Electronic Colloquium on Computational Complexity, Report No. 73 (2012)

Linear-algebraic list decoding for variants of Reed-Solomon codes *

VENKATESAN GURUSWAMI† CAROL WANG‡

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Folded Reed-Solomon codes are an explicit family of codes that achieve the optimal trade-off between rate and list error-correction capability: specifically, for any $\varepsilon > 0$, Guruswami and Rudra presented an $n^{O(1/\varepsilon)}$ time algorithm to list decode appropriate folded RS codes of rate R from a fraction $1 - R - \varepsilon$ of errors. The algorithm is based on multivariate polynomial interpolation and root-finding over extension fields. It was noted by Vadhan that interpolating a linear polynomial suffices for a statement of the above form. Here we give a simple linear-algebra based analysis of this variant that eliminates the need for the computationally expensive root-finding step over extension fields (and indeed any mention of extension fields). The entire list decoding algorithm is linear-algebraic, solving one linear system for the interpolation step, and another linear system to find a small subspace of candidate solutions. Except for the step of pruning this subspace, the algorithm can be implemented to run in *quadratic* time.

We also consider a closely related family of codes, called (order m) *derivative codes* and defined over fields of large characteristic, which consist of the evaluations of f as well as its first $m - 1$ formal derivatives at N distinct field elements. We show how our linear-algebraic methods for folded Reed-Solomon codes can be used to show that derivative codes can also achieve the above optimal trade-off.

The theoretical drawback of our analysis for folded RS codes and derivative codes is that both the decoding complexity and proven worst-case list-size bound are $n^{\Omega(1/\varepsilon)}$. By combining the above idea with a *pseudorandom subset* of all polynomials as messages, we get a Monte Carlo construction achieving a list size bound of $O(1/\varepsilon^2)$ which is quite close to the existential $O(1/\varepsilon)$ bound (however, the decoding complexity remains $n^{\Omega(1/\varepsilon)}$).

Our work highlights that constructing an explicit *subspace-evasive* subset that has small intersection with low-dimensional subspaces — an interesting problem in pseudorandomness in its own right — has applications to constructing explicit codes with better list-decoding guarantees.

*Preliminary conference versions of the results in this paper appeared as [9] and [15]. This is a merged and revised version of these papers.

†Supported in part by a Packard Fellowship and NSF grant CCF 0963975. Email: guruswami@cmu.edu

‡Research supported by an NSF graduate fellowship, NSF grants CCF 0963975 and CCF 0953155, and a grant from the MSR-CMU Center for Computational Thinking. Email: wangc@cs.cmu.edu

ISSN 1433-8092

1 Introduction

1.1 Background and context

Reed-Solomon (RS) codes are an important family of error-correcting codes with many applications in theory and practice. An $[n, k]_q$ *RS code* over the field $\mathbb{F}_q$ with q elements encodes polynomials $f \in \mathbb{F}_q[X]$ of degree at most $k-1$ by its evaluations at n distinct elements from $\mathbb{F}_q$. The encodings of any two distinct polynomials differ on at least $n-k+1$ positions, which bestows the RS code with an error-correction capability of $(n-k)/2$ worst-case errors. Classical algorithms, the first one due to Peterson [22] over 50 years ago, are able to decode such a RS code from up to $(n-k)/2$ errors (i.e., a fraction $(1-R)/2$ of errors, where $R = k/n$ is the rate of the code) in polynomial time.

Decoding beyond the radius $(1-R)/2$ is not possible if the decoder is required to always identify the correct message unambiguously. However, allowing the decoder to output a small list in the worst case enables decoding well beyond this bound. This notion is called *list decoding*, and has been an actively researched topic in the last decade. It has found many applications in complexity theory and pseudorandomness (see [25, 26, 28] for some surveys) beyond its direct relevance to error-correction and communication.

For RS codes, Sudan [24] gave a list decoding algorithm to decode beyond the $(1-R)/2$ radius for rates $R < 1/3$. For rates $R \to 0$, the algorithm could correct a fraction of errors approaching 1, a remarkable feature that led to many complexity-theoretic applications. Guruswami and Sudan [14] improved the error-correction radius to $1-\sqrt{R}$, matching the so-called "Johnson radius," which is the a priori lower bound on list-decoding radius of a code as a function of its distance alone. This result improved upon the traditional $(1-R)/2$ bound for *all* rates. The $1-\sqrt{R}$ bound remains the best error-correction radius achievable to date for list decoding RS codes.

A standard random coding argument, however, shows the *existence* of rate R codes $C \subseteq \Sigma^n$ list-decodable even up to radius $1-R-\varepsilon$. Specifically, C has the combinatorial property that for every $y \in \Sigma^n$, there are at most $L = O(1/\varepsilon)$ codewords of C within Hamming distance $(1-R-\varepsilon)n$ from y. Here $\varepsilon > 0$ can be an arbitrarily small constant, and the alphabet size $|\Sigma|$ is a large constant depending on ε. The quantity L is referred to as the *list-size*. Note that $1-R$ is a clear information-theoretic limit for error-correction, since at least roughly Rn received symbols must be correct to have any hope of recovering the Rn message symbols.

A few years back, Guruswami and Rudra [13], building upon the work of Parvaresh and Vardy [21], gave an *explicit* construction of codes of rate R which are list-decodable in polynomial time up to radius $1-R-\varepsilon$, with a list-size of $n^{O(1/\varepsilon)}$. These codes were a "folded" version of Reed-Solomon codes, defined below.

Definition 1 (m-folded Reed-Solomon code)**.** *Let $\gamma \in \mathbb{F}_q$ be a primitive element of $\mathbb{F}_q$. Let $n \leqslant q-1$ be a multiple of m, and let $1 \leqslant k < n$ be the* degree parameter*.*

The folded Reed-Solomon (FRS) code $\mathrm{FRS}_q^{(m)}[n, k]$ *is a code over alphabet $\mathbb{F}_q^m$ that encodes a poly-*

nomial $f \in \mathbb{F}_q[X]$ *of degree at most* $k-1$ *as*[1]

$$f(X) \mapsto \left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^m) \\ f(\gamma^{m+1}) \\ \vdots \\ f(\gamma^{2m-1}) \end{bmatrix}, \ldots, \begin{bmatrix} f(\gamma^{n-m}) \\ f(\gamma^{n-m+1}) \\ \vdots \\ f(\gamma^{n-1}) \end{bmatrix} \right). \tag{1}$$

The block length *of* $\mathrm{FRS}_q^{(m)}[n,k]$ *is* $N = n/m$*, and its* rate *is* $R = k/n$.

Observe that the FRS code has rate equal to the rate of the original, unfolded Reed-Solomon code, which corresponds to the choice $m = 1$. The difference is that the FRS code is defined over a larger alphabet, of size q^m. Thus when we deal with *symbol* errors for the larger alphabet, a correct codeword symbol over this large alphabet gives the correct value of the message polynomial f at m correlated points. Somewhat remarkably, this feature can be exploited to correct a number of errors approaching the information-theoretic limit of $(1-R)N$ as the parameter m is chosen large enough. Specifically, for any integer s, $1 \leqslant s \leqslant m$, a list decoding algorithm for the above FRS codes for a fraction $\approx 1 - \left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$ of errors is presented in [13], with decoding complexity $q^{O(s)}$ and list-size q^s. Given the close relation of the FRS codes to Reed-Solomon codes, the result of [13] can also be viewed as a better algorithm for list decoding Reed-Solomon codes when the errors occur in bursts, since the evaluation points of the RS encoding are usually ordered as powers of γ for some primitive γ.

For suitably large constants s, m depending on ε, the above list decoding radius for FRS codes exceeds $1 - R - \varepsilon$. However, it turns out that the list-size bound then becomes $n^{\Omega(1/\varepsilon)}$, which has a rather poor dependence on the distance ε to the optimal trade-off. Improving the list-size is therefore an important open question. Recall that existentially a list-size as small as $O(1/\varepsilon)$ is possible. The decoding algorithms in [21, 13] consist of two steps (see Section 2.1 for more details): (i) multivariate polynomial interpolation (to find an algebraic equation that candidate message polynomials f must satisfy), and (ii) solving this equation via root-finding over extension fields. The interpolation step reduces to finding a nonzero solution to a homogeneous linear system, and theoretically the second step is the computationally more expensive one.

Vadhan showed recently that a decoding radius of $1 - R - \varepsilon$ can be achieved by a simplified interpolation step that only interpolates a degree 1 multivariate polynomial [27]. The second root-finding step of the decoder remains unchanged. Further, there is *no need* to use multiplicities in the interpolation as in the earlier algorithms [14, 21, 13].[2] This offers a clean and simple exposition of a list decoding algorithm for FRS codes (that can be viewed as a multidimensional version of the Welch-Berlekamp decoder for RS codes) for a fraction $\approx \frac{s}{s+1}\left(1 - \frac{mR}{m-s+1}\right)$ of errors (see Section 2.2). This is weaker than the decoding radius of $\approx 1 - \left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$ using multiplicities, but still suffices to decode up to radius $1 - R - \varepsilon$ for constant $\varepsilon > 0$.

[1] The actual code depends also on the choice of the primitive element γ. But the results hold for any choice of primitive γ, so for notational convenience we suppress the dependence on γ and assume some canonical choice of γ is fixed.

[2] However, the method of multiplicities is still crucial if one wants a soft-decision list decoder, which, at least for Reed-Solomon codes, has been a very influential development [18], with many subsequent papers looking at practical decoding architectures.

1.2 Contributions of this work

Our contributions are three-fold.

1.2.1 List decoding folded Reed-Solomon codes

Here, we note that above-mentioned Welch-Berlekamp style "degree 1" list decoder, not only offers a simpler exposition, but also offers some promising advantages. Our starting point is the simple observation that in this case the candidate solutions to the algebraic equation form an affine subspace (of the full message space $\mathbb{F}_q^k$). This implies that the second step of the list decoding can also be tackled by solving a linear system!

