving the rate beyond the $O(\varepsilon^2)$ barrier is one of the central open questions in the subject of list-decoding.*

Here is a brief, and necessarily incomplete, overview of prior attempts at breaking this barrier. In [10], the authors present a randomized construction of codes of rate $\Omega(\varepsilon)$ together with an algorithm to list-decode such codes up to a fraction of $1 - \varepsilon$ errors. However, the construction of [10] is not explicit, since there is no efficient way to certify that the resulting codes have the desired list-decoding properties. Moreover, the list-decoding algorithm of [10] runs in *subexponential* time of the form $2^{O(n^{1/\tau})}$. The first explicit construction of codes with rate better than $\Omega(\varepsilon^2)$ was given by Guruswami in [8]. This construction uses *extractors* in an ingenious way to come up with $(1-\varepsilon, L)$ list-decodable codes. The rate that these code achieve is $\varepsilon/\log^{O(1)}(1/\varepsilon)$, which is not far from our results. However, the list-decoding algorithm of [8] is *not* polynomial-time: it runs in subexponential time of the form $2^{c(Rn)^{1/\tau}}$ and could return subexponential-sized lists as output. There are other graph-based constructions, most notably by Guruswami and Indyk [11], that for rates $\Omega(\varepsilon^2)$ correct a fraction of $O(1-\varepsilon)$ errors. The Guruswami-Indyk codes [11] are encodable and list-decodable in almost linear time; however, they do not break the $O(\varepsilon^2)$ rate-barrier. A completely different approach was recently developed by Muralidhara and Sen [17] to show that Reed-Solomon codes are $(1 - \varepsilon, L)$ list-decodable at a rate of $\Omega\big((\varepsilon + \frac{c}{n})^2\big)$ for any constant $c > 0$. Unfortunately, the list size and the complexity of decoding in [17] grow at least as $\Omega(n^c)$, which again does not allow to break the $O(\varepsilon^2)$ rate-barrier in polynomial time.

Finally, we should mention the work of [2], [3], and [18]. Bleichenbacher, Kiayias, and Yung [2] study Reed-Solomon codes of a certain type over an alphabet of size $Q = n^M$. They devise a unique (list of size one) decoding algorithm that, with *probability* at least $1 - O(1/n)$ on the *Q-ary symmetric channel*, corrects a fraction of $1 - \varepsilon$ errors using a code of rate $R = \big(\varepsilon(M+1) - 1\big)/M$. Coppersmith and Sudan [3] consider the same Reed-Solomon codes[1] over $\mathbb{F}_Q$ and devise a list-decoding algorithm that corrects a fraction of $1 - \varepsilon$ errors using a code of rate $R = \Omega(\varepsilon^\alpha)$, for any constant $\alpha > 1$. However, this correction is again achieved only with a certain *probability*, which Coppersmith and Sudan [3] show to be at least $\Omega\big(R^{M/(M+1)}\big)$, again assuming the same Q-ary symmetric channel model. Thus neither [2] nor [3] deal with adversarial errors. In our earlier work [18], we do not assume a probabilistic model for the error distribution. Nevertheless, we cannot provide a *guarantee* on the list-decoding radius of our algorithm in [18] that would break the $O(\varepsilon^2)$ rate-barrier either.

In summary, there is no construction known (to us) of error-correcting codes that are $(1 - \varepsilon, L)$ list-decodable in polynomial time against adversarial errors on one hand, and exceed the $O(\varepsilon^2)$ rate-barrier on the other hand.

[1] Neither Bleichenbacher, Kiayias, and Yung [2] nor Coppersmith and Sudan [3] recognize that the codes they consider are, in fact, Reed-Solomon codes, but this is shown in [19].

1.2. Our main results

Herein, we present such a construction. We will explicitly describe a new family of algebraic error-correcting codes and devise a novel list-decoding algorithm for these codes, which makes it possible to prove the following theorem.

Theorem 1. *Let q be a power of a prime. Then for all positive integers m, M, $n \leqslant q$, and $k \leqslant n$, there is a code $\mathbb{C}$ of length n and rate $R = k/nM$ over $\mathbb{F}_Q = \mathrm{GF}(q^M)$, equipped with an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ that have the following properties. Given any vector $\underline{v} \in \mathbb{F}_Q^n$, the decoder $\mathcal{D}$ outputs the list of all codewords that differ from $\underline{v}$ in at most*

$$t = \left\lfloor n - n \sqrt[M+1]{M^M R^M \left(1+\frac{1}{m}\right) \cdots \left(1+\frac{M}{m}\right)} - \frac{1}{m} \right\rfloor$$

positions (where m, M are arbitrary integer parameters). The size of this list is at most

$$L = \left\lceil m \sqrt[M+1]{\frac{(1+\frac{1}{m})(1+\frac{2}{m})\cdots(1+\frac{M}{m})}{MR}} \right\rceil^M + o(1)$$

Moreover, both the encoder $\mathcal{E}$ and the decoder $\mathcal{D}$ run in time (number of operations in $\mathbb{F}_q$) that is bounded by a polynomial in n and m, for every given $M \geqslant 1$.

To arrive at our main result, it remains to set the parameters in Theorem 1 appropriately. Given a small $\varepsilon > 0$, we write $t/n = 1 - \varepsilon$. Then, assuming m is sufficiently large, the best choice for M is given by $M = \ln(1/\varepsilon)$. Note that

$$F_M(m) \overset{\text{def}}{=} \prod_{i=1}^{M} \left(1 + \frac{i}{m}\right) = 1 + \frac{M^2 + M}{2m} + O\left(\frac{1}{m^2}\right)$$

Thus we can make this function arbitrarily close to 1, regardless of the value of ε, provided $m = \omega(M^2)$. For example, we could take $m = M^2 \log\log M$. Then the rate R and the list-size L in Theorem 1 are given by

$$R \simeq \frac{\varepsilon^{\frac{M+1}{M}}}{M} = \frac{e^{-1}\varepsilon}{\ln\frac{1}{\varepsilon}}; \qquad L \simeq \frac{m^M}{\varepsilon} = \left(\frac{1}{\varepsilon}\right)^{O\left(\log\log\frac{1}{\varepsilon}\right)}$$

Finally, taking $n = q$, the code $\mathbb{C}$ in Theorem 1 is defined over an alphabet of size $Q = q^M = n^{\lfloor \ln(1/\varepsilon) \rfloor}$.

Corollary 2. *For all sufficiently small $\varepsilon > 0$, we can explicitly construct a family of*

$$\left(1 - \varepsilon, (1/\varepsilon)^{O\left(\log\log\frac{1}{\varepsilon}\right)}\right)$$

list-decodable codes of rate $R = \Omega\left(\varepsilon/\log(1/\varepsilon)\right)$ defined over an alphabet of size $n^{O(\log(1/\varepsilon))}$. Moreover, such codes can be encoded and list-decoded in polynomial time.

The codes we construct to establish Theorem 1 are certain highly-structured subsets of certain highly-structured Reed-Solomon codes. As we shall see, the underlying ideas are extremely simple. Indeed, proving Theorem 1, at least for the important special case $M = 2$, will not require much technical sweat. It is also curious to note that, in general, our codes are *not linear* — neither over $\mathbb{F}_Q$ nor over $\mathbb{F}_q$. This unusual circumstance does not seem to impede in any way polynomial-time encoding and list-decoding of these codes.

