Electronic Colloquium on Computational Complexity, Report No. 179 (2020)

Decoding Multivariate Multiplicity Codes on Product Sets

Siddharth Bhandari* Prahladh Harsha* Mrinal Kumar† Madhu Sudan‡

Abstract

The multiplicity Schwartz-Zippel lemma bounds the total multiplicity of zeroes of a multivariate polynomial on a product set. This lemma motivates the multiplicity codes of Kopparty, Saraf and Yekhanin [J. ACM, 2014], who showed how to use this lemma to construct high-rate locally-decodable codes. However, the algorithmic results about these codes crucially rely on the fact that the polynomials are evaluated on a vector space and not an arbitrary product set.

In this work, we show how to decode multivariate multiplicity codes of large multiplicities in polynomial time over finite product sets (over fields of large characteristic and zero characteristic). Previously such decoding algorithms were not known even for a positive fraction of errors. In contrast, our work goes all the way to the distance of the code and in particular exceeds both the unique decoding bound and the Johnson bound. For errors exceeding the Johnson bound, even combinatorial list-decodablity of these codes was not known.

Our algorithm is an application of the classical polynomial method directly to the multivariate setting. In particular, we do not rely on a reduction from the multivariate to the univariate case as is typical of many of the existing results on decoding codes based on multivariate polynomials. However, a vanilla application of the polynomial method in the multivariate setting does not yield a polynomial upper bound on the list size. We obtain a polynomial bound on the list size by taking an alternative view of multivariate multiplicity codes. In this view, we glue all the partial derivatives of the same order together using a fresh set $\mathbf{z}$ of variables. We then apply the polynomial method by viewing this as a problem over the field $\mathbb{F}(\mathbf{z})$ of rational functions in $\mathbf{z}$.

*Tata Institute of Fundamental Research, Mumbai, India. {siddharth.bhandari,prahladh}@tifr.res.in. Research supported by the Department of Atomic Energy, Government of India, under project no. 12-R&D-TFR-5.01-0500 and in part by the Google PhD Fellowship and Swarnajayanti fellowship.

†Department of Computer Science & Engineering, IIT Bombay, Mumbai, India. mrinal@cse.iitb.ac.in.

‡School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. madhu@cs.harvard.edu. Supported in part by a Simons Investigator Award and NSF Award CCF 1715187.

ISSN 1433-8092

1 Introduction

The classical Schwartz-Zippel Lemma (due to Ore [Ore22], Schwartz [Sch80], Zippel [Zip79] and DeMillo & Lipton [DL78]) states that if $\mathbb{F}$ is a field, and $f \in \mathbb{F}[x_1, x_2, \ldots, x_k]$ is a *non-zero* polynomial of degree d, and $S \subseteq \mathbb{F}$ is an arbitrary finite subset of $\mathbb{F}$, then the number of points on the grid [1] S^k where f is zero is upper bounded by $d|S|^{k-1}$. A higher order *multiplicity* version of this lemma (due to Dvir, Kopparty, Saraf and Sudan [DKSS13]) states the number of points on the grid S^k where f is zero with *multiplicity*[2] at least s is upper bounded by $\frac{d|S|^{k-1}}{s}$.[3]

This innately basic statement about low degree polynomials has had innumerable applications in both theoretical computer science and discrete mathematics and has by now become a part of the standard toolkit when working with low degree polynomials [Sar11, Gut16]. Despite this, the following natural algorithmic version of this problem remains open.

Algorithmic SZ question. *Let $\mathbb{F}$ be a field, and S, d, k be as above. Design an efficient algorithm that takes as input an arbitrary function $P : S^k \to \mathbb{F}^{\binom{s+k-1}{k}}$ and finds a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_k]$ of degree at most d (if one exists) such that the function $\mathsf{Enc}(f) : S^k \to \mathbb{F}^{\binom{s+k-1}{k}}$ defined as*

$$\mathsf{Enc}(f)(\mathbf{a}) = \left(\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) : \deg(\mathbf{x}^{\mathbf{e}}) < s \right)$$

differs from P on less than $\frac{1}{2}\left(1 - \frac{d}{s|S|}\right)$ fraction of points on S^k.

The aforementioned multiplicity Schwartz-Zippel lemma (henceforth, referred to as the multiplicity SZ lemma for brevity) assures us that if there is a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_k]$ such that $\mathsf{Enc}(f)$ differs from P on less than $\frac{1}{2}\left(1 - \frac{d}{s|S|}\right)$ fraction of points, then it must be unique! Thus, in some sense, the above question is essentially asking for an algorithmic version of the multiplicity SZ lemma.

Although a seemingly natural problem, especially given the ubiquitous presence of the SZ lemma in computer science, this question continues to remain open for even bivariate polynomials! In fact, even the $s = 1$ case, which corresponds to an algorithmic version of the classical SZ lemma (without multiplicities) was only very recently resolved in a beautiful work of Kim and Kopparty [KK17]. Unfortunately, their algorithm does not seem to extend to the case of $s > 1$, and they mention this as one of the open problems.