By inspecting the structure of this linear system, we give an elementary linear-algebraic proof (Lemma 6) that the subspace of solutions has dimension at most $s-1$, a fact that was earlier proved by root counting over extension fields in [13, 27]. This shows that the exponential dependence in s of the list-size bound was inherently because of the dimension of the interpolation (and it was not crucial that we had the identity $f(\gamma^{s-1}X) = f(X)^{q^{s-1}}$ over some extension field[3]).

The linear-algebraic proof also gives a *quadratic* time algorithm to find a basis for the subspace (instead of the cubic time of Gaussian elimination). This leads to a quadratic runtime for the list decoder, *except* for the final step of pruning the subspace to actually find the close-by codewords (formal statement in Theorem 7). This pruning step needs to check each element of the subspace and thus unfortunately could still take q^s time. However, initial experiments with random error models indicate that the output dimension is likely to be small (often even 0, implying a unique solution), and in these cases the actual bottleneck in the runtime is the interpolation step rather than the second step.

1.2.2 Derivative codes

We also consider another natural variant of Reed-Solomon codes (over fields of large characteristic), called *derivative codes*, defined formally in Section 3. These are a special case of the multivariate *multiplicity codes* which were used in [20] to give locally decodable codes. We refer to our codes instead as *derivative codes* in order to single out the important univariate case with a different name. [4]

Informally, rather than bundling together evaluations of the message polynomial at consecutive powers of γ, in an order-m derivative code, we bundle together the evaluations of f as well as its first $(m-1)$ derivatives at each point. This might appear to cause a loss in rate (similar to the Parvaresh-Vardy construction [21]), but it does not, as one can pick higher degree polynomials while still maintaining the distance. (For two distinct degree ℓ polynomials, there can be at most ℓ/m points where they and their first $(m-1)$ derivatives agree.)

In Theorem 17 and Corollary 18, we show our main result that derivative codes also achieve

[3] This identity, however, seems to be the only known way to bound the list-size when higher degrees are used in the interpolation, as in [13].

[4] The change in name also highlights the fact that we use formal derivatives rather than Hasse derivatives in the encoding.

the optimal rate vs. list-decodability trade-off. Formally, for any $\varepsilon > 0$, for the choice $m \approx 1/\varepsilon^2$, we can list decode order-m derivative codes of rate R from a $1 - R - \varepsilon$ fraction of errors. The list-size and running time behavior is similar to the linear-algebraic algorithm for folded RS codes, and once again one can find, by just solving two linear systems, a low-dimensional affine space that contains all the close-by codewords.

It is natural to try to find other codes which achieve the optimal trade-off between rate and list decoding radius, and the derivative code construction is arguably just as natural as the folded Reed-Solomon one. Interestingly, it falls in the framework of Parvaresh-Vardy codes, where the correlated polynomials are formal derivatives. The special properties of derivatives ensures that one need not suffer any loss in rate, and at the same time enable list decoding up to a much larger radius than the bound for RS codes. Further, our algorithm for list decoding derivative codes offers some advantages with respect to efficient (unique) decoding with side information (see Section 3.2), and might have some benefits in practice as well.

1.2.3 Better list-size via subspace-evasive sets

Our third contribution is to exploit the subspace structure of the candidate solutions to improve the list-size bound. The idea is to restrict the coefficient vectors of the message polynomial to a large *"subspace-evasive" subset* that has small intersection with subspaces of low dimension. Subspace-evasive sets seem like fundamental combinatorial objects interesting in their own right. They are related to affine extractors, and also have applications to constructing bipartite Ramsey graphs [23]. As one would expect, a random set has excellent subspace-evasiveness, but finding matching explicit constructions is open. Our application to list decoding in this work provides another motivation for the interesting problem of constructing subspace-evasive sets.

Using a pseudorandom construction of subspace-evasive subsets (in fact, algebraic varieties) based on limited independence, we give a Monte Carlo construction (succeeding with high probability) of rate R codes list-decodable up to a fraction $1 - R - \varepsilon$ of errors with a list-size of $O(1/\varepsilon^2)$ (Theorem 24 gives the exact statement). Due to the pruning step, the worst-case runtime is however still $n^{\Omega(1/\varepsilon)}$. Nevertheless, this is the first construction with a better than $n^{\Omega(1/\varepsilon)}$ list-size for decoding up to the information-theoretic limit of $1 - R - \varepsilon$ fraction of errors.

We stress that only our code construction is randomized, and once it succeeds (which happens with high probability), the list decoding properties hold for every received word, and the encoding and list decoding procedures run in deterministic polynomial time.

1.3 Organization

We describe the list decoding algorithm for FRS codes and our linear-algebraic analysis of it in Section 2, concluding the section with some related remarks about the linear algebra approach. We describe derivative codes and their list-decoding in Section 3. We use subspace-evasive sets to give our Monte Carlo construction of codes achieving list decoding capacity with improved list-size in Section 4. Finally, we briefly discuss some follow-up work that this paper inspired in Section 5.

2 List decoding folded Reed-Solomon codes

Suppose a codeword of the m-folded RS code (Definition 1) was transmitted and we received a string in $\mathbf{y} \in (\mathbb{F}_q^m)^N$ which we view as an $m \times N$ matrix over $\mathbb{F}_q$ (recall $N = n/m$):

$$\begin{pmatrix} y_0 & y_m & & y_{n-m+1} \\ y_1 & y_{m+1} & & \vdots \\ y_2 & y_{m+2} & & \vdots \\ & & \ddots & \\ y_{m-1} & \cdots & & y_{n-1} \end{pmatrix} \tag{2}$$

We would like to recover a list of all polynomials $f \in \mathbb{F}_q[X]$ of degree $k-1$ whose folded RS encoding (1) agrees with $\mathbf{y}$ in at least $N - e$ columns, for some error bound e. Note that an agreement means that all m values in that particular column match. The following theorem is from [13].

Theorem 2. *For the folded Reed-Solomon code* $\mathrm{FRS}_q^{(m)}[n,k]$ *of block length* $N = n/m$ *and rate* $R = k/n$*, the following holds for all integers* s*,* $1 \leqslant s \leqslant m$*. Given a received word* $\mathbf{y} \in (\mathbb{F}_q^m)^N$*, in* $\left(O_\delta(q)\right)^{O(s)}$ *time, one can find a list of size at most* q^s *that contains all message polynomials* $f \in \mathbb{F}_q[X]$ *of degree less than* k *whose FRS encoding (1) differs from* $\mathbf{y}$ *in at most a fraction*

$$1 - (1+\delta)\left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$$

of the N *codeword positions.*

Note that the fractional agreement required by this algorithm as a function of the rate $R = k/n = k/(Nm)$ is

$$(1+\delta)\left(\frac{mR}{m-s+1}\right)^{s/(s+1)}. \tag{3}$$

By picking $\delta \approx \varepsilon$, $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$, the above quantity is at most $R + \varepsilon$, and the decoding complexity and list size are $\approx q^{O(1/\varepsilon)}$.

2.1 Overview of above decoding algorithm

We briefly recap the high level structure of this decoding algorithm. The quantity s is a parameter of the algorithm. In the first step, the algorithm interpolates a nonzero multivariate polynomial $Q \in \mathbb{F}_q[X, Y_1, Y_2, \ldots, Y_s]$ of low *weighted degree* (where the Y_i's have weight $k-1$ and X has weight 1) with the following guarantee:

> For every i, $0 \leqslant i < n$ with $i \mod m \in \{0, 1, 2, \ldots, m-s\}$, $Q(X, Y_1, \ldots, Y_s)$ vanishes at $(\gamma^i, y_i, y_{i+1}, \ldots, y_{i+s-1}) \in \mathbb{F}_q^{s+1}$ with high multiplicity (related to the other parameter δ of the algorithm).

Such a polynomial Q can be found in polynomial time by solving a homogeneous linear system over $\mathbb{F}_q$. The degree and multiplicity parameters in the interpolation step are carefully picked to ensure the following two properties: (i) a nonzero Q meeting the interpolation requirements exists, and (ii) every $f \in \mathbb{F}_q[X]$ of degree at most $(k-1)$ whose FRS encoding agrees with $\mathbf{y}$ on at least $N-e$ places (and which, therefore, must be output by the list decoder) satisfies the functional equation

$$Q\big(X, f(X), f(\gamma X), \cdots, f(\gamma^{s-1}X)\big) = 0\,.$$

In the second step of the decoder, all solutions f to the above equation are found. This is done by observing that $f(\gamma X) = f(X)^q \pmod{E(X)}$ where $E(X) = (X^{q-1} - \gamma)$, and therefore $f \bmod E(X)$ can be found by finding the roots of the univariate polynomial

$$T(Y) = Q(X, Y, Y^q, \ldots, Y^{q^{s-1}}) \mod E(X)$$

with coefficients from $L = \mathbb{F}_q[X]/(E(X))$. The polynomial $E(X)$ is irreducible over $\mathbb{F}_q$ and therefore L is an *extension field*. The parameter choices ensure that $T \neq 0$, and thus T cannot have more than q^{s-1} roots, and these roots may all be found in polynomial time. Finally, this list is pruned to only output those polynomials whose FRS encoding is in fact close to the received word $\mathbf{y}$.

2.2 A Welch-Berlekamp style interpolation

We will now describe a variant of the above scheme where the interpolation step will fit a nonzero "linear" polynomial $Q(X, Y_1, Y_2, \ldots, Y_s)$ (with degree 1 in the Y_i's). This can be viewed as a higher-dimensional generalization of the Welch-Berlekamp algorithm [29, 6]. This elegant version is due to Vadhan and is described in his monograph [27, Chap. 5] (and also used in the first author's lecture notes [8]). For completeness, and because it will be convenient to refer to it in the second step, we give a self-contained presentation here.