An important shortcoming of our result in Corollary 2 is that the alphabet size grows very fast: $n^{O(\log(1/\varepsilon))}$ is *much* larger than $O(1/\varepsilon)$, which is the best one could hope for [7]. However, in a recent preprint, Guruswami [9] extends our methods by relpacing the underlying Reed-Solomon codes with more general algebraic-geometric codes. Using this (highly nontrivial!) generalization, he then shows in [9] that it is possible to reduce the alphabet size to $(1/\varepsilon)^{O(\log(1/\varepsilon))}$ — which does not depend on the code length n — at the expense of slightly decreasing the rate to $\Omega\left(\varepsilon/\log^2(1/\varepsilon)\right)$.

1.3. The underlying ideas

Our results in this paper are based upon two key ideas. The first is the transition from bivariate polynomial interpolation, pioneered by Sudan [22] and Guruswami-Sudan [12], to *multivariate interpolation decoding*. It was recognized early on that decoding Reed-Solomon codes is equivalent to the problem of reconstructing univariate polynomials from their noisy evaluations. Conventional decoding [1, 23] thus attempts to solve this problem using *univariate* polynomial interpolation. The breakthrough achieved by Sudan [22] and Guruswami-Sudan [12] was due in large part to the transition from univariate to *bivariate* polynomial interpolation. Herein, we move on to interpolation decoding in $M+1$ variables, where $M \geqslant 1$ is the integer parameter in Theorem 1. Much of this framework was developed in our earlier papers [18, 19]. However, since that work is not yet readily accessible, we will give a brief primer on multivariate interpolation decoding in the next section.

The second key idea is to part ways with Reed-Solomon codes, for which numerous prior attempts [2,3,12,17,18] at breaking the $O(\varepsilon^2)$ rate barrier in the worst-case proved unsuccessful. Rather than trying to devise a better list-decoder for Reed-Solomon codes (which, as it turns out [19], is what all the previous papers attempted to do), we construct better codes. Our construction is still based upon the Reed-Solomon codes, yet differs from them in an essential way. Standard Reed-Solomon encoders view a message as a polynomial $f(X)$ over a field $\mathbb{F}_q$, and produce the corresponding codeword by evaluating $f(X)$ at n distinct elements of $\mathbb{F}_q$. In this paper, given $f(X)$, we first compute $M - 1$ related polynomials $g_1(X), g_2(X), \ldots, g_{M-1}(X)$ and produce the corresponding codeword by evaluating *all* these polynomials. In the degenerate case $M = 1$, we thus rederive the Guruswami-Sudan list-decoding algorithm [12] for RS codes. However, when $M \geqslant 2$, we can do better: correlation between $f(X)$ and $g_i(X)$, carefully designed into our encoder, provides precisely the kind of information we need to break the $O(\varepsilon^2)$ rate-barrier for adversarial errors.

1.4. Overview of the paper

The next section is a primer on multivariate interpolation. In Sections 3 and 4, we consider in detail the first nontrivial case $M = 2$. As we shall see, for $M = 2$, there are several different ways to obtain the necessary correlation between $f(X)$ and $g(X)$. In Section 3, we describe a particularly elegant and simple way to do so, namely

$$g(X) = \big(f(X)\big)^a \mod e(X) \tag{2}$$

where $e(X)$ is an arbitrary irreducible (over $\mathbb{F}_q$) polynomial of degree k, and a is an arbitrary (but sufficiently large) integer. The encoding rule (2) leads to a simple list-decoding algorithm. Using this algorithm, we give in Section 3 a proof of Theorem 1 for the special case $M = 2$. In Section 4, we show that the encoding rule (2) is just an instance of a much more general situation where the pair $\big(f(X), g(X)\big)$ can be regarded as a zero of an irreducible polynomial $\mathcal{T}(Y, Z)$ over the field $\mathbb{K} = \mathbb{F}_{q^k}$. We furthermore extend the list-decoding algorithm of Section 3 to this more general scenario. Finally, in Section 5, we generalize the construction and the decoding algorithm of Section 3 for all $M \geqslant 2$, and thereby complete the proof of Theorem 1.

2. Multivariate interpolation

We will describe in detail only the (first nontrivial) case of trivariate interpolation. Generalization to multivariate interpolation in an arbitrary number of variables should be fairly straightforward, and will be briefly sketched-out later on.

We need some definitions pertaining to trivariate polynomials. Thus let $Q(X,Y,Z) = \sum_{i,j,k=0}^{\infty} q_{i,j,k} X^i Y^j Z^k$ be a polynomial over $\mathbb{F}_q$, and let $\mathbb{N}$ denote the natural numbers. For $a, b, c \in \mathbb{N}$, the corresponding *Hasse derivative* of Q is

$$\mathscr{D}_{a,b,c}\big[Q(X,Y,Z)\big] \overset{\text{def}}{=} \sum_{i=a}^{\infty}\sum_{j=b}^{\infty}\sum_{k=c}^{\infty} \binom{i}{a}\binom{j}{b}\binom{k}{c} q_{i,j,k} X^{i-a} Y^{j-b} Z^{k-c} \tag{3}$$

The polynomial $Q(X,Y,Z)$ is said to have a *zero of multiplicity* m at a point $(x_0, y_0, z_0) \in \mathbb{F}_q \times \mathbb{F}_q \times \mathbb{F}_q$ if

$$\mathscr{D}_{a,b,c}\big[Q(X,Y,Z)\big]\Big|_{(x_0,y_0,z_0)} = 0 \qquad \text{for all } a,b,c \in \mathbb{N} \text{ with } a+b+c < m \tag{4}$$

We next define the weighted degree of trivariate polynomials. While this is a fairly general concept, we will only need the $(1, k-1, k-1)$-weighted degree, where $k-1$ is the degree of the message polynomial $f(X)$. Thus, we define the *weighted degree* of a monomial $X^a Y^b Z^c$ as follows

$$\text{wdeg}\, X^a Y^b Z^c \overset{\text{def}}{=} a + (k-1)b + (k-1)c \tag{5}$$

We extend the weighted degree to a monomial ordering $\prec_{\text{w}}$ by augmenting it with the lex order (cf. [4, Chapter 2]). The weighted degree of a polynomial can be now defined as the weighted degree of its leading (under $\prec_{\text{w}}$) monomial.

Now let $x_1, x_2, \ldots, x_n$ be some n distinct elements of $\mathbb{F}_q$, and consider the *interpolation set* $\mathcal{P}$ given by

$$\mathcal{P} \overset{\text{def}}{=} \Big\{(x_1,y_1,z_1),(x_2,y_2,z_2),\ldots,(x_n,y_n,z_n)\Big\} \tag{6}$$

where $y_j, z_j \in \mathbb{F}_q$ are unrestricted. We define the *interpolation polynomial* with respect to $\mathcal{P}$ as the least weighted degree nonzero polynomial $\mathcal{Q}_m(X,Y,Z)$ in $\mathbb{F}_q[X,Y,Z]$ that has a zero of multiplicity m at each of the n points of $\mathcal{P}$.

It should be obvious that, given $\mathcal{P}$, the interpolation polynomial $\mathcal{Q}_m(X,Y,Z)$ can be computed in time that is bounded by a polynomial in n and m, say with straightforward Gaussian elimination and/or matrix inversion. In fact, using a generalization of the algorithm due to Koetter [14], it is shown in [19] that this computation takes $O\big(n^{8/3}m^8/k^{2/3}\big)$ operations in $\mathbb{F}_q$. Notably, the complexity analysis of [19] easily carries over to general multivariate interpolation.