In this work, we make some progress towards answering the algorithmic SZ question. In particular, we design an efficient deterministic algorithm for this problem when the field $\mathbb{F}$ has characteristic zero or larger than the degree d, the dimension k is an arbitrary constant and the

[1] We use "grids" and "product sets" interchangeably (see also Remark 1.2).

[2] This means that all the partial derivatives of f of order at most $s-1$ are zero at this point. See Section 3 for a formal definition.

[3] This bound is only interesting when $|S| > d/s$ so that $\frac{d|S|^{k-1}}{s}$ is less than the trivial bound of $|S|^k$.

multiplicity parameter s is a sufficiently large constant. In fact, in this setting we prove a stronger result, which we now informally state (see Theorem 1.1 for a formal statement).

Main result. *Let $\varepsilon \in (0,1)$ be an arbitrary constant, $k \in \mathbb{N}$ be a positive constant and s be a large enough positive integer. Over fields $\mathbb{F}$ of characteristic zero or characteristic larger than d, there is a deterministic polynomial algorithm that on input P outputs all degree d polynomials $f \in \mathbb{F}[x_1, x_2, \ldots, x_k]$ such that $\mathsf{Enc}(f)$ differs from the input function $P : S^k \to \mathbb{F}^{\binom{s+k-1}{k}}$ on less than $\left(1 - \frac{d}{s|S|} - \varepsilon\right)$ fraction of points on the grid S^k.*

We note that the fraction of errors that can be tolerated in the above result is $1 - \frac{d}{s|S|} - \varepsilon$, which is significantly larger than the error parameter in the algorithmic SZ question. Therefore, we no longer have the guarantee of a unique solution f such that the function $\mathsf{Enc}(f)$ which is *close* to P. In fact, for this error regime, it is not even clear that the number of candidate solutions is polynomially bounded. The algorithm stated in the main result outputs all such candidate solutions, and in particular, shows that their number is polynomially bounded (for constant k). This fraction of errors is the best one can hope for since there are functions P (for instance, the all zero's function) which have super-polynomially many polynomials of degree d which are $\left(1 - \frac{d}{s|S|}\right)$-close to P. (see Appendix A).

In the language of error correcting codes, the algorithmic SZ question is the question of designing efficient unique decoding algorithms for multivariate multiplicity codes over arbitrary product sets when the error is at most half the minimum distance, and main result gives an efficient algorithm for the possibly harder problem of list decoding these codes from relative error $\delta - \varepsilon$, where $\delta := 1 - \frac{d}{s|S|}$ is the distance of the code, provided that the field has characteristic larger than d or zero, k is a constant and s is large enough. In the next section, we define some of these notions, state and discuss the results and the prior work in this language.

1.1 Multiplicity codes

Polynomial based error correcting codes, such as the Reed-Solomon codes and Reed-Muller codes, are a very important family of codes in coding theory both in theory and practice. Multiplicity codes are a natural generalization of Reed-Muller codes wherein at each evaluation point, one not only gives the evaluation of the polynomial f, but also all its derivatives up to a certain order.

Formally, let $\mathbb{F}$ be a field, s a positive integer, $S \subset \mathbb{F}$ an arbitrary subset of the field $\mathbb{F}$, $d \leq s|S|$ the degree parameter and $k \geq 1$ the ambient dimension. The codewords of the k-variate order-s multiplicity code of degree-d polynomials over $\mathbb{F}$ on the grid S^k is obtained by evaluating a k-variate polynomial of total degree at most d, along with all its derivatives of order less than s at all points in the grid S^k. Thus, a codeword corresponding to the polynomial f of total degree at most

k can be viewed as a function $\mathsf{Enc}_{s,S}(f)\colon S^k \to \mathbb{F}^{|E|}$ where $E := \{\mathbf{e} \in \mathbb{Z}^k_{\geq 0} \mid 0 \leq \|\mathbf{e}\|_1 < s\}$ and

$$\mathsf{Enc}_{s,S}(f)|_{\mathbf{a}} = \left(\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) : \mathbf{e} \in E\right)$$

where $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}$ is the Hasse derivative of the polynomial f with respect to $\mathbf{x}^{\mathbf{e}}$. The $s = 1$ version of these multiplicity codes corresponds to the classical Reed-Solomon codes (univariate case, $k = 1$) and Reed-Muller codes (multivariate setting, $k > 1$). The distance of these codes is $\delta := 1 - \frac{d}{s|S|}$, which follows from the multiplicity SZ Lemma mentioned earlier in the introduction.

Univariate multiplicity codes were first studied by Rosenbloom & Tsfasman [RT97] and Nielsen [Nie01]. Multiplicity codes for general k and s were introduced by Kopparty, Saraf and Yekhanin [KSY14] in the context of local decoding. Subsequently, Kopparty [Kop15] and Guruswami & Wang [GW13] independently proved that the univariate multiplicity codes over prime fields (or more generally over fields whose characteristic is larger than the degree of the underlying polynomials) achieve "list-decoding capacity". In the same work, Kopparty [Kop15] proved that multivariate multiplicity codes were list decodable up to the Johnson bound.