The original motivation for this variant was that it had simpler parameter choices and an easier exposition (even though the error-correction guarantee worsened, it still allowed approaching a decoding radius of $1-R$ in the limit). In particular, it has the advantage of *not* requiring the use of multiplicities in the interpolation. (Essentially, the freedom to do s-variate interpolation for a parameter s of our choosing allows us to work with simple interpolation while still gaining in error-correction radius with increasing s. This phenomenon also occurred in one of the algorithms in [12] for list decoding correlated algebraic-geometric codes.)

In this work, our contribution is to put the simple linear structure of the interpolated polynomial to good use and exploit it to substitute the root-finding step with a more efficient step of solving a linear system. Our algorithm also works for any γ of order at least k, rather than requiring γ to be primitive as in the previous algorithm. See Section 2.4 for more details.

Given a received word as in (2) we will interpolate a nonzero polynomial

$$Q(X, Y_1, Y_2, \ldots, Y_s) = A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \cdots + A_s(X)Y_s \tag{4}$$

over $\mathbb{F}_q$ with the degree restrictions $\deg(A_i) \leqslant D$ for $i = 1, 2, \ldots, s$ and $\deg(A_0) \leqslant D + k - 1$, where the degree parameter D is chosen to be

$$D = \left\lfloor \frac{N \cdot (m - s + 1) - k + 1}{s + 1} \right\rfloor\,. \tag{5}$$

The number of monomials in a polynomial Q with these degree restrictions equals

$$(D+1)s + D + k = (D+1)(s+1) + k - 1 > N(m-s+1) \tag{6}$$

for the above choice (5) of D. The interpolation requirements on $Q \in \mathbb{F}_q[X, Y_1, \ldots, Y_s]$ are the following:

$$Q(\gamma^{im+j}, y_{im+j}, y_{im+j+1}, \cdots, y_{im+j+s-1}) = 0 \quad \text{for } i = 0, 1, \ldots, N-1,\ j = 0, 1, \ldots, m-s\,. \tag{7}$$

(Recall that $N = n/m$.) We then have the following.

Lemma 3. *Let*

$$D = \left\lfloor \frac{N(m-s+1) - k + 1}{s+1} \right\rfloor . \tag{8}$$

A nonzero $Q \in \mathbb{F}_q[X, Y_1, \ldots, Y_s]$ of the form (4) satisfying the interpolation conditions (7) with $\deg(A_0) \leqslant D + k - 1$ and $\deg(A_j) \leqslant D$ for $1 \leqslant j \leqslant s$ exists and can be found by solving a homogeneous linear system over $\mathbb{F}_q$ with at most n constraints and variables. Further, this interpolation can be performed in $O(n \log^2 n \log\log n)$ operations over $\mathbb{F}_q$.

Proof. This holds since the number of interpolation conditions $N \cdot (m - s + 1)$ is less than the number of degrees of freedom (monomials) in Q. Regarding the claimed runtime, even though the best known algorithms for solving a general $n \times n$ linear system take time n^ω where ω is the exponent of matrix multiplication (currently $\approx 2.37...$), the above linear system has a special structure (involving evaluations at powers of γ). This can be exploited to solve the system in near-linear runtime as shown in [2] (see Proposition 5.11 in Chapter 5). □

The following lemma shows that any such polynomial Q gives an algebraic condition that the message polynomials $f(X)$ we are interested in list decoding must satisfy.

Lemma 4. *Let Q satisfy the conclusion of Lemma 3. If $f \in \mathbb{F}[X]$ is a polynomial of degree at most $k-1$ whose FRS encoding (1) agrees with the received word $\mathbf{y}$ in at least t columns for $t > \frac{D+k-1}{m-s+1}$, then*

$$Q(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1} X)) = 0\,. \tag{9}$$

Proof. Define $\Lambda(X) = Q(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1} X))$. Due to the degree restrictions on Q, the degree of $\Lambda(X)$ is easily seen to be at most $D + k - 1$. If the FRS encoding of f agrees with $\mathbf{y}$ in the i'th column (for some $i \in \{0, 1, \ldots, N-1\}$), we have

$$f(\gamma^{im}) = y_{im}, \quad f(\gamma^{im+1}) = y_{im+1}, \quad \ldots \quad , f(\gamma^{im+m-1}) = y_{im+m-1}\,.$$

Together with the interpolation conditions (7), this implies $\Lambda(\gamma^{im+j}) = 0$ for $j = 0, 1, \ldots, m-s$. In other words, Λ picks up at least $m - s + 1$ distinct roots for each such column i. Thus Λ must have at least $t(m-s+1)$ roots in all. Since $\deg(\Lambda) \leqslant D + k - 1$, if $t > (D+k-1)/(m-s+1)$, we must have $\Lambda = 0$. □

For the choice of D in (5), the requirement on t in Lemma 4 is met if $t \cdot (m-s+1) > \frac{N \cdot (m-s+1) + s(k-1)}{s+1}$, and hence if the fractional agreement t/N satisfies

$$\frac{t}{N} \geqslant \frac{1}{s+1} + \frac{s}{s+1}\frac{k}{N(m-s+1)} = \frac{1}{s+1} + \frac{s}{s+1}\frac{mR}{m-s+1}\,. \tag{10}$$

In other words, the fractional agreement needed is $\frac{1}{s+1} + \frac{s}{s+1}\frac{mR}{m-s+1}$. (Recall that $R = k/n$ is the rate of the code.) Note that by the AM-GM inequality, this agreement is always higher than the agreement fraction $\left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$ needed in (3).[5] Thus this variant corrects a smaller fraction of errors. Nevertheless, with the choice $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$, the fraction of errors corrected can still exceed $1-R-\varepsilon$. Further, as we see next, it offers some advantages when it comes to retrieving the solutions f to (9).

2.3 Retrieving candidate polynomials f

By the preceding section, to complete the list decoding we need to find all polynomials $f \in \mathbb{F}_q[X]$ of degree at most $k-1$ that satisfy

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0\,. \tag{11}$$

We note the following simple but very useful fact:

Observation 5. *The above is a system of linear equations over $\mathbb{F}_q$ in the coefficients $f_0, f_1, \cdots, f_{k-1}$ of the polynomial $f(X) = f_0 + f_1 X + \cdots + f_{k-1}X^{k-1}$. Thus, the solutions $(f_0, f_1, \ldots, f_{k-1})$ of (11) form an affine subspace of $\mathbb{F}_q^k$.*

In particular, the above immediately gives an efficient algorithm to find a compact representation of all the solutions to (11) — simply solve the linear system! This simple observation is the starting point driving this work.

We next prove that when γ is primitive, the space of solutions has dimension at most $s-1$. Note that we *already* knew this by the earlier argument in Section 2.1 over the extension field $\mathbb{F}_q[X]/(X^{q-1}-\gamma)$. But it is instructive to give a direct proof of this working only over $\mathbb{F}_q$. The proof in fact works when the order of γ is at least k. Further, it exposes the simple structure of the linear system which can be used to find a basis for the solutions in quadratic time.

Lemma 6. *If the order of γ is at least k (in particular when γ is primitive), the affine space of solutions to (11) has dimension d at most $s-1$.*

Further, one can compute using $O(sk^2)$ field operations over $\mathbb{F}_q$ a lower-triangular matrix $H \in \mathbb{F}_q^{k\times k}$ of rank at least $k-s+1$ and a vector $\mathbf{z} \in \mathbb{F}_q^k$, such that the coefficient (column) vectors $\mathbf{f} = (f_0, f_1, \ldots, f_{k-1})^T$ of solutions to (11) are contained in the affine space $H\mathbf{f} = \mathbf{z}$.

Proof. First, by factoring out a common powers of X that divide all of $A_0(X), A_1(X), \ldots, A_s(X)$, we can assume that at least one $A_{i^*}(X)$ for some $i^* \in \{0, 1, \ldots, s\}$ is not divisible by X, and has nonzero constant term. Further, if $A_1(X), \ldots, A_s(X)$ are all divisible by X, then so is $A_0(X)$, so we can take $i^* > 0$.

Let us denote $A_i(X) = \sum_{\ell=0}^{D+k-1} a_{i,\ell}X^\ell$ for $0 \leqslant i \leqslant s$. (We know that the degree of $A_i(X)$ for $i \geqslant 1$ is at most D, so $a_{i,\ell} = 0$ when $i \geqslant 1$ and $\ell > D$, but for notational ease let us introduce these

[5] Recall that for Reed-Solomon codes ($m = 1$) this was also exactly the case: the classical algorithms unique decoded the codeword when the agreement fraction was at least $\frac{1+R}{2}$, and the list decoding algorithm in [14] list decoded from agreement fraction $\sqrt{R}$.

coefficients.) For $j = 0, 1, 2, \ldots, k-1$, define the polynomial

$$B_j(X) = a_{1,j} + a_{2,j}X + a_{3,j}X^2 + \cdots + a_{s,j}X^{s-1} \,. \tag{12}$$

We know that $a_{i^*,0} \neq 0$, and therefore $B_0 \neq 0$.

By Condition (11), for each $r = 0, 1, 2, \ldots$, the coefficient of X^r in the polynomial

$$\Lambda(X) = A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X)$$

equals 0.

The constant term of $\Lambda(X)$ equals $a_{0,0} + a_{1,0}f_0 + a_{2,0}f_0 + \cdots + a_{s,0}f_0 = a_{0,0} + B(1)f_0$. Thus if $B(1) \neq 0$, then f_0 is uniquely determined as $-a_{0,0}/B(1)$. If $B(1) = 0$, then $a_{0,0} = 0$ or else there will be no solutions to (11) and in that case f_0 can take an arbitrary value in $\mathbb{F}_q$.