Lemma 3.

$$\text{wdeg}\, \mathcal{Q}_m(X,Y,Z) \leqslant \left\lceil \sqrt[3]{n(k-1)^2 m(m+1)(m+2)} \right\rceil$$

Proof. The proof is a straightforward generalization of the argument used in [12] and [15]. Let $N_3(\Delta)$ denote the number of trivariate monomials whose weighted degree is at most Δ. We claim that there exists a nonzero polynomial of weighted degree at most Δ that passes through each point in $\mathcal{P}$ with multiplicity m, provided

$$N_3(\Delta) > n\,\frac{m(m+1)(m+2)}{6} \tag{7}$$

Indeed, referring to (3), it is clear that (4) is just a system of linear constraints on the coefficients of such a polynomial. The total number of linear constraints imposed by passing with multiplicity m through each of n points is given by the right-hand side of (7). The total number of unknowns (coefficients of the polynomial) is $N_3(\Delta)$. If the number of unknowns is strictly greater than the number of constraints, the linear system is guaranteed to have a nonzero solution. It follows that any Δ such that $N_3(\Delta)$ satisfies (7) is an upper bound on the weighted degree of the interpolation polynomial. It remains to estimate $N_3(\Delta)$. To this end, consider the correspondence between monomials in $\mathbb{F}_q[X,Y,Z]$ and unit cubes in $\mathbb{R}^3$, defined by $X^aY^bZ^c \mapsto K(a,b,c)$, where $K(a,b,c) = [a, a{+}1) \times [b, b{+}1) \times [c, c{+}1)$. Then $N_3(\Delta)$ is equal to the *volume* of a union of such cubes $K(a,b,c)$, the union being taken over all $a,b,c \in \mathbb{N}$ that satisfy

$$a + (k-1)b + (k-1)c \leqslant \Delta$$

Let $\mathscr{U} \subset \mathbb{R}^3$ denote this union of cubes, and consider the pyramid $\mathscr{P} \subset \mathbb{R}^3$ defined by the four half-planes: $x \geqslant 0$, $y \geqslant 0$, $z \geqslant 0$, and $x + (k-1)y + (k-1)z \leqslant \Delta$. It is easy to see that $\mathscr{P} \subset \mathscr{U}$, so $N_3(\Delta) = \text{Vol}(\mathscr{U}) > \text{Vol}(\mathscr{P})$. But $\text{Vol}(\mathscr{P}) = \Delta^3/6(k-1)^2$ and the lemma follows. ∎

Lemma 4. *Let $\mathcal{Q}_m(X,Y,Z)$ be the interpolation polynomial with respect to the set $\mathcal{P}$ in (6). Given a pair of polynomials $f(X)$ and $g(X)$ over $\mathbb{F}_q$, define*

$$\Big|(f,g)\Big|_{\mathcal{P}} \overset{\text{def}}{=} \Big|\{\, j \,:\, f(x_j) = y_j \text{ and } g(x_j) = z_j \}\Big| \tag{8}$$

Provided the degree of both $f(X)$ and $g(X)$ is at most $k-1$, and moreover

$$\Big|(f,g)\Big|_{\mathcal{P}} \geqslant \left\lceil \sqrt[3]{n\,(k-1)^2\Big(1+\frac{1}{m}\Big)\Big(1+\frac{2}{m}\Big)} \;+\; \frac{1}{m} \right\rceil$$

we have $\mathcal{Q}_m\big(X,f(X),g(X)\big) \equiv 0$. Namely, the univariate polynomial $p(X) = \mathcal{Q}_m\big(X,f(X),g(X)\big)$ is identically zero.

Proof. If $\deg f(X),\ \deg g(X) \leqslant k-1$, it follows from the definition of the weighted degree in (5) that

$$\deg p(X) \;\leqslant\; {}_{\mathbf{w}}\!\deg \mathcal{Q}_m(X,Y,Z) \tag{9}$$

On the other hand, for every integer $j \in \{1,2,\ldots,n\}$ that is counted in (8), the interpolation polynomial $\mathcal{Q}_m(X,Y,Z)$ has a zero of multiplicity m at the point $\big(x_j, f(x_j), g(x_j)\big)$. Hence the total number of zeros of $p(X)$ in $\mathbb{F}_q$, counted with multiplicity, is lower-bounded by

$$\#\text{ zeros of } p(X) \;\geqslant\; m\,\Big|(f,g)\Big|_{\mathcal{P}} \tag{10}$$

If the right-hand side of (10) is strictly greater than the right-hand side of (9), then $p(X)$ must be the all-zero polynomial. The lemma now follows from Lemma 3. ∎

Before concluding this section, we need to establish one more technical lemma.

Lemma 5. *A polynomial $Q(X,Y,Z) \in \mathbb{F}_q[X,Y,Z]$ satisfies $Q\big(X,f(X),g(X)\big) \equiv 0$ if and only if it belongs to the ideal generated by $Y - f(X)$ and $Z - g(X)$, that is, if and only if there exist some polynomials $\mathcal{A},\mathcal{B} \in \mathbb{F}_q[X,Y,Z]$ such that*

$$Q(X,Y,Z) \;=\; \mathcal{A}\big(Y - f(X)\big) + \mathcal{B}\big(Z - g(X)\big) \tag{11}$$

Proof. It should be obvious that if $Q(X,Y,Z)$ can be expressed as in (11), then $Q\big(X,f(X),g(X)\big) \equiv 0$. To prove the converse, fix a monomial order $\prec$ such that $Y \succ f(X)$ and $Z \succ g(X)$, then divide $Q(X,Y,Z)$ by $Y - f(X)$ and $Z - g(X)$ to express it as follows:

$$Q(X,Y,Z) \;=\; \mathcal{A}\big(Y - f(X)\big) + \mathcal{B}\big(Z - g(X)\big) + R \tag{12}$$

The polynomial division algorithm guarantees that none of the monomials in the remainder polynomial $R \in \mathbb{F}_q[X,Y,Z]$ is divisible by the leading term of $Y - f(X)$ or $Z - g(X)$. But since $Y \succ f(X)$ and $Z \succ g(X)$, this simply means that $R(X,Y,Z) = R(X)$. This, in conjunction with (12) and the assumption that $Q\big(X,f(X),g(X)\big) \equiv 0$, then implies that $R(X) \equiv 0$, and the lemma follows. ∎

3. Simple trivariate codes and their decoding

We are now ready to describe the construction of our codes for the simplest nontrivial case: $M = 2$ (trivariate interpolation) with the encoding rule of (2). This is done in the next subsection. Then, in Section 3.2, we describe a decoding algorithm for these codes that achieves the error-correction claimed in Theorem 1 for the special case of $M = 2$.

3.1. Code parameters and encoder mapping

A specific code $\mathbb{C}$ in the family constructed in this section is specified by the following set of parameters.

Underlying field order: An integer q, which is a power of a prime; determines the message symbol alphabet.

Length: A positive integer $n \leqslant q$, which determines the length of the code (a good choice is $n = q$).

Evaluation set: A set of n elements $x_1, x_2, \ldots, x_n$ of $\mathbb{F}_q$, that must be distinct but are otherwise arbitrary.

Rate parameter: A positive integer k which, along with n, determines the rate of the code.

Basis: A fixed basis $\{1,\alpha\}$ for $\mathbb{F}_{q^2}$ over $\mathbb{F}_q$.

Multiplicity parameter: An arbitrary positive integer m, which determines decoding interpolation multiplicity.