We remark that in the case of univariate multiplicity codes (both Reed-Solomon and larger order multiplicity codes), the decoding algorithms work for all choices of the set $S \subset \mathbb{F}$. However, all decoding algorithms for the multivariate setting (both Reed-Muller and larger order multiplicity codes) work only when the underlying set S has a nice algebraic structure (eg., $S = \mathbb{F}$) or when the degree d is very small (cf, the Reed-Muller list-decoding algorithm of Sudan [Sud97] and its multiplicity variant due to Guruswami & Sudan [GS99]). The only exception to this is the unique decoding algorithm of Kim and Kopparty [KK17] of Reed-Muller codes over product sets.

1.2 Our results

Below we state and contrast our results on the problem of decoding multivariate multiplicity codes (over grids) from a $\delta - \varepsilon$ fraction of errors for any constant $\varepsilon \in (0, 1)$ where δ is the distance of the code. Our first result is as follows.

Theorem 1.1 (List decoding of multivariate multiplicity codes with polynomial list size)**.** *For every $\varepsilon \in (0, 1)$ and integer k, there exists an integer s_0 such that for all $s \geq s_0$, degree parameter d, fields $\mathbb{F}$ of size q and characteristic larger than d, and any set $S \subseteq \mathbb{F}$ where $d < s|S|$, the following holds.*

For k-variate order-s multiplicity code of degree-d polynomials over $\mathbb{F}$ on the grid S^k, there is an efficient algorithm which when given a received word P , outputs all code words with agreement at least $(1 - \delta + \varepsilon)$ with P, where $\delta = 1 - d/(s|S|)$ is the relative distance of this code.

Remark 1.2. *A general product set in $\mathbb{F}^k$ is of the form $S_1 \times S_2 \times \cdots S_k$, where each S_i is a subset of $\mathbb{F}$. For the ease of notation, we always work with product sets which are grids S^k for some $S \subseteq \mathbb{F}$ even though all of our results hold for general product sets.* ⌟

As indicated before, this is the best one can hope for with respect to polynomial time list-decoding algorithms for multiplicity codes since there are super-polynomially many codewords with minimum distance $\delta = 1 - d/(s|S|)$ (see Appendix A). Till recently, it was not known if multivariate multiplicity codes were list decodable beyond the Johnson bound (even for the case $S = \mathbb{F}$). For the case of grids S^k, where $S \subseteq \mathbb{F}$ is an arbitrary set, even unique decoding algorithms were not known. We note that the above result does not yield a list-decoding algorithm for all multiplicities, but only for large enough multiplicities (based on the dimension k and the error parameter ε).

Kopparty, Ron-Zewi, Saraf and Wootters [KRSW18] showed how to reduce the size of the list for univariate multiplicity codes from polynomial to constant (dependent only on the error parameter ε). We use similar ideas, albeit in the multivariate setting, to reduce the list size in Theorem 1.1 to constant (dependent only on the error parameter ε and the dimension k).

Theorem 1.3 (List decoding of multivariate multiplicity codes with constant list size)**.** *For every $\varepsilon \in (0,1)$ and integer k, there exists an integer s_0 such that for all $s \geq s_0$, degree parameter d, fields $\mathbb{F}$ of size q and characteristic larger than d, and any set $S \subseteq \mathbb{F}$ where $d < s|S|$, the following holds.*

For k-variate order-s multiplicity code of degree-d polynomials over $\mathbb{F}$ on the grid S^k, there is a randomized algorithm which requires $\mathrm{poly}\left(d^{k^2}, |S|^{k^2}, \exp\left(O\left(\frac{k^2}{\varepsilon}\log^3\frac{1}{\varepsilon}\right)\right)\right)$ operations over the field $\mathbb{F}$ and which when given a received word P, outputs all code words with agreement at least $(1-\delta+\varepsilon)$ with P, where $\delta = 1 - d/(s|S|)$ is the relative distance of this code.

Moreover, the number of such codewords is at most $\exp\left(O\left(\frac{k^2}{\varepsilon}\log^2\frac{1}{\varepsilon}\right)\right)$.

Remark 1.4. *We remark that by taking a slightly different view of the list decoding algorithm Theorem 1.1 and Theorem 1.3, the upper bound on the number of field operations needed in Theorem 1.1 and Theorem 1.3 can be improved to $\mathrm{poly}(|S|^k, d^k)$. We sketch this view in subsection 4.7 and note the runtime analysis in Remark 4.8.* ⌟

The above two results are a generalization (and imply) the corresponding theorems for the univariate setting due to Kopparty [Kop15] and Guruswami & Wang [GW13] and Kopparty, Ron-Zewi, Saraf & Wootters [KRSW18]. We remark that Kopparty, Ron-Zewi , Saraf and Wootters [KRSW18] in the recent improvement to their earlier work prove a similar list-decoding algorithm for multivariate multiplicity codes as Theorem 1.3 for the case when $S = \mathbb{F}$. Though their list-decoding algorithm does not extend to products sets, it has the added advantage that it is *local*.