The coefficient of X^r of $\Lambda(X)$, for $0 \leqslant r < k$, equals

$$\begin{aligned} &a_{0,r} + f_r \cdot (a_{1,0} + a_{2,0}\gamma^r + \cdots + a_{s,0}\gamma^{(s-1)r}) + f_{r-1} \cdot (a_{1,1} + a_{2,1}\gamma^{r-1} + \cdots + a_{s,1}\gamma^{(s-1)(r-1)}) + \qquad (13)\\ &\quad \cdots + f_1 \cdot (a_{1,r-1} + a_{2,r-1}\gamma + \cdots + a_{s,r-1}\gamma^{s-1}) + f_0 \cdot (a_{1,r} + \cdots + a_{s,r})\\ &\quad = B_0(\gamma^r)f_r + \left(\sum\nolimits_{j=1}^{r} B_j(\gamma^{r-j})f_{r-j}\right) + a_{0,r} \qquad (14)\end{aligned}$$

recalling the definition (12) of the polynomials B_j. The linear form (14) must thus equal 0. The key point is that if $B_0(\gamma^r) \neq 0$, then this implies that f_r is an affine combination of $f_0, f_1, \ldots, f_{r-1}$ and in particular is uniquely determined given values of $f_0, f_1, \ldots, f_{r-1}$.

Thus the dimension of the space of solutions is at most the number of r, $0 \leqslant r < k$, for which $B_0(\gamma^r) = 0$. Since γ has order at least k, the powers γ^r for $0 \leqslant r < k$ are all distinct. Also we know that B_0 is a *nonzero* polynomial of degree at most $s-1$. Thus $B_0(\gamma^r) = 0$ for at most $s-1$ values of r.

We have thus proved that the solution space has dimension at most $s-1$. To justify the claim about the decoding complexity and structure of solution space, note that the linear system satisfied by $\mathbf{f} = (f_0, f_1, \ldots, f_{k-1})^T$ is $H\mathbf{f} = \mathbf{z}$ where $\mathbf{z} = (-a_{0,0}, -a_{0,1}, \ldots, -a_{0,k-1})^T$ and the (r, j)'th entry of H equals $B_{r-j}(\gamma^j)$ for $j \leqslant r$ and 0 otherwise. The computation of H amounts to evaluating the polynomials B_j, $0 \leqslant j < k$, each of which has degree less than s, at the points $\{1, \gamma, \ldots, \gamma^{k-1}\}$. This can be accomplished in $O(sk^2)$ operations over $\mathbb{F}_q$. □

Combining Lemmas 3 and 6 and the decoding bound (10), we can conclude the following.

Theorem 7. *For the folded Reed-Solomon code* $\mathrm{FRS}_q^{(m)}[n, k]$ *of block length* $N = n/m$ *and rate* $R = k/n$*, the following holds for all integers* s*,* $1 \leqslant s \leqslant m$*. Given a received word* $\mathbf{y} \in (\mathbb{F}_q^m)^N$*, using* $O(n^2 + sk^2)$ *operations over* $\mathbb{F}_q$*, one can find a subspace of dimension at most* $s-1$ *that contains all message polynomials* $f \in \mathbb{F}_q[X]$ *of degree less than* k *whose FRS encoding (1) differs from* $\mathbf{y}$ *in at most a fraction*

$$\frac{s}{s+1}\left(1 - \frac{mR}{m-s+1}\right)$$

of the N *codeword positions.*

Note: When $s = m = 1$, the above just reduces to a unique decoding algorithm up to a fraction $(1 - R)/2$ of errors.

Choosing $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$ suffices to ensure decoding from a $1 - R - \varepsilon$ fraction of errors. Note that the decoding guarantee of Theorem 7 improves with s and as m increases relative to s. However, as s increases, so does our worst-case list size guarantee of q^{s-1}. For fixed parameters k and n, as m increases, the absolute number of errors which can be corrected decreases.

Comment on list size and runtime. To get the actual list of close-by codewords, one can prune the solution subspace, which unfortunately may take $q^s > n^s$ time in the worst-case. This quantity is about $n^{O(1/\varepsilon)}$ for the parameter choice $s \approx 1/\varepsilon$ which achieves a list decoding radius of $1 - R - \varepsilon$. Theoretically, we are not able to improve the worst-case list size bound of $\approx n^{1/\varepsilon}$ in this regime. This motivates our results in Section 4 where we show that using a carefully chosen subset of all possible degree at most $k - 1$ polynomials as messages, one can ensure that the list-size is much smaller while losing only a tiny amount in the rate.

Except for final step of pruning the subspace of candidate solutions, the decoding takes only quadratic time (and is perhaps even practical, as it just involves solving two structured linear systems). If some side information about the true message f is available that disambiguates the true message in the list [7], that might also be useful to speed up the pruning.

2.4 Some remarks

We now make some salient remarks about the above linear-algebra based method for list decoding folded Reed-Solomon codes..

Tightness of q^{s-1} bound. For folded Reed-Solomon codes, the upper bound of q^{s-1} on the number of solutions f to the Equation (11) cannot be improved in general. Indeed, let $A_0 = 0$, and A_i for $1 \leqslant i \leqslant s$ be the coefficient of Y^{i-1} in the polynomial $(Y - 1)(Y - \gamma) \cdots (Y - \gamma^{s-2})$. Then for $0 \leqslant \ell \leqslant s - 2$, we have

$$A_1 \cdot X^\ell + A_2 \cdot (\gamma X)^\ell + \cdots + A_s \cdot (\gamma^{s-1} X)^\ell = X^\ell \cdot \left(A_1 + A_2 \cdot \gamma^\ell + A_3 \cdot (\gamma^\ell)^2 + \cdots + A_s \cdot (\gamma^\ell)^{s-1}\right) = 0 \,.$$

By linearity, every polynomial $f \in \mathbb{F}_q[X]$ of degree at most $s - 2$ satisfies (11).

We should add that this does *not* lead to any non-trivial list-size lower bound for decoding, as we do not know if such bad polynomials can occur as the output of the interpolation step, and moreover the pruning step could potentially reduce the size of the list further.

Requirement on γ. The argument in Lemma 6 only required that the order of γ is at least k, and not that γ is primitive. In particular, Theorem 7 holds as long as the order of γ is at least k. The polynomial $X^{q-1} - \gamma$ is irreducible if and only if γ is primitive, and therefore the approach based on extension fields discussed in Section 2.1 requires γ to be primitive. Usually in constructions of Reed-Solomon codes, one takes the block length $n \approx q$ and therefore the dimension k is linear in q (for constant rate codes). So this weakened requirement on γ does not buy much flexibility in this case. However, in settings where one uses RS codes of small (say sub-constant) rate (which is often useful in complexity-theoretic applications of list decoding), the new argument applies to a broader set of choices of evaluation points for the RS codes.

Hensel lifting. An alternate approach (to root-finding over extension fields) for finding the low-degree solutions f to the equation $Q\big(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1}X)\big) = 0$ is based on Hensel-lifting. Here the idea is to solve for $f \mod X^i$ for $i = 1, 2, \ldots$ in turn. For example, the constant term f_0 of $f(X)$ must satisfy $Q(0, f_0, f_0, \ldots, f_0) = 0$. If $Q(0, Y, Y, \ldots, Y)$ is a nonzero polynomial, then this will restrict the number of choices for f_0. For each such choice f_0, solving $Q(X, f(X), \ldots, f(\gamma^{s-1}X)) \mod X^2 = 0$ gives a polynomial equation for f_1, and so on. This approach is discussed in [1] and [2, Chap. 5]. It is mentioned that this algorithm is very fast experimentally and almost never explores too many candidate solutions. A similar approach was also considered in [17] for folded versions of algebraic-geometric codes. However, theoretically it has not been possible to derive any polynomial guarantees on the size of the list returned by this approach or its running time (the obvious issue is that in each step there may be more than one candidate value of f_i, leading to an exponential product bound on the runtime). Polynomial bounds in special cases (eg. when $s = 2$) are presented in [2], and obtaining such theoretical bounds is posed as an interesting challenge for future work. Our Lemma 6 provides an analysis of the Hensel-lifting approach when the interpolated polynomial is linear in the Y_i's.

Multiplicities, soft decoding, and list recovery. For the linear interpolation of the form (4), using multiplicities in the interpolation stage, as in [14], only hurts the performance. This is because the degree of the Y_i's cannot be increased to meet the needs of the larger number of interpolation conditions. Thus in order to get a good decoder that can handle *soft* information on reliabilities of various symbols [14, 18], one has to resort to the method behind the original algorithm of Guruswami and Rudra [13]. A weaker form of soft decoding is the problem of *list recovery*, where for each position i of the code the input is a set S_i of up to ℓ possible values, and the goal is to find all codewords whose i'th symbol belongs to S_i for at least t values of i. Interpolation based decoding methods extend in a straightforward way to this setting. For instance, for the method of Section 2.2, we can interpolate a polynomial through at most $N \cdot \ell \cdot (m - s + 1)$ s-tuples, at most $\ell \cdot (m - s + 1)$ corresponding to the symbols in S_i for each of N codeword positions i. This gives an algorithm that works for agreement fraction $\tau = t/N$ satisfying

$$\tau > \frac{\ell}{s+1} + \frac{s}{s+1}\frac{mR}{m-s+1}\,.$$

Comparing this bound with (10), we see that the first term is factor ℓ larger. The key point now is that for any fixed ℓ, by picking $s \approx \ell/\varepsilon$ and $m \approx \ell/\varepsilon^2$, we can list recover with agreement fraction $\tau = R + \varepsilon$. Crucially, the agreement fraction required does *not* degrade with increasing ℓ (though the alphabet size of the code does). A list recovery guarantee of this form is very useful in list decoding concatenated codes, for example to construct binary codes list-decodable up to the Zyablov radius, or codes list-decodable up to radius $1 - R - \varepsilon$ over alphabets of fixed size independent of n; see [13, Sect. V].