Exponentiation modulus: A polynomial $e(X) \in \mathbb{F}_q[X]$ of degree k, that is irreducible over $\mathbb{F}_q$ but otherwise arbitrary; determines the encoder mapping.

Exponentiation degree: A positive integer a that satisfies

$$a \;\geqslant\; \left\lceil m\,\sqrt[3]{\frac{n}{k-1}\Big(1+\frac{1}{m}\Big)\Big(1+\frac{2}{m}\Big)} + \frac{1}{k-1} \right\rceil \tag{13}$$

but is otherwise arbitrary; along with $e(X)$, determines the encoder mapping, also determines the list-size.

We assume that all these parameters are fixed in advance, and known to both the encoder and the decoder for $\mathbb{C}$.

Our code $\mathbb{C}$ is a subset of the set of vectors of length n over $\mathbb{F}_{q^2}$. We define $\mathbb{C}$ by describing a bijective encoder map $\mathcal{E}: \mathbb{F}_q^k \to \mathbb{C}$, as follows. Given k arbitrary message symbols $u_0, u_1, \ldots, u_{k-1} \in \mathbb{F}_q$, the encoder first constructs the polynomial $f(X) = \sum_{i=0}^{k-1} u_i X^i$, and then computes

$$g(X) \;=\; \big(f(X)\big)^a \mod e(X) \tag{14}$$

over $\mathbb{F}_q$. The codeword $(c_1, c_2, \ldots, c_n) \in \mathbb{C}$ corresponding to the message $u_0, u_1, \ldots, u_{k-1}$ is then given by

$$c_j \;=\; f(x_j) + \alpha g(x_j) \qquad \text{for } j = 1,2,\ldots,n \tag{15}$$

It is obvious from the above description that the rate of $\mathbb{C}$ is $R = \log_{q^2}|\mathbb{C}|/n = k/2n$. The minimum distance of $\mathbb{C}$ is at least $n-k+1$, since $\mathbb{C}$ is a subset of a Reed-Solomon code $\mathscr{C}$ of length n and dimension k over $\mathbb{F}_{q^2}$. We leave the proof of this fact as an exercise for the reader.

It is interesting to observe that in most cases $\mathbb{C}$ is not a *linear subspace* of $\mathscr{C}$ because the computation in (14) is, in general, not a linear operation. However, if $q = p^s$ for a prime p and we take the exponentiation degree a to be of the form $a = p^b$, where b is a multiple of s, then the code $\mathbb{C}$ becomes $\mathbb{F}_q$-linear. It is still not a linear code in the usual sense of the word, since it is not linear over the field $\mathbb{F}_{q^2}$ over which it is defined. Making it linear over $\mathbb{F}_{q^2}$ requires $a-1$ to be a multiple of q^k-1, and leads to the degenerate case where $g(X) \equiv f(X)$ since the modular exponentiation in (14) then becomes the identity operation.

Finally, we need to show that the encoder map in (14) and (15) can be computed in polynomial time. To this end, it is convenient to think of the polynomials $f(X)$ and $g(X)$ of degree $\leqslant k-1$ as elements in the extension field

$$\mathbb{K} \overset{\text{def}}{=} \mathbb{F}_q[X] \,/\, \langle e(X) \rangle \tag{16}$$

of order q^k. Let β and γ denote the elements of $\mathbb{K}$ that correspond to $f(X)$ and $g(X)$, respectively. Then the computation in (14) becomes simply $\gamma = \beta^a$ in $\mathbb{K}$. We can now use a result of Von zur Gathen and Nöcker [6], who show how to compute β^a with $O(k^2 \log\log k)$ operations in the *ground field* $\mathbb{F}_q$, using storage for only $O(k/\log^2 k)$ elements of $\mathbb{K}$.

3.2. Simple trivariate interpolation decoding

We have seen in the foregoing subsestion that the construction of our codes is conceptually quite simple. Their decoding is equally simple. Given a vector $\underline{v} = (v_1, v_2, \ldots, v_n)$ over $\mathbb{F}_{q^2}$, we first decompose each v_j into its two components over $\mathbb{F}_q$, using the fixed basis $\{1, \alpha\}$. Thus we write

$$v_j = y_j + \alpha z_j \qquad \text{for } j = 1, 2, \ldots, n$$

where y_j and z_j are in $\mathbb{F}_q$. We then set up the interpolation set $\mathcal{P}$ exactly as in (6), namely

$$\mathcal{P} = \Big\{ (x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n) \Big\}$$

where $x_1, x_2, \ldots, x_n$ is the evaluation set for $\mathbb{C}$. As observed in Section 2, the corresponding interpolation polynomial $\mathcal{Q}_m(X, Y, Z)$ can be computed in time $O\big(n^{8/3} m^8 / k^{2/3}\big)$.

Lemma 6. *Suppose that a codeword $\underline{c} \in \mathbb{C}$ differs from the given vector $\underline{v} = (v_1, v_2, \ldots, v_n)$ in at most*

$$t = \left\lfloor n - \sqrt[3]{n\,(k-1)^2 \left(1+\frac{1}{m}\right)\left(1+\frac{2}{m}\right)} - \frac{1}{m} \right\rfloor \tag{17}$$

positions, and let $f(X)$, $g(X)$ denote the message polynomial(s) that produce $\underline{c}$ according to the mapping (14)–(15). *Then $\mathcal{Q}_m(X, Y, Z)$ satisfies $\mathcal{Q}_m\big(X, f(X), g(X)\big) \equiv 0$.*

Proof. This follows immediately from Lemma 4, upon observing that $\big|(f, g)\big|_{\mathcal{P}} \geqslant n - t$. ∎

It remains to show how to recover the message $f(X)$ from the interpolation polynomial $\mathcal{Q}_m(X, Y, Z)$, assuming that $\mathcal{Q}_m\big(X, f(X), g(X)\big) \equiv 0$. To this end, we first reduce $\mathcal{Q}_m(X, Y, Z)$ modulo $e(X)$. That is, we compute

$$P(Y, Z) \overset{\text{def}}{=} \mathcal{Q}_m(X, Y, Z) \mod e(X) \tag{18}$$

To see that $P(Y, Z)$ is indeed a bivariate polynomial, first write $\mathcal{Q}_m(X, Y, Z)$ as an element of $\mathbb{F}_q[X][Y, Z]$, namely

$$\mathcal{Q}_m(X, Y, Z) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} q_{i,j}(X)\, Y^i Z^j \tag{19}$$

and let $p_{i,j}(X) = q_{i,j}(X) \mod e(X)$. Then clearly $p_{i,j}(X)$ can be regarded as an element of the extension field $\mathbb{K}$ defined in (16), and therefore $P(Y, Z) \in \mathbb{K}[Y, Z]$. We point out that this is mainly a syntactic trick, but it greatly simplifies the derivation in what follows.