As noted earlier the only previous algorithmic method for decoding polynomial-based codes over product sets was that of Kim and Kopparty [KK17]. We describe the ideas in our algorithm shortly (in Section 2), but stress here that our approach is very different from that of Kim and Kopparty. Their work may be viewed as an algorithmic version of the inductive proof of the SZ lemma, and indeed recovers the SZ lemma as a consequence. Their work uses algorithmic aspects of algebraic decoding as a black box (to solve univariate cases). Our work, in contrast, only relies on the multiplicity SZ lemma as a black box. Instead, we open up the "algebraic decoding"

black box and make significant changes there, thus adding to the toolkit available to deal with polynomial evaluations over product sets.

1.3 Further discussion and open problems

Our result falls short of completely resolving the algorithmic SZ question in two respects; though it works for all dimensions k it only works when the multiplicity parameter s is large enough and when the characteristic of the field is either zero or larger than the degree parameter. Making improvements on any of these fronts is an interesting open problem.

All multiplicities: The algorithms presented in this paper decode all the way up to distance if the multiplicity parameter s is large enough. However, for small multiplicities, even the unique decoding problem is open. For $s = 1$, the result due to Kim and Kopparty [KK17] addresses the unique decoding question, but the list-decoding question for product sets is open.

Fields of small characteristic: All known proofs of list-decoding multiplicity codes beyond the Johnson bound (both algorithmic and combinatorial) require the field to be of zero characteristic or large enough characteristic. The problem of list-decoding multiplicity codes over small characteristic beyond the Johnson bound is open even for the univariate setting. As pointed to us by Swastik Kopparty, this problem of list-decoding univariate multiplicity codes over fields of small characteristic beyond the Johnson bound is intimately related to list-decoding Reed-Solomon codes beyond the Johnson bound.

For a more detailed discussion of multiplicity codes and related open problems, we refer the reader to the excellent survey by Kopparty [Kop14].

Organization

The rest of this paper is organized as follows. We begin with an overview of our proofs in Section 2 followed by some preliminaries (involving Hasse derivatives, their properties, multiplicity codes) in Section 3. We then describe and analyze the list-decoding algorithm for multivariate multiplicity codes in Section 4, thus proving Theorem 1.1. In Section 5, we then show how to further reduce the list-size to a constant, thus proving Theorem 1.3. In Section 6, we prove some properties of subspace restriction of multivariate multiplicity codes needed in Section 5. In Appendix A, we show that there are super-polynomially many minimum-weight codewords, thus proving the tightness of Theorems 1.1 and 1.3 with respect to list-decoding radius.

2 Proof overview

In this section, we first describe some of the hurdles in extending the univariate algorithms of Kopparty [Kop15] and Guruswami & Wang [GW13] to the multivariate setting, especially for

product sets and then given a detailed overview of the proofs of Theorem 1.1 and Theorem 1.3.

2.1 Background and motivation for our algorithm

To explain our algorithm, it will be convenient to recall the general polynomial method framework underlying the list-decoding algorithms in the univariate setting due to Kopparty [Kop15] and Guruswami & Wang [GW13]. . Let $P : S \to \mathbb{F}^s$ be the received word and $1 \leq m \leq s$

Step 1: Algebraic Explanation. Find a polynomial $Q(x, y_1, \ldots, y_m) \in \mathbb{F}[x, y_1, \ldots, y_m]$ of appropriate degree constraints that "explains" the received word P.

Step 2: Q contains the close codewords. Show that every low-degree polynomial f whose encoding agrees with P in more than $(1 - \delta + \varepsilon)$-fraction of points satisfies the following condition.

$$Q\left(x, f(x), \frac{\partial f}{\partial x}, \frac{\partial f}{\partial x^2}, \ldots, \frac{\partial f}{\partial x^{m-1}}\right) = 0\,.$$

Step 3: Reconstruction step. Recover every polynomial f that satisfies the above condition.

The main (and only) difference between the list-decoding algorithms of Kopparty [Kop15] and Guruswami & Wang [GW13] is that Guruswami and Wang show that it suffices to work with a polynomial Q which is linear in the y-variables, more precisely, $Q(x, y_1, \ldots, y_m)$ of the form $Q_0(x) + Q_1(x) \cdot y_1 + \cdots + Q_m(x) \cdot y_m$, while Kopparty allows for larger degrees in the y-variables. As a result, Kopparty performs the recovery step by solving a differential equation while Guruswami and Wang observe that dueto the simple structure of Q, the solution can be obtained by solving a linear system of equations.

How is multivariate list-decoding performed? There are by now two standard approaches. Inspired by the Pellikaan-Wu [PW04] observation that Reed-Muller codes are a subcode of Reed-Solomon codes over an extension field, Kopparty performs a similar reduction of the multivariate multiplicity code to a univariate multiplicity code over an extension field. Another approach is to solve the multivariate case by solving the univariate subproblem on various lines in the space. However, both these approaches work only if the set $S = \mathbb{F}$ or has some special algebraic structure.

For our proof, we take an alternate approach and always work in the multivariate setting without resorting to a reduction to the univariate setting. As we shall see, our approach has some advantages over that of Kopparty [Kop15], both in quantitative terms, since the algorithm can tolerate a larger number of errors, and in qualitative terms, since the underlying set of evaluation points does not have to be an algebraically nice subset of $\mathbb{F}^k$ as in [Kop15]; evaluations on an arbitrary grid suffice for the algorithm to work.