3 Derivative codes

For a polynomial $f \in \mathbb{F}_q[X]$, we denote by f' its formal derivative, i.e. if $f(X) = f_0 + f_1 X + \ldots + f_\ell X^\ell$, then $f'(X) = \sum_{i=1}^{\ell} i f_i X^{i-1}$, where the coefficient i is $\underbrace{1 + \cdots + 1}_{i \text{ times}}$. We denote by $f^{(i)}$ the i'th formal derivative of f.

Definition 8 (m'th order derivative code)**.** *Let $0 \leqslant m \in \mathbb{Z}$. Let $a_1, \ldots, a_N \in \mathbb{F}_q$ be distinct, let $n = Nm$, and let the parameters satisfy $m \leqslant k < n \leqslant q$. Further assume that $\mathrm{char}(\mathbb{F}_q) > k$.*

The derivative code $\mathrm{Der}_q^{(m)}[n,k]$ over the alphabet $\mathbb{F}_q^m$ encodes a polynomial $f \in \mathbb{F}_q[X]$ of degree at most $k-1$ by

$$f \mapsto \left(\begin{bmatrix} f(a_1) \\ f'(a_1) \\ \vdots \\ f^{(m-1)}(a_1) \end{bmatrix}, \begin{bmatrix} f(a_2) \\ f'(a_2) \\ \vdots \\ f^{(m-1)}(a_2) \end{bmatrix}, \ldots, \begin{bmatrix} f(a_N) \\ f'(a_N) \\ \vdots \\ f^{(m-1)}(a_N) \end{bmatrix} \right). \tag{15}$$

Remark 9. This code has block length N and rate $R = k/n$. The minimum distance is $N - \lfloor \frac{k-1}{m} \rfloor \approx (1-R)N$.

Note that the case $m = 1$ is a Reed-Solomon code. These are also the univariate version of the *multiplicity codes* of [20], where they were analyzed in the context of local decoding.

3.1 List decoding derivative codes

Suppose we have received the corrupted version of a codeword from the derivative code $\mathrm{Der}_q^{(m)}[n,k]$ as a string $\mathbf{y} \in (\mathbb{F}_q^m)^N$, which we will, as in the folded Reed-Solomon case, consider as an $m \times N$ matrix over $\mathbb{F}_q$ (recall $N = n/m$):

$$\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1N} \\ y_{21} & y_{22} & \cdots & y_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mN} \end{pmatrix}. \tag{16}$$

The goal is to recover all polynomials f of degree at most $k-1$ whose derivative encoding (15) agrees with $\mathbf{y}$ in at least t columns. This corresponds to decoding from $N-t$ symbol errors for the derivative code $\mathrm{Der}_q^{(m)}[n,k]$. When $t > (N + k/m)/2$, the polynomial f, if it exists, is unique, and in this regime an efficient decoding algorithm was given in [20] by adapting the Welch-Berlekamp algorithm for Reed-Solomon codes [29, 6].

We adapt the algebraic list-decoding method of Theorem 7 to the derivative code setting. As in the folded Reed-Solomon setting, the algorithm is a higher-dimensional analog of the Welch-Berlekamp algorithm consisting of two steps — (i) interpolation of an algebraic condition (that must be obeyed by all candidate polynomials f), and (ii) retrieving the list of candidate solutions f (from the algebraic condition found by the interpolation step). For the same settings of parameters as in Section 2.2, we achieve the same decoding radius, but the runtime bound we show for the interpolation and retrieval steps is a larger polynomial.

Recently, a different list decoding algorithm for derivative codes was given in [19]. Similar to the relationship between the algorithms in Theorem 2 and Theorem 7, this algorithm uses multiplicities to achieve a higher decoding radius than our algorithm for a fixed setting of parameters, at the cost of a more complicated algorithm and analysis.

3.1.1 Interpolation

Let $\mathcal{W}$ denote the $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q[X, Y_1, \ldots, Y_m]$ consisting of polynomials that have total degree at most 1 in the Y_i's, i.e, $\mathcal{W}$ contains polynomials of the form $B_0(X)+B_1(X)Y_1+B_2(X)Y_2+\cdots+B_m(X)Y_m$ for some polynomials $B_i \in \mathbb{F}_q[X]$.

Let Δ be the $\mathbb{F}_q$-linear map on $\mathcal{W}$ defined as follows: For $p \in \mathbb{F}_q[X]$, and $1 \leqslant i \leqslant m$,

$$\Delta(p)(X, Y_1, \ldots, Y_m) = p'(X) \tag{17}$$

and

$$\Delta(pY_i)(X, Y_1, \ldots, Y_m) = p'(X)Y_i + p(X)Y_{i+1}. \tag{18}$$

where we take $Y_{m+1} = 0$ for definitiveness.

The following lemma shows why this map is useful to us.

Lemma 10. *Suppose $P(X, Y_1, \ldots, Y_m) \in \mathbb{F}_q[X, Y_1, \ldots, Y_m]$ has degree at most 1 in $Y_1, \ldots, Y_i$ for some $i < m$ and degree 0 in $Y_{i+1}, \ldots, Y_m$. Then*

1. *$\frac{d}{dx}P\big(X, f(X), f'(X), f^{(2)}(X), \ldots, f^{(m)}(X)\big) = (\Delta P)\big(X, f(X), f'(X), \ldots, f^{(m)}(X)\big)$, and*
2. *ΔP has degree at most 1 in $Y_1, \ldots, Y_{i+1}$ and degree 0 in $Y_{i+2}, \ldots, Y_m$.*

Proof. By linearity, for $p(X) \in \mathbb{F}_q[X]$, it suffices to check this for $p(X)$ and $p(X)Y_j$ for $j \leqslant i$.

We have $\Delta p = p'$, and $\Delta(pY_j) = p'Y_j + pY_{j+1}$, which both have degree at most 1 in $Y_1, \ldots, Y_{i+1}$ and degree 0 in $Y_{i+2}, \ldots, Y_m$. Moreover,

$$\Delta(pY_j)\big(X, f(X), f'(X), \ldots, f^{(m)}(X)\big) = p'f^{(j-1)} + pf^{(j)} = \frac{d}{dx}pf^{(j-1)},$$

as desired. □

Let s, $1 \leqslant s \leqslant m$, be an integer parameter in the decoding algorithm. The goal in the interpolation step is to interpolate a **nonzero** polynomial $Q \in \mathbb{F}_q[X, Y_1, Y_2, \ldots, Y_m]$ of the form

$$A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \cdots + A_s(X)Y_s \tag{19}$$

satisfying the following conditions for each i, $1 \leqslant i \leqslant N$:

$$Q(a_i, y_{1i}, \ldots, y_{si}) = 0 \text{ and } (\Delta^j Q)(a_i, y_{1i}, \ldots, y_{mi}) = 0 \quad (j = 1, \ldots, m-s), \tag{20}$$

where Δ^j denotes the j-fold composition of the map Δ.

Observation 11. *For each i, the conditions (20) are a collection of $(m - s + 1)$ homogeneous linear constraints on the coefficients of the polynomial Q.*

Lemma 10 gives us the following.

Corollary 12. *Suppose Q of the form (19) satisfies the conditions (20). If the received word (16) agrees with the encoding of f at location i, that is, $f^{(j)}(a_i) = y_{j+1,i}$ for $0 \leqslant j < m$, then the univariate polynomial $\Lambda(X) := Q(X, f(X), \ldots, f^{(s-1)}(X))$ satisfies $\Lambda(a_i) = 0$ as well as $\Lambda^{(k)}(a_i) = 0$ for $k = 1, \ldots, m - s$, where $\Lambda^{(k)}(X)$ is the k'th derivative of Λ.*

We next argue, similarly to Lemma 3 in the folded Reed-Solomon case, that a nonzero interpolation polynomial Q exists and can be found efficiently. In this case, we only claim cubic runtime for solving the linear system (of course we can also state a runtime of $O(n^\omega)$ using faster matrix multiplication).

Lemma 13. *Let*

$$D = \left\lfloor \frac{N(m-s+1)-k+1}{s+1} \right\rfloor. \tag{21}$$

Then a nonzero Q of the form (19) satisfying the interpolation conditions (20) with $\deg(A_0) \leqslant D+k-1$ and $\deg(A_j) \leqslant D$ for $1 \leqslant j \leqslant s$ exists and can be found in $O(n^3)$ field operations over $\mathbb{F}_q$ by solving a homogeneous linear system over $\mathbb{F}_q$ with at most n constraints and variables.

Proof. Under the stated degree restrictions, the number of monomials in Q is

$$(D+1)s + D + k = (D+1)(s+1) + k - 1 > N(m-s+1). \tag{22}$$

where the last inequality follows from the choice (21) of D. The number of homogeneous linear equations imposed on the coefficients of Q in order to meet the interpolation conditions (20) is $N \cdot (m-s+1)$. As this is less than the number of monomials in Q, the existence of a nonzero Q follows, and it can be found by solving a linear system over $\mathbb{F}_q$ with at most $Nm = n$ constraints and at most $N \cdot (m-s+1) + (s+1) < Nm = n$ variables. □

Suppose we have a polynomial $Q(X, Y_1, \ldots, Y_s)$ satisfying the interpolation conditions (20). The following lemma gives an identity satisfied by any f which has good agreement with the received word.

Lemma 14. *Let Q satisfy the conclusion of Lemma 13. If $f \in \mathbb{F}[X]$ is a polynomial of degree at most $k-1$ whose derivative encoding (15) agrees with the received word $\mathbf{y}$ in at least t columns for $t > \frac{D+k-1}{m-s+1}$, then*

$$Q\big(X, f(X), f'(X), \ldots, f^{(s-1)}(X)\big) = 0.$$

Proof. Let $\Lambda(X) = Q(X, f(X), \ldots, f^{(s-1)}(X))$. By Corollary 12, an agreement in column i means that $\Lambda(X)$ satisfies $\Lambda(a_i) = 0$ and that the kth derivative $\Lambda^{(k)}(a_i)$ is also zero for $k = 1, \ldots, m-s$. In particular, t column agreements yield at least $t \cdot (m-s+1)$ roots (counting multiplicities) for Λ.