Lemma 7. *The polynomial $P(Y, Z)$ defined by* (18) *is not the all-zero polynomial.*

Proof. Assume to the contrary that $P(Y, Z) \equiv 0$, or equivalently that $e(X)$ is a factor of $q_{i,j}(X)$ for all i and j, where $q_{i,j}(X)$ are the coefficients in (19). But this means that $e(X)$ is a factor of $\mathcal{Q}_m(X, Y, Z)$, so we can define the polynomial

$$\mathcal{Q}'(X, Y, Z) \overset{\text{def}}{=} \frac{\mathcal{Q}_m(X, Y, Z)}{e(X)} \tag{20}$$

Using the fact that $e(X)$ is irreducible, it is not difficult to show that if $\mathcal{Q}_m(X, Y, Z)$ satisfies all the interpolation constraints, then so does $\mathcal{Q}'(X, Y, Z)$. Furthermore, we have $\mathbf{w}\deg \mathcal{Q}'(X, Y, Z) = \mathbf{w}\deg \mathcal{Q}_m(X, Y, Z) - \deg e(X)$ in view of (20). But this contradicts the definition of the interpolation polynomial $\mathcal{Q}_m(X, Y, Z)$ as the *least weighted-degree* polynomial satisfying the interpolation constraints. ∎

Lemma 8. *Suppose that $\mathcal{Q}_m\big(X, f(X), g(X)\big) \equiv 0$, and let β and γ denote the elements of $\mathbb{K}$ that correspond to $f(X)$ and to $g(X)$, respectively. Then $P(\beta, \gamma) = 0$.*

Proof. If $\mathcal{Q}_m\big(X, f(X), g(X)\big) \equiv 0$, then by Lemma 5 it can be written in the form

$$\mathcal{A}(X, Y, Z)\big(Y - f(X)\big) + \mathcal{B}(X, Y, Z)\big(Z - g(X)\big)$$

for some polynomials $\mathcal{A}(X,Y,Z)$, $\mathcal{B}(X,Y,Z)$ in $\mathbb{F}_q[X,Y,Z]$. Therefore $P(Y, Z)$ can be written in the form

$$P(Y, Z) = \widetilde{\mathcal{A}}(Y, Z)\big(Y - \beta\big) + \widetilde{\mathcal{B}}(Y, Z)(Z - \gamma) \tag{21}$$

where $\widetilde{\mathcal{A}}(Y, Z)$ and $\widetilde{\mathcal{B}}(Y, Z)$ in $\mathbb{K}[Y, Z]$ are the remainders of $\mathcal{A}(X,Y,Z)$ and $\mathcal{B}(X,Y,Z)$ upon division by $e(X)$. It is now obvious from (21) that $P(\beta, \gamma) = 0$. ∎

Here is the crucial part of our decoding algorithm. From the encoding rule (14), we know *a priori* that $\gamma = \beta^a$ in $\mathbb{K}$. Hence $P(\beta, \gamma) = 0$ implies that β is a root of the polynomial $H(Y) = P(Y, Y^a)$. But $H(Y)$ is a univariate polynomial, so all of its roots can be found using well-known techniques. First, however, we should make sure that $H(Y) \not\equiv 0$.

Lemma 9. *The polynomial $H(Y) = P(Y, Y^a)$ is not the all-zero polynomial.*

Proof. Assume to the contrary that $H(Y) = P(Y, Y^a) \equiv 0$. This happens if and only if $Z - Y^a$ is a factor of $P(Y, Z)$. Clearly, this cannot be the case if $\deg_Y P(Y, Z) < a$. But it follows from (18) and (5) that

$$\deg_Y P(Y, Z) \leqslant \frac{{}_{\mathbf{W}}\deg \mathcal{Q}_m(X, Y, Z)}{k-1} \tag{22}$$

Lemma 3 provides an upper bound on ${}_{\mathbf{W}}\deg \mathcal{Q}_m(X, Y, Z)$, and our condition for a in (13) uses this bound to guarantee that the value of a is strictly greater than the RHS of (22). ∎

Based upon Lemmas 6, 7, 8, and 9, we can now summarize the entire decoding algorithm as follows:

$\boxed{1}$ Given a vector $(v_1, v_2, \dots, v_n)$ over $\mathbb{F}_{q^2}$, set up the interpolation set $\mathcal{P}$ and compute the interpolation polynomial $\mathcal{Q}_m(X, Y, Z)$ in $\mathbb{F}_q[X, Y, Z]$.

$\boxed{2}$ Compute $P(Y, Z) = \mathcal{Q}_m(X, Y, Z) \mod e(X)$, interpreted as an element of $\mathbb{K}[Y, Z]$.

$\boxed{3}$ Compute the univariate polynomial $H(Y) = P(Y, Y^a)$.

$\boxed{4}$ Find all the roots of $H(Y)$ in $\mathbb{K}$, and output this list.

It should be obvious that this algorithm runs in polynomial time. The only potential quibble is at Step 4, where we are required to find roots of a polynomial over $\mathbb{K}$, which is an exponentially large field. Note, however, that while the order of $\mathbb{K}$ is q^k its characteristic is still equal to that of $\mathbb{F}_q$. Shoup [21] gives a deterministic algorithm for factoring polynomials over large finite fields of small characteristic. Using this algorithm, factoring a polynomial of degree D over a field of order p^K (for a prime p) takes only

$$\widetilde{O}(D^2K^2)\log p \;+\; \widetilde{O}(D^{3/2}K^{3/2})\log p\sqrt{p}$$

operations in $\mathbb{F}_p$. Thus there are no problems with Step 4 as well. The following theorem uses all of the above to establish Theorem 1 for the special case of $M = 2$.

Theorem 10. *Given a vector $\underline{v} = (v_1, v_2, \dots, v_n)$ over $\mathbb{F}_{q^2}$, the algorithm above outputs in polynomial time the list of all codewords of $\mathbb{C}$ that differ from $\underline{v}$ in at most*

$$t \;=\; \left\lfloor n - n\sqrt[3]{4R^2\left(1+\frac{1}{m}\right)\left(1+\frac{2}{m}\right)} - \frac{1}{m}\right\rfloor \tag{23}$$

positions, where $m \geqslant 1$ is an arbitrary multiplicity parameter. The size of this list is at most L^2, where

$$L \;=\; \left\lceil m\sqrt[3]{\frac{(1+\frac{1}{m})(1+\frac{2}{m})}{2R}}\,\right\rceil + 1 \tag{24}$$

Proof. The expression for t in (23) follows immediately from (17), when taking into account that $R = k/2n$. The list-size is obviously bounded by the degree of $H(Y)$. It is clear that $\deg H(Y) \leqslant a \deg_{\mathrm{tot}} P(Y, Z)$, where $\deg_{\mathrm{tot}}$ is the total degree. But $\deg_{\mathrm{tot}} P(Y, Z)$ is bounded by the RHS of (22), whence (24) now follows by Lemma 3 and (13).

In general, we will always choose a to be just larger than the RHS of (22). Hence, the list size is essentially bounded by the square of $a \simeq {}_{\mathbf{W}}\deg \mathcal{Q}_m(X, Y, Z)/(k-1)$. ∎

4. General trivariate codes and their decoding

We now show that the codes constructed in the foregoing section are, in fact, just one member of a much more general family of codes. All the codes in this family can be list-decoded in essentially the same way as the codes of Section 3.

4.1. Code parameters and encoder mapping

To the list of code parameters detailed in Section 3.1, we now add one more ingredient:

Encoder polynomial: A polynomial $\mathcal{T}(Y, Z) \in \mathbb{K}[Y, Z]$ of total degree a, that is irreducible over $\mathbb{K}$ and has positive degree in both Y and Z, but is otherwise arbitrary.