To extend the univariate list-decoding algorithm outlined above to the multivariate setting, we adopt the following approach. We consider a new set of formal variables $\mathbf{z}$ and instead of

directly working with the information about partial derivatives in the received word, we think of the partial derivatives of the same order as being *glued* together using monomials in $\mathbf{z}$. With this reorganized (and somewhat mysterious) view of the partial derivatives, we follow the outline of the univariate setting as described above. We find a polynomial Q with coefficients from the field of fractions $\mathbb{F}(\mathbf{z})$ instead of just $\mathbb{F}$ in the interpolation step to explain the received word P. Thus, in this instance, the linear system in the interpolation step is over the field $\mathbb{F}(\mathbf{z})$. We then argue that Q *contains* information about all the codewords that are close to the received word, and eventually *solve* Q to recover all the codewords close to the received word. This might seem rather strange to begin with, but these ideas of gluing together the partial derivatives and working over the field $\mathbb{F}(\mathbf{z})$ immediately generalize the univariate list decoding algorithm to the multivariate setting. Working with this field of fractions $\mathbb{F}(\mathbf{z})$ comes with its costs; it makes some of the steps costly and in particular, the recovery step far more elaborate than that in the Guruswami-Wang setting. However, this recovery step happens to be a special case of similar step in the recent work of Guo, Kumar, Saptharishi and Solomon [GKSS19] and we adapt their algorithm to our setting.

As a first attempt, a more standard way to generalize the algorithms of Kopparty [Kop15] and Guruswami & Wang [GW13] to the multivariate setting would have been to work with the partial derivatives directly. And, while this approach seems alright for the interpolation step, it seems hard to work with when we try to solve the resulting equation to recover all the close enough codewords. In particular, it isn't even clear in this set up that the number of solutions of the algebraic explanation (and hence, the number of close enough codewords) is polynomially bounded. This mysterious step of gluing together derivatives of the same order in a reversible manner (in the sense that we can read off the individual derivatives from the glued term) gets around this problem, and makes it viable to prove a polynomial upper bound on the number of solutions, and eventually solve the equation to recover all the close enough codewords.

Given this background, we now give a more detailed outline of our algorithm below.

2.2 Theorem 1.1 : Multivariate list-decoding algorithm with polynomial-sized lists

Viewing the encoding as a formal power series

Multiplicity codes are described by saying that the encoding of a polynomial $f \in \mathbb{F}[\mathbf{x}]$ consists of the evaluation of *all* partial derivatives of f of order at most $s-1$ at every point in the appropriate evaluation set, e.g. the grid S^k. For our algorithm, we think of these partial derivatives of f as being rearranged on the basis of the order of the derivatives as follows. We take a fresh set of formal variables $\mathbf{z}$ and define the following differential operators.

$$\Delta_i(f) := \sum_{\mathbf{e}:\|\mathbf{e}\|_1=i} \mathbf{z}^{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}$$

where $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}$ denotes the Hasse derivative[4] of the polynomial f with respect to $\mathbf{x}^{\mathbf{e}}$.

Let $\Delta(f)$ be an s tuple of polynomials defined as follows.

$$\Delta(f) := (\Delta_0(f), \Delta_1(f), \ldots, \Delta_{s-1}(f)) \,.$$

We view the encoding for f as giving us the evaluation of the tuple $\Delta(f) \in \mathbb{F}[\mathbf{x}, \mathbf{z}]$ as $\mathbf{x}$ varies in S^k. Note that for every fixing of $\mathbf{x}$ to some $\mathbf{a} \in S^k$, $\Delta(f)(\mathbf{a})$ is in $\mathbb{F}[\mathbf{z}]^s$. Thus, the alphabet size is still large. Clearly, this is just a change of viewpoint, as we can go from the original encoding to this and back efficiently, and at this point it is unclear that this change of perspective would be useful.

Finding an equation satisfied by all close enough codewords

Let P be a received word. We view P as a function $P : S^k \to \Sigma^s$, where $\Sigma = \mathbb{F}[\mathbf{z}]$, as discussed in the previous step. The goal of the decoding step is to find all the polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d, whose encoding is close enough to P.

As a first step towards this, we find a non-zero polynomial $Q(\mathbf{x}, \mathbf{y}) \in \mathbb{F}(\mathbf{z})[\mathbf{x}, \mathbf{y}]$ of the form

$$Q(\mathbf{x}, \mathbf{y}) = Q_1(\mathbf{x})y_1 + \cdots + Q_m(\mathbf{x})y_m \,,$$

which *explains* the received word P, i.e., for every $\mathbf{a} \in S^k$, $Q(\mathbf{a}, P(\mathbf{a})) = 0$, and Q satisfies some appropriate degree constraints. Here $m \leq s$ is a parameter. For technical reasons, we also end up imposing some more constraints on Q in terms of its partial derivatives, the details of which can be found in Section 4.3. Each of these constraints can be viewed as a homogeneous linear equation in the coefficients of Q over the field $\mathbb{F}(\mathbf{z})$. We choose the degree of Q to be large enough to ensure that this system has more variables than constraints, and therefore, has a non-zero solution.