The degree of Λ is at most $D+k-1$, as f and each of its derivatives has degree at most $k-1$. Then as Λ is univariate of degree at most $D+k-1$, Λ has at most $D+k-1$ roots if it is nonzero. Thus if $t > (D+k-1)/(m-s+1)$, it must be that $\Lambda(X) = 0$. □

3.1.2 Retrieving candidate polynomials

With our chosen value of D from (21), the preceding section implies that any f which agrees with $\mathbf{y}$ on at least

$$\frac{N}{s+1} + \frac{s}{s+1}\frac{k}{m-s+1} \tag{23}$$

columns satisfies $Q\big(X, f(X), f'(X), \ldots, f^{(s-1)}(X)\big) = 0$. So in the second step, our goal is to find all polynomials f of degree at most $k-1$ such that

$$A_0(X) + A_1(X)f(X) + A_2(X)f'(X) + \ldots + A_s(X)f^{(s-1)}(X) = 0. \tag{24}$$

Let $A_i(X) = \sum_{j=0}^{\deg(A_i)} a_{i,j}X^j$ for each i. Note that the above constraint (24) gives a *linear system* over $\mathbb{F}$ in the coefficients of $f = f_0 + f_1X + \cdots + f_{k-1}X^{k-1}$. In particular, the set of solutions $(f_0, f_1, \ldots, f_{k-1})$ is an affine space, and we can find it by solving the linear system. Our goal now is to bound the dimension of the space of solutions by exposing its special structure and also use this to efficiently find an explicit basis for the space.

Lemma 15. *It suffices to give an algorithm in the case that the constant term a_{s0} of A_s is nonzero.*

Proof. If $A_s(X) \not\equiv 0$, since $\deg(A_s) \leqslant D < Nm \leqslant q$, then there is some $\alpha \in \mathbb{F}_q$ such that $A_s(\alpha) \neq 0$, so we can consider a "translate" of this problem by α; that is, $A_s(X+\alpha)$ has nonzero constant term, so we can solve the system with the translated polynomial $Q(X+\alpha, Y_1, \ldots, Y_m)$ and recover candidate messages by translating each solution $g(X)$ to $f(X) = g(X-\alpha)$.

If $A_s(X) = 0$, we simply reduce the problem to a smaller one with s rather than $s+1$ interpolation variables. Note that this must terminate since Q is nonzero and so at least one A_i for $i \geqslant 1$ is nonzero. □

We can now show the following analog of Lemma 6:

Lemma 16. *If $a_{s,0} \neq 0$, the solution space to (24) has dimension at most $s-1$. Furthermore, a basis for this subspace can be found in $O(sk^2)$ operations over $\mathbb{F}_q$.*

Proof. For each power X^i, the coefficient of X^i in $A_0(X) + A_1(X)f(X) + \cdots + A_s(X)f^{(s-1)}(X)$ is

$$\begin{aligned}
&a_{0,i} + \big(a_{1,0}f_i + a_{1,1}f_{i-1} + \cdots + a_{1,i}f_0\big) + \big(a_{2,0}(i+1)f_{i+1} + a_{2,1}if_i + \cdots + a_{2,i}f_1\big) \\
&\quad + \cdots + \big(a_{s,0}(i+s-1)(i+s-2)\cdots(i+1)f_{i+s-1} + \cdots + a_{s,i}(s-1)!f_{s-1}\big) \\
=&a_{0,i} + \sum_{j=1}^{s}\sum_{\ell=0}^{i} \frac{(\ell+j-1)!}{\ell!} a_{j,i-\ell}f_{\ell+j-1}\,.
\end{aligned} \tag{25}$$

If $(f_0, \ldots, f_{k-1})$ is a solution to (24), then this coefficient is zero for every i.

Our expression for the coefficient of X^i for each i depends only on f_j for $j < i+s$. Furthermore, the coefficient of f_{i+s-1} in this expression is $a_{s,0} \cdot (i+s-1)(i+s-2)\cdots(i+1)$, which is nonzero when $i+s \leqslant k$ since $\mathrm{char}(\mathbb{F}_q) > k$. Thus, if we fix $f_0, f_1, \ldots, f_{s-2}$, the rest of the message symbols $f_{s-1}, \ldots, f_{k-1}$ are uniquely determined. In particular, the dimension of the solution space is at most $s-1$. Also, by (25), each f_l, $l \geqslant s-1$, is specified as a linear combination of f_i for $i < l$, and implies that we can compute a basis of the solution space $(f_0, \ldots, f_{k-1})$ using $O(sk^2)$ field operations. □

Combining Lemmas 14 and 16, and recalling the bound (23) on the number of agreements for successful decoding, we have our result on list decoding derivative codes.

Theorem 17. *For the derivative code $\mathrm{Der}_q^{(m)}[n,k]$ (where $\mathrm{char}(\mathbb{F}_q) > k$) of block length $N = n/m$ and rate $R = k/n$, the following holds for all integers s, $1 \leqslant s \leqslant m$. Given a received word $\mathbf{y} \in \mathbb{F}_q^{m \times N}$, in $O\big(n^3 + sk^2\big)$ operations over $\mathbb{F}_q$, one can find a basis for a subspace of dimension at most $s-1$ that contains all message polynomials $f \in \mathbb{F}_q[X]$ of degree less than k whose derivative encoding (15)differs from $\mathbf{y}$ in at most a fraction*

$$\frac{s}{s+1}\left(N - \frac{k}{(m-s+1)}\right)$$

of the N codeword positions.

Now by setting $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$, and recalling that the rate of $\mathrm{Der}_q^{(m)}[n,k]$ is $k/n = k/(Nm)$, we can conclude the following.

Corollary 18. *For all $R \in (0,1)$ and all $\varepsilon > 0$, for a suitable choice of parameters, there are derivative codes $\mathrm{Der}_q^{(m)}[n,k]$ of rate at least R which can be list decoded from a fraction $1 - R - \varepsilon$ of errors with a list-size of $q^{O(1/\varepsilon)}$.*

3.2 Some remarks

Tightness of q^{s-1} bound. As in the folded Reed-Solomon case, the bound of Lemma 16 is tight for arbitrary linear systems. Indeed, if

$$Q(X, Y_1, \ldots, Y_s) = \sum_{i=0}^{s-1} \frac{(-1)^i}{i!} X^i Y_{i+1},$$

then *any* polynomial $f(X)$ of degree less than s with zero constant term satisfies the identity $Q(X, f(X), \ldots, f^{(s-1)}(X)) = 0$. This is because any monomial $f(X) = X^j$ for $0 < j \leqslant s-1$ is a solution, and our solution space is linear. Again as in the FRS case, we do not know if such a bad polynomial can occur as the output of the interpolation step when decoding a noisy codeword of the derivative code.

Decoding derivative codes with side information. The decoding described in the previous section consists of trying all choices for the coefficients $f_0, \ldots, f_{s-2}$ and using each to uniquely determine a candidate for f. Note however that for each i, the f_i is essentially the ith derivative of f evaluated at 0, and can be recovered as $f^{(i)}(0)/i!$. Thus if the decoder somehow knew the correct values of f and its first $s-1$ derivatives at 0, f could be recovered uniquely (as long as $A_s(0) \neq 0$).

Now, suppose the encoder could send a small amount of information along a noiseless side channel in addition to sending the (much longer) codeword on the original channel. In such a case, the encoder could choose $\alpha \in \mathbb{F}_q$ uniformly at random and transmit $f(\alpha), f'(\alpha), \ldots, f^{(s-1)}(\alpha)$ on the noiseless channel. The decoding then fails only if $A_i(\alpha) = 0$ for i which is the largest index such that $A_i(X) \neq 0$. As the $A_i(X)$ have bounded degree, by increasing the field size q, f can be uniquely recovered with probability arbitrarily close to 1. More precisely, we have the following claim.

Theorem 19. *Given a uniformly random $\alpha \in \mathbb{F}_q$ and the values $f(\alpha), f'(\alpha), \ldots, f^{(s-1)}(\alpha)$ of the message polynomial f, the derivative code $\mathrm{Der}_q^{(m)}[n,k]$ can be uniquely decoded from up to*

$$\frac{s}{s+1}\left(N - \frac{k}{m-s+1}\right)$$

errors with probability at least $1 - n/(sq)$ over the choice of α.

Proof. As in the proof of Lemma 15, as long as $A_s(\alpha) \neq 0$, we may translate the problem by α and use the values $f(\alpha), f'(\alpha), \ldots, f^{(s-1)}(\alpha)$ to uniquely determine the shifted coefficients $g_0, \ldots, g_{s-1}$.

As $A_s \neq 0$, and A_s is univariate of degree at most D, A_s has at most D roots, and so the probability that $A_s(\alpha) \neq 0$ is at least $1 - D/q \geqslant 1 - \frac{n}{sq}$, where the last inequality follows from our choice of $D \leqslant n/s$ in (21). □

Remark 20. In the context of communicating with side information, there is a generic, black-box solution combining list-decodable codes with hashing to guarantee unique recovery of the correct message with high probability [7]. In such a scheme, the side information consists of a random hash function h and its value $h(f)$ on the message f. The advantage of the solution in Theorem 19 is that there is *no need* to compute the full list (which is the computationally expensive step, since the list size bound depends exponentially on s) and then prune it to the unique solution. Rather, we can uniquely identify the first $(s-1)$ coefficients of the polynomial $f(X+\alpha)$ in the linear system (24), after applying the shift $X \mapsto X + \alpha$, as $f(\alpha), f'(\alpha), \ldots, f^{(s-2)}(\alpha)$. Then, as argued in the proof of Lemma 16, the remaining coefficients are determined as linear combinations of these $s-1$ coefficients. So the whole algorithm can be implemented in cubic time.