We then generalize the encoder mapping of Section 3.1 as follows. Let $\mathscr{X}$ be the set of zeros of $\mathcal{T}(Y, Z)$ in $\mathbb{K}^2$, namely

$$\mathscr{X} \overset{\text{def}}{=} \{\, (\beta, \gamma) \in \mathbb{K} \times \mathbb{K} \;:\; \mathcal{T}(\beta, \gamma) = 0 \,\} \tag{25}$$

We shall assume for the time being (but see the remark at the end of this section) that there is a way to enumerate the rational points of $\mathcal{T}(Y, Z)$ in polynomial time. That is, we assume that there is a bijection from $\{1, 2, \dots, |\mathscr{X}|\}$ to $\mathscr{X}$ that can be computed in time $\operatorname{polylog}|\mathscr{X}|$. We can then think of an encoder for our code as a mapping $\mathcal{E} : \mathscr{X} \to \mathbb{C}$ from rational points of $\mathcal{T}(Y, Z)$ to codewords.

We now describe this mapping. Given a $(\beta, \gamma) \in \mathscr{X}$, we first write β and γ as

$$\beta \;=\; f_0 + f_1\zeta + \cdots + f_{k-1}\zeta^{k-1}, \quad \text{with } f_i \in \mathbb{F}_q \tag{26}$$

$$\gamma \;=\; g_0 + g_1\zeta + \cdots + g_{k-1}\zeta^{k-1}, \quad \text{with } g_i \in \mathbb{F}_q \tag{27}$$

where ζ is a zero of $e(X)$ in $\mathbb{K}$ and, hence, $\{1, \zeta, \dots, \zeta^{k-1}\}$ is a polynomial basis for $\mathbb{K}$ over $\mathbb{F}_q$. Then $f_0, f_1, \dots, f_{k-1}$ and $g_0, g_1, \dots, g_{k-1}$ define polynomials $f(X)$ and $g(X)$ in $\mathbb{F}_q[X]$ of degree at most $k-1$. Henceforth, the encoder proceeds as in (15). Namely, the codeword $(c_1, c_2, \dots, c_n) \in \mathbb{C}$ corresponding to the rational point (β, γ) is given by

$$c_j \;=\; f(x_j) + \alpha g(x_j) \qquad \text{for } j = 1, 2, \dots, n \tag{28}$$

It should be obvious that the codes constructed in Section 3 are just a special case of the above, which results by taking $\mathcal{T}(Y, Z)$ as the absolutely irreducible polynomial $Z - Y^a$.

Do we gain anything from this generalization? As we shall see, the error-correction radius of the general codes is still given by (17). But their rate is no longer simply $k/2n$.

Proposition 11. *Let $\mathfrak{g}$ denote the genus of $\mathcal{T}(Y,Z)$, and assume that $\mathcal{T}(Y,Z)$ is absolutely irreducible. Then the rate R of the code $\mathbb{C}$ defined by* (25) – (28) *is bounded as follows*

$$R \geqslant \frac{k}{2n} + \frac{1}{2n}\left(\log_q\!\left(1-\frac{2\mathfrak{g}}{q^{k/2}}\right) - \log_q a\right) \tag{29}$$

$$R \leqslant \frac{k}{2n} + \frac{1}{2n}\left(\log_q\!\left(1+\frac{2\mathfrak{g}}{q^{k/2}}\right)\right) \tag{30}$$

Proof. Clearly, the rate of $\mathbb{C}$ is given by $R = \log_{q^2}|\mathscr{X}|/n$. By the Hasse-Weil bound [13], the number of rational points of $\mathcal{T}(Y,Z)$, counted with multiplicity, is bounded by

$$\Big|\#\,\text{zeros}[\mathcal{T}](\mathbb{K}) - (1+q^k)\Big| \leqslant 2\mathfrak{g}\sqrt{q^k} \tag{31}$$

Since the total degree of $\mathcal{T}(Y,Z)$ is a, the multiplicity of any rational point of $\mathcal{T}(Y,Z)$ is at most a. The proposition now follows from (31), upon straightforward manipulations. ∎

What Proposition 11 tells us is that, in principle, it should be possible to improve upon the simple trivariate codes of Section 3 by choosing an irreducible polynomial $\mathcal{T}(Y,Z)$ with many rational points. This would yield codes with the same error-correction radius whose rate is higher than $k/2n$. But not by much. For example, in the (reasonable) regime where $n = q$, $k = \Theta(n)$, and $m = O(1)$, it follows from Proposition 11 and (13) that $R \to k/2n$ as $n \to \infty$, *regardless of the choice* of the encoder polynomial $\mathcal{T}(Y,Z)$.

4.2. General trivariate interpolation decoding

The decoding commences exactly as before: we set up the interpolation set $\mathcal{P}$, compute the corresponding interpolation polynomial $\mathcal{Q}_m(X,Y,Z)$, then reduce it modulo $e(X)$ to obtain the polynomial $P(Y,Z) \in \mathbb{K}[Y,Z]$. In other words, Steps 1 and 2 of the decoding algorithm of Section 3.2 remain unchanged, and Lemmas 6, 7, 8 still apply.

In view of Lemma 8, we are now interested in all pairs $(\beta,\gamma) \in \mathbb{K}^2$ such that $P(\beta,\gamma) = 0$. However, as suggested by (31), there could be an *exponential number* of such pairs, at least in certain cases. The key to our decoding algorithm is, again, the *a priori* relationship between β and γ embedded into the encoder. We know *a priori* that $\mathcal{T}(\beta,\gamma) = 0$, so (β,γ) is a solution to the following system of equations

$$P(Y,Z) = 0 \quad \text{and} \quad \mathcal{T}(Y,Z) = 0 \tag{32}$$

The standard way to solve such a system of polynomial equations is to use *resultants*. Thus we compute

$$H(Y) \overset{\text{def}}{=} \text{Res}(P,\mathcal{T};Z) \tag{33}$$

The resultant of two polynomials is the determinant of their Sylvester matrix; for a precise definition of the resultant operation $\text{Res}(\cdot,\cdot;Z)$ see e.g. [4, p.158]. We will furthermore make use of the following lemma from [4, Chapter 3].

Lemma 12. *Let $P(Y,Z)$ and $\mathcal{T}(Y,Z)$ be arbitrary polynomials over a field $\mathbb{K}$, both having positive degree in Z, and let $H(Y) = \text{Res}(P,\mathcal{T};Z)$. Then*

a. *The polynomial $H(Y)$ belongs to the ideal generated by $P(Y,Z)$ and $\mathcal{T}(Y,Z)$. That is, there exist polynomials $\mathcal{A}(Y,Z)$ and $\mathcal{B}(Y,Z)$ in $\mathbb{K}[Y,Z]$ such that*

$$H(Y) = \mathcal{A}(Y,Z)P(Y,Z) + \mathcal{B}(Y,Z)\mathcal{T}(Y,Z) \tag{34}$$

b. *The polynomial $H(Y)$ is the all-zero polynomial if and only if the polynomials $P(Y,Z)$, $\mathcal{T}(Y,Z)$ have a common factor in $\mathbb{K}[Y,Z]$ which has positive degree in Z.*

Lemma 13. *The resultant polynomial $H(Y)$ given by* (33) *is not the all-zero polynomial. Moreover, if (β,γ) is a solution of the system* (32), *then $H(\beta) = 0$.*

Proof. This follows easily from Lemma 12. Strictly speaking, in order to apply this lemma, we need to make sure that $P(Y,Z)$ has positive degree in Z. But we can assume this w.l.o.g., otherwise compute the resultant in (33) with respect to Y rather than Z. Now, the fact that $H(\beta) = 0$ follows by evaluating both sides of (34) at the point (β,γ) and using (32). Since $\mathcal{T}(Y,Z)$ is irreducible, it follows from Lemma 12.b that $H(Y) \equiv 0$ if and only if $\mathcal{T}(Y,Z)$ *divides* $P(Y,Z)$. Our condition for a in (13) again guarantees that $\deg_{\text{tot}}\mathcal{T}(Y,Z) > \deg_{\text{tot}}P(Y,Z)$, so this cannot happen. ∎

Given a $\beta \in \mathbb{K}$, we can find all $\gamma \in \mathbb{K}$ such that (β,γ) is a solution of (32) by computing the set of common roots of the univariate polynomials $P(\beta,Z)$ and $\mathcal{T}(\beta,Z)$. To summarize, decoding the general trivariate codes of Section 4.1 amounts to modifying the last two steps of the decoding algorithm of Section 3.2 as follows:

$\boxed{3'}$ Compute the resultant $H(Y) = \text{Res}(P,\mathcal{T};Z) \in \mathbb{K}[Y]$.