This step is the interpolation step which shows up in any standard application of the polynomial method, and our set up is closest and a natural generalization of the set up in the list decoding algorithm of Guruswami and Wang [GW13] for univariate multiplicity codes.

The key property of the polynomial Q thus obtained is that for every degree d polynomial $f \in \mathbb{F}[\mathbf{x}]$ whose encoding is close enough to P,

$$Q(\mathbf{x}, \Delta(f)) = Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0 \,.$$

To see this, we note that from the upper bound on the degree of Q and the fact that f has degree at most d, the polynomial $Q(\mathbf{x}, \Delta(f)) \in \mathbb{F}(\mathbf{z})[\mathbf{x}]$ is of not too high degree in $\mathbf{x}$. Moreover, from the constraints imposed on Q during interpolation, it follows that at every $\mathbf{a} \in S^k$ where the encoding of f and P agree, $Q(\mathbf{x}, \Delta(f))$ vanishes with high multiplicity. Thus, if the parameters are favorably

[4]Since we have both $\mathbf{x}$ and $\mathbf{z}$ variables, we use the notation $\frac{\partial f}{\partial x}$ to denote the Hasse derivative wrt variable x to explicitly indicate which variable the derivative is being taken

set, it follows that $Q(\mathbf{x}, \Delta(f))$ has too many zeroes of high multiplicity on a grid, and hence by the multiplicity Schwartz-Zippel emma (see Lemma 3.4), $Q(\mathbf{x}, \Delta(f))$ must be identically zero.

We note that this is the only place in the proof where we use anything about the structure of the set of evaluation points, i.e., the set of evaluation points is a grid.

Solving the equation to recover all close enough codewords

As the final step of our algorithm, we try to recover all polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d such that

$$Q(\mathbf{x}, \Delta(f)) = Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

$Q(\mathbf{x}, \Delta(f))$ can be viewed as a partial differential equation of order $m-1$ and degree one, and we construct all candidate solutions f via the method of power series. We start by trying all possible choices of field elements for coefficients of monomials of degree at most $m-1$ in f, and iteratively recover the remaining coefficients of f by reconstructing f one homogeneous component at a time. Moreover, we observe that for each choice of the initial coefficients, there is a unique lift to a degree d polynomial. Thus, the number of solutions is upper bounded by the number of initial choices, which is at most $|\mathbb{F}|^{\binom{m+k-1}{k}}$.

We note that this is one place where working with $\Delta_i(f)$ as opposed to having an equation in the individual partial derivatives of f is of crucial help. Even though the equation $Q(\mathbf{x}, \Delta(f)) = 0$ is a partial differential equation of high order in f, the fact that these derivatives appear in a structured form via the operators $\Delta_i(f)$ helps us prove a polynomial upper bound on the number of such solutions and solve for f. Without this additional structure, it is unclear if one can prove a polynomial upper bound on the number of solutions of the corresponding equation.

This reconstruction step is a multivariate generalization of similar reconstruction steps in the list decoding algorithms of Kopparty [Kop15] and Guruswami & Wang [GW13] for univariate multiplicity codes. Interestingly, this is also a special case of a similar reconstruction procedure in the work of Guo, Kumar, Saptharishi and Solomon [GKSS19], where the polynomial Q could potentially be of higher degree in $\mathbf{y}$ variables, and is given to us via an arithmetic circuit of small size and degree and the goal is to show that all (low degree) polynomials f, satisfying $Q(\mathbf{x}, \Delta(f)) \equiv 0$ have small circuits. In contrast, we are working with Q which is linear in $\mathbf{y}$ and we have access to the coefficient representation of this polynomial, and construct the solutions f in the monomial representation. As a consequence, the details of this step are much simpler here, when compared to that in [GKSS19].

In this step of our algorithm viewing the encoding in terms of the differential operators $\Delta_i()$ turns out to be useful. The iterative reconstruction outlined above crucially uses the fact that for any homogeneous polynomial $g \in \mathbb{F}[\mathbf{x}]$ of degree r, $\Delta_i(g)$ is a homogeneous polynomial in the $\mathbf{x}$

variables of degree exactly $r - i + 1$. The other property that we use from $\Delta_i()$ is that given $\Delta_i(g)$ for any homogeneous polynomial g, we can uniquely read off all the partial derivatives of order $i-1$ of g, and via a folklore observation of Euler, uniquely reconstruct the polynomial g itself (see Lemma 4.4).

Finally, we note that the precise way of *gluing* together the partial derivatives of order i in the definition of the operator $\Delta_i()$ is not absolutely crucial here, and as is evident in Lemma 4.4, many other candidates would have satisfied the necessary properties.

The details of this step are in Section 4.5, and essentially complete the proof of Theorem 1.1.

2.3 Theorem 1.3: Reducing the list size to a constant

In Section 5, we combine our proof of Theorem 1.1 with the techniques in the recent work of Kopparty, Ron-Zewi, Saraf and Wootters [KRSW18] to show that the list size in the decoding algorithm in Theorem 1.1 can be reduced to a constant.