Note that we do not know how to apply the approach of Theorem 19 to the case of folded Reed-Solomon codes. The key difference is that in the derivative code case, it is known that the decoder will need the first $s-1$ message coefficients. In the folded Reed-Solomon case, the required coefficients could depend on the interpolated polynomial Q, which would mean that the correct values could not be sent ahead of time.

Remark 21. The decoder could use the columns of the received word $\mathbf{y}$ as a guess for the side information $f(a_i), f'(a_i), \ldots, f^{(s-2)}(a_i)$ for $i = 1, 2, \ldots, N$. Since f agrees with $\mathbf{y}$ on more than $t > RN$ positions, as long as $A_s(a_i) = 0$ for less than t of the evaluation points a_i, we will recover every solution f this way. This would lead to a list size bound of at most $N - t < N$. Unfortunately, however, there seems to be no way to ensure that A_s does not vanish at most (or even all) of the points a_i used for encoding. But perhaps some additional ideas can be used to make the list size polynomial in both q, s, or at least $\exp\big(O(s)\big)q^c$ for some absolute constant c.

4 Improving list size via subspace-evasive subsets

Based on Theorems 7 and 17, in this section we pursue one possible approach to improve the provable worst-case list size bound for list decoding up to a fraction $1 - R - \varepsilon$ of errors. Instead of allowing all polynomials $f_0 + f_1 X + \cdots + f_{k-1}X^{k-1}$ of degree less than k as messages, the idea is to restrict the coefficient vector $(f_0, f_1, \ldots, f_{k-1})$ to belong to some special subset $\mathcal{V} \subseteq \mathbb{F}_q^k$, satisfying the following two conflicting demands:

Largeness: The set $\mathcal{V}$ must be large, say $|\mathcal{V}| \geqslant q^{(1-\varepsilon)k}$, so that the rate is reduced by at most a $(1-\varepsilon)$ factor.

Low intersection with subspaces: For every subspace $S \subset \mathbb{F}_q^k$ of dimension s, $|S \cap \mathcal{V}| \leqslant L$. (Let us call this property of $\mathcal{V}$ as (s, L)-*subspace-evasive* for easy reference. The field $\mathbb{F}_q$ and the ambient dimension k will be fixed in our discussion.)

Using such a set $\mathcal{V}$ will ensure that after pruning an affine subspace output by the algorithms of Theorem 7 and 17, the number of codewords will be at most L. (Note that an affine subspace of dimension $s-1$ is contained in a subspace of dimension s.) Thus the list size will go down from q^{s-1} to L.

Subspace-evasive subsets were used in [23] to construct bipartite Ramsey graphs, and in fact we borrowed the term *evasive* from that work. In their work, the underlying field was $\mathbb{F}_2$ and the subsets had to be evasive for dimension $s \approx k/2$. Our interest is in a different (and hopefully easier?) regime — we can work over large fields, and are interested in evasiveness with respect to s-dimensional subspaces for constant s.

A random large subset of $\mathbb{F}_q^k$ meets the low subspace intersection requirement very well, as shown below. The argument is straightforward; a similar bound appears in [3] in the geometric context of point-subspace incidences.

Lemma 22. *Let k be a large enough positive integer, and let $s < k/4$ be a positive integer. For some α with $0 < \alpha < k/4$, let $\mathcal{W}$ be a random subset of $\mathbb{F}_q^k$ chosen by including each $x \in \mathbb{F}_q^k$ in $\mathcal{W}$ with probability $q^{-s-\alpha}$. Then with probability at least $1 - q^{-k}$, $\mathcal{W}$ satisfies both the following conditions: (i) $|\mathcal{W}| \geqslant q^{k-s-\alpha}/2$, and (ii) $\mathcal{W}$ is $(s, 2sk/\alpha)$-subspace-evasive.*

Proof. The first part follows by a standard Chernoff bound calculation: the expected value of $|\mathcal{W}|$ equals $q^{k-s-\alpha}$, and thus the probability that it is less than half the expected value is at most $\exp(-q^{k-s-\alpha}/8) \leqslant q^{-k}$.

For the second part, fix a subspace $S \subseteq \mathbb{F}_q^k$ of dimension s, and a subset $T \subseteq S$ of size $t = \lceil 2ks/\alpha \rceil$. The probability that $\mathcal{W} \supseteq T$ equals $q^{-(s+\alpha)t}$. By a union bound over the at most q^{ks} choices for the s-dimensional subspace S, and the at most q^{st} choices of t-element subsets T of S, we get that the probability that $\mathcal{W}$ is not $(s, t-1)$-subspace-evasive is at most $q^{ks+st} \cdot q^{-(s+\alpha)t} \leqslant q^{-ks}$ since $t \geqslant 2ks/\alpha$. □

Picking $\alpha \approx \varepsilon k$, the above guarantees the existence of subsets $\mathcal{W}$ of $\mathbb{F}_q^k$ of size $q^{(1-\varepsilon)k-s-1}$ which are $\big(s, O(s/\varepsilon)\big)$-subspace-evasive. Restricting the coefficient vector $(f_0, f_1, \ldots, f_{k-1})$ of the message polynomial to belong to such a subset will guarantee a list-size upper bound of $O(s/\varepsilon)$ in Theorem 7 or Theorem 17. This list-size bound is independent of q, and for the choice $s \approx 1/\varepsilon$ which enables list decoding a fraction $1 - R - \varepsilon$ of errors, it is $O(1/\varepsilon^2)$. This is quite close to the bound of $O(1/\varepsilon)$ achieved by random codes [10].

Unfortunately, an explicit construction of subspace-evasive subsets approaching the trade-off guaranteed by the probabilistic construction of Lemma 22 is not known. This appears to be a challenging and extremely interesting question. One natural choice for such a subset would be some *variety* $\mathcal{V} \subseteq \mathbb{F}_q^k$ defined by a collection of polynomial equations, i.e., $\mathcal{V} = \{\mathbf{a} \in \mathbb{F}_q^k \mid g_1(\mathbf{a}) = g_2(\mathbf{a}) = \cdots = g_l(\mathbf{a}) = 0\}$ for some polynomials $g_1, g_2, \ldots, g_l \in \mathbb{F}_q[Z_1, Z_2, \ldots, Z_k]$. Indeed for $s = 1$

and $s = k-1$, varieties in $\mathbb{F}_q^k$ (the modular moment surface and modular moment curve) with low intersection with s-dimensional affine subspaces are known [3].

4.1 Pseudorandom construction of subspace-evasive subsets

The construction of Lemma 22 takes exponential time and produces a random unstructured set that takes exponential space to store. In this section, we show that a subset with similar guarantees can be constructed in probabilistic polynomial time, producing a polynomial size representation of the constructed subspace-evasive set. The idea is to note that the probabilistic argument to argue about (s,t)-subspace-evasiveness only needed t-wise independence and not complete independence of different elements of $\mathbb{F}_q^k$ landing in the random subset $\mathcal{W}$. We now describe such a pseudorandom construction.

For some parameter $\zeta \in (0, 1/2)$, let $k' = (1-\zeta)k$. Let $\mathbb{K}$ be the extension field $\mathbb{F}_{q^{k'}}$, and fix an arbitrary basis B of $\mathbb{K}$ over $\mathbb{F}_q$.

We will define a subspace-evasive embedding of $\mathbb{F}_q^{k'}$ into $\mathbb{F}_q^k$ as follows. Because we have fixed the basis B, we can consider any $v \in \mathbb{F}_q^{k'}$ as an element of $\mathbb{K} = \mathbb{F}_{q^{k'}}$. Let $P \in \mathbb{K}[X]$ be a polynomial of degree at most t, and let $Q(v)$ be the first ζk coordinates of $P(v)$ with respect to the basis B.

Then we will map $v \in \mathbb{F}_q^{k'}$ to $\big(v, Q(v)\big) \in \mathbb{F}_q^k$. Let $\mathcal{W}_{\zeta,k}(P)$ be the image of $\mathbb{F}_q^{k'}$ under this map. As the map is injective, $|\mathcal{W}_{\zeta,k}(P)| = q^{k'} = q^{(1-\zeta)k}$.

Lemma 23. *Let $k \geqslant 1$ be an integer. Let $\zeta \in (0, 1/2)$, and let s be an integer satisfying $1 \leqslant s \leqslant \zeta k/2$. Let $t \geqslant \lceil 4s/\zeta \rceil$ be a positive integer and $P \in \mathbb{K}[X]$ be a random polynomial of degree at most t.*

Define $\mathcal{W} = \mathcal{W}_{\zeta,k}(P)$. Then with probability at least $1 - q^{-ks}$ over the choice of P, $\mathcal{W}$ is a $(s, 4s/\zeta)$-subspace-evasive subset of $\mathbb{F}_q^k$ of size $q^{(1-\zeta)k}$.

Proof. We already argued that $|\mathcal{W}| = q^{(1-\zeta)k}$. For each $\mathbf{x} \in \mathbb{F}_q^k$, the probability that $\mathbf{x}$ is in $\mathcal{W}$ is the probability that the last ζk coordinates are Q evaluated at the first $(1-\zeta)k$ coordinates. As P was random, this is $q^{-\zeta k}$.

Since the values of P at any t distinct points in $\mathbb{K}$ are independent, the events $\mathbf{x} \in \mathcal{W}$ are t-wise independent as long as no two share the same initial k' coordinates.