$\boxed{4'}$ Find all the roots of $H(Y)$ in $\mathbb{K}$. For each such root β, find all $\gamma \in \mathbb{K}$ such that $P(\beta,\gamma) = 0$ and $\mathcal{T}(\beta,\gamma) = 0$. Output the resulting list of pairs (β,γ).

The error-correction radius of this algorithm is still given by (17) and (23), and the size of the resulting list is still bounded by a^2, where a is given by (13). The latter observation follows from Bézout's Theorem [4, p. 420]: since $P(Y,Z)$ and $\mathcal{T}(Y,Z)$ have no common factors, the number of solutions to (32) is at most the product of their total degrees.

Remark. Observe that $\text{Res}(P,\mathcal{T};Z) = P(Y,Y^a)$ in the special case $\mathcal{T}(Y,Z) = Z - Y^a$. Thus the decoding algorithm of Section 3.2 is just an instance of the general algorithm above. While there are no problems with decoding, the encoder assumes the existence of a polynomial-time algorithm for mapping information into rational points of $\mathcal{T}(Y,Z)$. In general, finding such an algorithm is *hard*. So far, we are not aware of any choices for $\mathcal{T}(Y,Z)$ that yield codes of rate $R > k/2n$ and come equipped with an efficient encoder.

5. Extension to multivariate interpolation

We finally discuss the general case $M \geqslant 2$, where the encoder uses $M-1$ irreducible polynomials $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{M-1}$. While, in principle, these polynomials can be essentially arbitrary (cf. Section 4), we will only consider the special case where $\mathcal{T}_i(Y, Z_i) = Z_i - Y^{a_i}$ for all i (cf. Section 3).

Code parameters and encoder mapping. Our code $\mathbb{C}$ will now be a subset of $\mathbb{F}_Q^n$, where $Q = q^M$. We modify the list of code parameters in Section 3.1 as follows:

Basis: A fixed basis $\{1, \alpha_1, \ldots, \alpha_{M-1}\}$ for $\mathbb{F}_Q$ over $\mathbb{F}_q$.

Exponentiation degrees: A sequence of $M-1$ positive integers $a_1, a_2, \ldots, a_{M-1}$ given by

$$a_i \overset{\text{def}}{=} \sum_{j=0}^{i} \left\lceil \sqrt[M+1]{\frac{n}{k-1} \prod_{\ell=0}^{M} (m+\ell)} \right\rceil^j \tag{35}$$

All other parameters remain unchanged. Given k arbitrary symbols $u_0, u_1, \ldots, u_{k-1} \in \mathbb{F}_q$, the encoder first constructs the polynomial $f(X) = \sum_{i=0}^{k-1} u_i X^i$, and then computes

$$g_i(X) = \big(f(X)\big)^{a_i} \mod e(X) \tag{36}$$

for $i = 1, 2, \ldots, M-1$. The codeword $(c_1, c_2, \ldots, c_n) \in \mathbb{C}$ corresponding to the message $u_0, u_1, \ldots, u_{k-1}$ is given by

$$c_j = f(x_j) + \sum_{i=1}^{M-1} \alpha_i g_i(x_j) \qquad \text{for } j = 1, 2, \ldots, n \tag{37}$$

It is obvious from the above description that the rate of $\mathbb{C}$ is $R = \log_Q |\mathbb{C}|/n = k/Mn$. The minimum distance of $\mathbb{C}$ is at least $n-k+1$, since $\mathbb{C}$ is again a subset of a Reed-Solomon code of length n and dimension k over $\mathbb{F}_Q$.

Multivariate Decoding. Given a vector $\underline{v} = (v_1, v_2, \ldots, v_n)$ over $\mathbb{F}_Q$, we decompose each v_j into its components over $\mathbb{F}_q$ using the basis $\{1, \alpha_1, \ldots, \alpha_{M-1}\}$. Thus we write

$$v_j = y_j + \sum_{i=1}^{M-1} \alpha_i z_{i,j} \qquad \text{for } j = 1, 2, \ldots, n$$

where y_j and $z_{i,j}$ are in $\mathbb{F}_q$. We then set up the interpolation set $\mathcal{P}$ consisting of the n points $(x_j, y_j, z_{1,j}, \ldots, z_{M-1,j})$. As before, the first decoding step consists of computing the *interpolation polynomial* $\mathcal{Q}_m(X, Y, Z_1, \ldots, Z_{M-1})$, defined as the least $(1, k-1, k-1, \ldots, k-1)$-weighted degree polynomial that has a zero of multiplicity m at each point of $\mathcal{P}$.

Lemma 14. *Suppose that a codeword* $\underline{c} \in \mathbb{C}$ *differs from the given vector* $\underline{v} = (v_1, v_2, \ldots, v_n)$ *in at most*

$$t = \left\lfloor n - n \sqrt[M+1]{\left(\frac{k-1}{n}\right)^M \left(1 + \frac{1}{m}\right) \cdots \left(1 + \frac{M}{m}\right)} - \frac{1}{m} \right\rfloor$$

positions, and let $f(X)$, $g_1(X), g_2(X), \ldots, g_{M-1}(X)$ *denote the message polynomial(s) that produce* $\underline{c}$ *according to the mapping* (36) – (37). *Then the interpolation polynomial satisfies* $\mathcal{Q}_m\big(X, f(X), g_1(X), \ldots, g_{M-1}(X)\big) \equiv 0$.

Proof. The key observation is that the weighted-degree of the interpolation polynomial is at most

$$\Delta_M \overset{\text{def}}{=} \left\lceil \sqrt[M+1]{n(k-1)^M \prod_{\ell=0}^{M} (m+\ell)} \right\rceil \tag{38}$$

Since the number of interpolation constraints is now given by $n(M+m)!/(M+1)!(m-1)!$, this follows from a simple geometric argument as in Lemma 3. The rest is an easy generalization of Lemma 4 and Lemma 6. ▮

Next, as in (18), we reduce the interpolation polynomial mod $e(X)$ to obtain the polynomial $P(Y, Z_1, \ldots, Z_{M-1})$ in the ring $\mathbb{K}[Y, Z_1, \ldots, Z_{M-1}]$, where $\mathbb{K}$ is defined by (16). Then exactly the same argument as in Lemma 7 shows that $P(Y, Z_1, \ldots, Z_{M-1})$ is not the all-zero polynomial. Moreover, under the conditions of Lemma 14, we have

$$P(\beta, \gamma_1, \ldots, \gamma_{M-1}) = P(\beta, \beta^{a_1}, \ldots, \beta^{a_{M-1}}) = 0 \tag{39}$$

where β and $\gamma_1, \gamma_2 \ldots, \gamma_{M-1}$ are the elements of $\mathbb{K}$ corresponding to $f(X)$ and to $g_1(X), g_2(X), \ldots, g_{M-1}(X)$, respectively. We omit the proof of (39), since it is a straightforward generalization of the arguments in Lemma 5 and Lemma 8. The third decoding step consists of computing

$$H(Y) \overset{\text{def}}{=} P(Y, Y^{a_1}, Y^{a_2}, \ldots, Y^{a_{M-1}}) \tag{40}$$

The fourth and final decoding step is exactly as before: find all the roots of $H(Y)$ in $\mathbb{K}$, and output this list. It is obvious from (39) that, under the conditions of Lemma 14, β is indeed a root of $H(Y)$. There is only one thing left to prove.