The key to this step is the observation that since $Q(\mathbf{x}, \mathbf{y})$ is linear in the $\mathbf{y}$ variables, the solutions f of the equation $Q(\mathbf{x}, \Delta(f)) \equiv 0$ form an affine subspace of polynomials. The reconstruction algorithm in Section 4.5 in fact gives us an affine subspace $V \subseteq \mathbb{F}[\mathbf{x}]$ of polynomials of degree at most d which consists of all the solutions of $Q(\mathbf{x}, \Delta(f)) \equiv 0$.

This is precisely the setting in the work of Kopparty, Ron-Zewi, Saraf and Wootters [KRSW18] in the context of folded Reed-Solomon codes and univariate multiplicity codes, and we essentially apply their ideas off the shelf, and combine them with our proof of Theorem 1.1 to reduce the list size to a constant.

In general, this idea of solving $Q(\mathbf{x}, \Delta(f)) \equiv 0$ to recover a subspace, and then using the ideas in [KRSW18] to recover codewords in the subspace which are close to the received word has the added advantage that it can be applied over all fields. As an immediate consequence, we get an analog of Theorem 1.1 over infinite fields like rationals as well.

3 Preliminaries

3.1 Notation

We use the following notation.

- $\mathbb{F}$ is the field we work over, and we assume the characteristic of $\mathbb{F}$ to be either zero or larger than the degree parameter d of the message space.
- We use bold letters to denote tuples of variables (i.e., $\mathbf{x}$, $\mathbf{z}$, $\mathbf{y}$ for $(x_1, \ldots, x_k)$, $(z_1, \ldots, z_k)$ and $(y_1, \ldots, y_m)$ respectively).

- We work with polynomials which are in general members of $\mathbb{F}(\mathbf{z})[\mathbf{x}, \mathbf{y}]$. We denote monomials in $\mathbf{x}$ and $\mathbf{z}$ by $\mathbf{x}^{\mathbf{e}}$ $(= \prod_{i\in[k]} x_i^{e_i})$, $\mathbf{z}^{\mathbf{e}}$ $(= \prod_{i\in[k]} z_i^{e_i})$ respectively where $\mathbf{e} \in \mathbb{Z}^k_{\geq 0}$. The degree of the monomial is $\|\mathbf{e}\|_1 = \sum_{i=1}^k e_i$.
- For $\mathbf{e}, \mathbf{e}' \in \mathbb{Z}^k_{\geq 0}$ we say $\mathbf{e}' \leq \mathbf{e}$ iff for all $i \in [k]$ we have $e'_i \leq e_i$. Also, we use $\binom{\mathbf{e}}{\mathbf{e}'}$ to denote $\prod_{i\in[k]} \binom{e_i}{e'_i}$.
- For a natural number n, $[n]$ denotes the set $\{1, 2, \ldots, n\}$.

3.2 Hasse derivatives

Throughout the paper we work with Hasse derivatives: we interchangeably use the term partial derivatives.

Definition 3.1 (Hasse Derivative)**.** *For a polynomial $f \in \mathbb{F}[\mathbf{x}]$ the Hasse derivative of type $\mathbf{e}$ is the coefficient of $\mathbf{z}^{\mathbf{e}}$ in the polynomial $f(\mathbf{x}+\mathbf{z}) \in \mathbb{F}[\mathbf{x}, \mathbf{z}]$. We denote this by $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}$ or $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}$* ⌟

We state some basic properties of Hasse Derivatives below. Some of these are taken from [DKSS13, Proposition 4].

Proposition 3.2 (Basic Properties of Hasse Derivatives)**.** *Let $f, g \in \mathbb{F}[\mathbf{x}]$ and consider $\mathbf{e}, \mathbf{e}' \in \mathbb{Z}^k_{\geq 0}$.*

1. $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}} + \frac{\partial g}{\partial \mathbf{x}^{\mathbf{e}}} = \frac{\partial (f+g)}{\partial \mathbf{x}^{\mathbf{e}}}$.
2. *If f is a homogeneous polynomial of degree d then $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}$ is homogeneous polynomial of degree $d - \|\mathbf{e}\|_1$.*
3. *If $f = \mathbf{x}^{\mathbf{e}'}$ then $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}} = \binom{\mathbf{e}}{\mathbf{e}'} \mathbf{x}^{\mathbf{e}-\mathbf{e}'}$.*
4. *Hasse derivatives compose in the following manner:*

$$\frac{\partial}{\partial \mathbf{x}^{\mathbf{e}}} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}} = \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}+\mathbf{e}'}}.$$

5. *Product rule for Hasse derivatives:*

$$\frac{\partial \left(\prod_{i\in[w]} f_i\right)}{\partial \mathbf{x}^{\mathbf{e}}} = \sum_{\mathbf{u}_1+\mathbf{u}_2+\ldots+\mathbf{u}_w=\mathbf{e}} \left(\prod_{i\in[w]} \frac{\partial f_i}{\partial \mathbf{x}^{\mathbf{u}_i}}\right).$$