Now fix a subspace $S \subseteq \mathbb{F}_q^k$ of dimension s, and a subset $T \subseteq S$ of size t. Let us compute the probability that $T \subseteq \mathcal{W}$. As $\mathcal{W}$ contains exactly one element for each setting of the initial k' coordinates, we may assume no two elements in T share the same initial k' coordinates. The t events $\beta \in \mathcal{W}$ for various $\beta \in T$ are independent due to the above t-wise independence property. Thus the probability that $T \subseteq \mathcal{W}$ equals $q^{-\zeta kt}$. The remaining calculation is as in Lemma 22 and involves a union bound over the at most q^{ks} choices for the s-dimensional subspace S, and the at most q^{st} choices of t-element subsets T of S. □

Note that the set $\mathcal{W}$ has a compact representation, and given P of degree $t \leqslant O(s/\zeta)$, the bijection from $\mathbb{F}_{q^{k'}}$ to $\mathcal{W}$ can be computed using $\mathrm{poly}(k, s, 1/\zeta)$ $\mathbb{F}_q$-operations, and membership in $\mathcal{W}$ can be checked in the same time. This implies that we can efficiently encode any $v \in \mathbb{F}_q^{k'}$ by computing its representative in $\mathcal{W}$ and then applying either the folded Reed-Solomon or derivative encoding. Combining this with Theorems 7 and 17, we can conclude the following final result.

Theorem 24. *For any ζ, $0 < \zeta < 1/2$, there is a Monte Carlo construction of a subcode C of $\mathrm{FRS}_q^{(m)}[n,k]$ or $\mathrm{Der}_q^{(m)}[n,k]$ of rate $(1-\zeta)R$ where $R = k/n$, consisting of encodings of polynomials whose coefficients belong to a subspace-evasive subset $\mathcal{W} \subset \mathbb{F}_q^k$, such that*

(i) there is a efficient encoder computing a bijection $\mathbb{F}_q^{(1-\zeta)k} \to C$ using $\mathrm{poly}(n, m, 1/\zeta)$ $\mathbb{F}_q$-operations, and

(ii) with high probability C can be list decoded from error fraction $\frac{s}{s+1}\left(1 - \frac{mR}{m-s+1}\right)$ for any $1 \leqslant s \leqslant m$ in $q^{O(s)}$ time with an output list size of at most $O(s/\zeta)$.

In particular, picking $\zeta = \Theta(\varepsilon)$, $s = \Theta(1/\varepsilon)$ and $m = \Theta(1/\varepsilon^2)$, for any desired $R' \in (0,1)$, the construction yields codes of rate R' which can be list decoded from a fraction $1 - R' - \varepsilon$ of errors in $q^{O(1/\varepsilon)}$ time, with at most $O(1/\varepsilon^2)$ codewords output in the list.

5 Subsequent work

We conclude the paper by mentioning two follow-up works improving upon certain results of this paper. Besides these, the authors and Narayanan also used the linear-algebraic approach to list decode *subspace codes* based on linearized polynomials [11].

Explicit constructions of subspace-evasive sets. Following this work, Dvir and Lovett [5] gave an *explicit* construction of a set $S \subseteq \mathbb{F}_q^n$ of size at least $q^{(1-\varepsilon)n}$ which is $\left(s, (s/\varepsilon)^s\right)$-subspace-evasive. Their construction uses so-called *everywhere-finite varieties*, and has the nice property that the intersection of S with any s-dimensional subspace can be computed in time which is polynomial in the size of the intersection. This avoids the q^s time search in the pruning step of the decoding.

However, it is not known how to improve the $(s/\varepsilon)^s$ bound to match the existential $O(s/\varepsilon)$ bound of Lemma 23.

Folded algebraic-geometric codes. Guruswami and Xing [16] extended the linear-algebraic approach to list decode folded versions of certain algebraic-geometric codes. They made use of certain automorphisms of the function field in combination with the power series expansion of functions around a special place for the decoding. An extension of subspace-evasive sets called *hierarchical* subspace-evasive sets was used to prune the subspace of candidate messages. Instantiated with one of the optimal function field towers due to Garcia and Stichtenoth, this enabled list decoding up to a fraction $(1 - R - \varepsilon)$ of errors with a list size of $O(1/\varepsilon)$ over an alphabet of size $(1/\varepsilon)^{O(1/\varepsilon^2)}$, almost matching the random coding bound in all aspects simultaneously.

One of the tricks used in [16] to bring down the list size to $O(1/\varepsilon)$, which is based on taking the intersection of two independent pseudorandom subspace-evasive sets, can also be applied to our construction. This would improve the list size bound in Theorem 24 to $O(1/\varepsilon)$ (instead of $O(1/\varepsilon^2)$), though the drawback of the large $q^{O(1/\varepsilon)}$ decoding complexity would remain.

Acknowledgments

The first author is grateful to Salil Vadhan for telling him about the degree 1 interpolation method for list decoding folded RS codes. We thank Noga Alon, Swastik Kopparty, Po-Shen Loh, and David Zuckerman for their input on the literature on subspace-evasive sets. We thank Ran Raz for discussions about explicit constructions of subspace-evasive varieties. Thanks to Swastik Kopparty for pointing us to [23], and to Po-Shen Loh for pointers to work on related concepts in geometric settings [4]. Thanks to Atri Rudra and the anonymous referees for their careful reading and several valuable comments on the write-up. In particular, a reviewer comment about the earlier construction of pseudorandom subspace-evasive sets got us thinking towards obtaining the current (simpler) construction in Section 4.1.

References

[1] P. Beelen and K. Brander. Decoding Folded Reed-Solomon codes using Hensel-lifting. In *Gröbner Bases, Coding, and Cryptography*, pages 389–394. Springer-Verlag, 2009. 12

[2] K. Brander. *Interpolation and list decoding of algebraic codes*. PhD thesis, Technical University of Denmark, 2010. 8, 12

[3] P. Braß and C. Knauer. On counting point-hyperplane incidences. *Comput. Geom.*, 25(1-2):13–20, 2003. 19, 20

[4] P. Brass, W. Moser, and J. Pach. *Research Problems in Discrete Geometry*. Springer, New York, 2005. 22

[5] Z. Dvir and S. Lovett. Subspace evasive sets. In *Proceedings of the 44th ACM Symposium on Theory of Computing*, pages 351–358, 2012. 21

[6] P. Gemmell and M. Sudan. Highly resilient correctors for multivariate polynomials. *Information Processing Letters*, 43(4):169–174, 1992. 7, 13

[7] V. Guruswami. List decoding with side information. In *Proceedings of the 18th IEEE Conference on Computational Complexity (CCC)*, pages 300–309, 2003. 11, 18

[8] V. Guruswami. List decoding Folded Reed-Solomon codes, 2010. Lecture notes, available at `http://www.cs.cmu.edu/~venkatg/teaching/codingtheory/notes/notes11.pdf`. 7

[9] V. Guruswami. Linear-algebraic list decoding of folded Reed-Solomon codes. In *Proceedings of the 26th IEEE Conference on Computational Complexity*, pages 77–85, June 2011. 1

[10] V. Guruswami, J. Håstad, M. Sudan, and D. Zuckerman. Combinatorial bounds for list decoding. *IEEE Transactions on Information Theory*, 48(5):1021–1035, 2002. 19

[11] V. Guruswami, S. Narayanan, and C. Wang. List decoding subspace codes from insertions and deletions. In *Proceedings of Innovations in Theoretical Computer Science (ITCS 2012)*, pages 183–189, January 2012. 21

[12] V. Guruswami and A. Patthak. Correlated Algebraic-Geometric codes: Improved list decoding over bounded alphabets. *Mathematics of Computation*, 77(261):447–473, 2008. 7

[13] V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008. 2, 3, 4, 6, 12

[14] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and Algebraic-geometric codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999. 2, 3, 9, 12

[15] V. Guruswami and C. Wang. Optimal rate list decoding via derivative codes. In *Proceedings of APPROX/RANDOM 2011*, pages 593–604, August 2011. 1

[16] V. Guruswami and C. Xing. Folded codes from function field towers and improved optimal rate list decoding. In *Proceedings of the 44th ACM Symposium on Theory of Computing*, pages 339–350, 2012. 21

[17] M. D. Huang and A. K. Narayanan. Folded algebraic geometric codes from Galois extensions. *ArXiv CoRR*, `http://arxiv.org/abs/0901.1162`, 2009. 12

[18] R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 49(11):2809–2825, August 2003. 3, 12

[19] S. Kopparty. List-decoding multiplicity codes. *Electronic Colloquium on Computational Complexity, TR12-044*, 2012. 13

[20] S. Kopparty, S. Saraf, and S. Yekhanin. High-rate codes with sublinear-time decoding. *Electronic Colloquium on Computational Complexity, TR10-148*, 2010. 4, 13

[21] F. Parvaresh and A. Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 285–294, 2005. 2, 3, 4

[22] W. W. Peterson. Encoding and error-correction procedures for Bose-Chaudhuri codes. *IEEE Transactions on Information Theory*, 6:459–470, 1960. 2

[23] P. Pudlák and V. Rödl. Pseudorandom sets and explicit constructions of Ramsey graphs. In *Complexity of Computations and Proofs. Quad. Mat., 13, Dept. Math., Seconda Univ. Napoli, Caserta*, pages 327–346, 2004. 5, 19, 22

[24] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997. 2

[25] M. Sudan. List decoding: Algorithms and applications. *SIGACT News*, 31:16–27, 2000. 2

[26] L. Trevisan. Some applications of coding theory in computational complexity. *Quaderni di Matematica*, 13:347–424, 2004. 2

[27] S. Vadhan. *Pseudorandomness*. Foundations and Trends in Theoretical Computer Science (FnT-TCS). NOW publishers, 2010. To appear. Draft available at `http://people.seas.harvard.edu/~ salil/pseudorandomness/`. 3, 4, 7

[28] S. P. Vadhan. The unified theory of pseudorandomness. In *Proceedings of the International Congress of Mathematicians*, 2010. 2

[29] L. R. Welch and E. R. Berlekamp. Error correction of algebraic block codes. *US Patent Number 4,633,470*, December 1986. 7, 13

ECCC ISSN 1433-8092

http://eccc.hpi-web.de