Lemma 15. *The polynomial* $H(Y)$ *defined in* (40) *is not the all-zero polynomial.*

Proof. Consider the M polynomials $P_0, P_1, \ldots, P_{M-1}$ defined as follows: P_i is a polynomial in the $M-i$ variables $Y, Z_{i+1}, Z_{i+2}, \ldots, Z_{M-1}$ defined by the recursive rule

$$P_i(Y, Z_{i+1}, \ldots, Z_{M-1}) \overset{\text{def}}{=} P_{i-1}(Y, Y^{a_i}, Z_{i+1}, \ldots, Z_{M-1})$$

To initialize this recursion, set $P_0 = P(Y, Z_1, \ldots, Z_{M-1})$, where $P(Y, Z_1, \ldots, Z_{M-1})$ is the interpolation polynomial modulo $e(X)$, as above. It is then clear that $H(Y) = P_{M-1}$. To establish the lemma, we will show by induction on i that none of the polynomials $P_0, P_1, \ldots, P_{M-1}$ is the all-zero polynomial. As induction hypothesis, assume that P_{i-1} is nonzero. As noted above (see also Lemma 7), this is indeed true for $i = 1$. By induction hypothesis, P_i is the all-zero polynomial if and only if $Z_i - Y^{a_i}$ is a factor of P_{i-1}. Thus it would suffice to show that $a_i > \deg_{\text{tot}} P_{i-1}$. Observe that

$$\deg_{\text{tot}} P_{i-1} \leqslant \max\{a_1, a_2, \ldots, a_{i-1}\} \deg_{\text{tot}} P_0 \tag{41}$$

and since $a_1 < a_2 < \cdots < a_{M-1}$ in view of (35), it follows that $\deg_{\text{tot}} P_{i-1} \leqslant a_{i-1} \deg_{\text{tot}} P_0$. Thus it remains to show that $a_i > a_{i-1}\,\deg_{\rm tot} P_0$. But $\deg_{\rm tot} P_0 \leqslant \Delta_M/(k-1)$, as in Lemma 9, where Δ_M is defined by (38). The induction step now follows by observing that $a_i/a_{i-1} > \Delta_M/(k-1)$. ∎

As a by-product of the proof of Lemma 15, we also get an upper bound on the degree of $H(Y)$. Applying (41) for the special case $i-1 = M-1$, we find that

$$\deg H(Y) \leqslant a_{M-1}\,\deg_{\rm tot} P_0 \leqslant \frac{\Delta_M a_{M-1}}{k-1} \tag{42}$$

Since the list-size of our decoder is obviously bounded by $\deg H(Y)$, this completes the proof of Theorem 1. Using the fact that $R = k/Mn$, the expression for the error-correction radius t follows readily from Lemma 14 while the expression for the list-size L follows from (35), (38), and (42).

Acknowledgments. We are grateful to Venkat Guruswami, Ralf Koetter, and Madhu Sudan for very useful discussions. We would also like to thank the FOCS program committee and the anonymous referees, whose comments have greatly improved the presentation of our work.

References

[1] E.R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, New York, 1968.

[2] D. Bleichenbacher, A. Kiayias, and M. Yung. Decoding of interleaved Reed-Solomon codes over noisy data, *Lect. Notes in Comp. Sci.*, **2719**: 97–108, 2003.

[3] D. Coppersmith and M. Sudan. Reconstructing curves in three (and higher) dimensional spaces from noisy data, *Proc. 35th ACM Symp. on Theory of Computing (STOC)*, pp. 136–142, San Diego, CA., June 2003.

[4] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms*. 2-nd Edition, Springer, New York, 1996.

[5] P. Elias. List decoding for noisy channels, Tech. Report 335, Research Lab. Electronics, MIT, 1957.

[6] J. von zur Gathen and M. Nöcker. Exponentiation in finite fields: Theory and practice, *Lecture Notes in Computer Science*, **1255**: 88–113, 1997.

[7] V. Guruswami. *List Decoding of Error-Correcting Codes*, MIT Ph.D. Thesis, 2001; also *Lecture Notes in Computer Science*, **3282**, Springer, New York, 2005.

[8] V. Guruswami. Better extractors for better codes?, *Proc. 36th ACM Symposium on Theory of Computing (STOC)*, pp. 436–444, Chicago, IL., June 2004.

[9] V. Guruswami. Algebraic-geometric generalizations of the Parvaresh-Vardy codes, preprint, July 2005.

[10] V. Guruswami and P. Indyk. Expander-based constructions of efficiently decodable codes, *Proc. 42nd Annual Symposium on Foundactions of Computer Science (FOCS)*, pp. 658–667, Las Vegas, NV., October 2001.

[11] V. Guruswami and P. Indyk. Near-optimum linear-time codes for unique decoding and new list-decodable codes over smaller alphabets, *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 812–821, Montreal, Quebec, May 2002.

[12] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometric codes, *IEEE Trans. Inform. Theory*, **45**: 1755–1764, Sept. 1999.

[13] H. Hasse. *Number Theory*. Springer, Berlin, 1949.

[14] R. Kötter, *On Algebraic Decoding of Algebraic-Geometric and Cyclic Codes*, Ph.D. Thesis, University of Linköping, Sweden, 1996.

[15] R. Koetter and A. Vardy, Algebraic soft-decision decoding of Reed-Solomon codes, *IEEE Trans. Inform. Theory*, **49**: 2809–2825, November 2003.

[16] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*. Elsevier, Amsterdam, 1978.

[17] V.N. Muralindhara and S. Sen. On the tightness of the Johnson bound for Reed-Solomon codes, *16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Miami, FL., submitted for publication, 2005.

[18] F. Parvaresh and A. Vardy. Multivariate interpolation decoding beyond the Guruswami-Sudan radius, *Proc. 42nd Annual Allerton Conference on Communication, Control and Computing*, Urbana, IL., October 2004.

[19] F. Parvaresh and A. Vardy. Multivariate interpolation decoding of Reed-Solomon codes, preprint, 2005.

[20] T.J. Richardson and R.L. Urbanke. *Modern Coding Theory*. Book preprint, August 2005, draft available at http://lthcwww.epfl.ch/mct/.

[21] V. Shoup. A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic, *Proc. Intern. Symp. Symbolic and Algebraic Computation (ISSAC)*, pp. 14–21, Bonn, Germany, July 1991.

[22] M. Sudan, Decoding of Reed-Solomon codes beyond the error correction bound, *J. Complexity*, **12**: 180–193, March 1997.

[23] L.R. Welch and E.R. Berlekamp. *Error correction for algebraic block codes*, US Patent No. 4,633,470, 1986.

[24] J.M. Wozencraft. List decoding, *Quart. Progress Report*, Research Lab. Electronics, MIT, **48**: 90–95, 1958.