Proof. Items 1 to 3 and 5 follow directly from Definition 3.1. For Item 4, observe that by linearity of Hasse derivatives we may assume WLOG that f is a monomial, say $\mathbf{x}^{\tilde{\mathbf{e}}}$: in this case the claim follows from Item 3 and the fact that $\binom{\tilde{\mathbf{e}}}{\mathbf{e}} \cdot \binom{\tilde{\mathbf{e}}-\mathbf{e}}{\mathbf{e}'} = \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \binom{\tilde{\mathbf{e}}}{\mathbf{e}+\mathbf{e}'}$. □

3.3 Multiplicity code

We now define the notion of multiplicity of a polynomial $f \in \mathbb{F}[\mathbf{x}]$ at a point $\mathbf{a} \in \mathbb{F}^k$. The multiplicity of f at the origin is ℓ iff ℓ is the highest integer such that no monomial of total degree less than ℓ appears in the coefficient representation of f. We formalize this below using Hasse derivatives.

Definition 3.3 (multiplicity)**.** *A polynomial $f \in \mathbb{F}[\mathbf{x}]$ is said to have multiplicity ℓ at a point $\mathbf{a} \in \mathbb{F}^k$, denoted by* $\mathrm{mult}(f, \mathbf{a})$*, iff ℓ is the largest integer such that for all $\mathbf{e} \in \mathbb{Z}^k_{\geq 0}$ with $\|\mathbf{e}\|_1 < \ell$ we have $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) = 0$. If no such ℓ exists then* $\mathrm{mult}(f, \mathbf{a}) = \infty$. ⌟

Dvir, Kopparty, Saraf and Sudan proved the following higher order multiplicity version of the classical Schwartz-Zippel lemma.

Lemma 3.4 (multiplicity SZ lemma [DKSS13, Lemma 2.7])**.** *Let $\mathbb{F}$ be any field and let S be an arbitrary subset of $\mathbb{F}$. Then, for any non-zero k-variate polynomial P of degree at most d,*

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

The above lemma implies the classical SZ lemma, which states that two distinct k-variate polynomials of degree d cannot agree everywhere on a grid S^k for any set S of size larger than d trivially. This in particular tells us that the grid S^k serves as hitting set for polynomials of degree at most d provided $d < |S|$.

As mentioned before, a multiplicity code over a grid S^k consists of evaluations of the message polynomial f along with its derivatives of various orders (up to $s-1$), at the points of the grid.

Definition 3.5 (multiplicity code)**.** *Let $s, k \in \mathbb{N}$, $d \in \mathbb{Z}_{\geq 0}$, $\mathbb{F}$ a field and $S \subset \mathbb{F}$ a non-empty finite subset. The k-variate order-s multiplicity code of degree-d polynomials over $\mathbb{F}$ on the grid S^k is defined as follows.*

Let $E := \{\mathbf{e} \in \mathbb{Z}^k_{\geq 0} \mid 0 \leq \|\mathbf{e}\|_1 < s\}$. Note that $|E| = \binom{s+k-1}{k}$. The code is over alphabet $\mathbb{F}^E$ and has length S^k (where the coordinates are indexed by elements of S^k).

The code is an $\mathbb{F}$-linear map from the space of degree d polynomials in $\mathbb{F}[\mathbf{x}]$ to $\left(\mathbb{F}^E\right)^{S^k}$. The encoding of $f \in \mathbb{F}[\mathbf{x}]$ at a point $\mathbf{a} \in S^k$ is given by:

$$\mathsf{Enc}_{s,S}(f)|_{\mathbf{a}} = \left(\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) : \mathbf{e} \in E\right).$$ ⌟

Remark 3.6.

- *The distance of the code is exactly $\delta := 1 - \frac{d}{s|S|}$ and the rate of the of the code is $\frac{\binom{d+k}{k}}{\binom{s+k-1}{k} \cdot |S|^k}$.*
- *As mentioned in the introduction we can also view the encoding by clubbing partial derivatives of the same degree. Thus, the encoding of f at a point $\mathbf{a}$ is $(\Delta_0(f)(\mathbf{a}), \Delta_1(f)(\mathbf{a}), \ldots, \Delta_{s-1}(f)(\mathbf{a})) \in \mathbb{F}[\mathbf{z}]^s$*

where $\Delta_i(f)(\mathbf{a}) = \sum_{\mathbf{e}:\|\mathbf{e}\|_1=i} \mathbf{z}^{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a})$.

- *We think of k, m and s as constants, but m much larger than k and s is much larger than m. The precise trade-offs will be alluded to when we need to set parameters in our proofs.*

⌟

3.4 Computing over polynomial rings

In this section, we state a few basic results that show how to perform algebraic operations over polynomial rings.

The following lemma, proved via an easy application of polynomial interpolation, lets us construct the coefficient representation of a polynomial given an arithmetic circuit for it.

Lemma 3.7. *Let $k \in \mathbb{N}$. There exists a deterministic algorithm that takes as input an arithmetic circuit C of size s that computes a k-variate polynomial $P \in \mathbb{F}[\mathbf{z}]$ of degree at most d and outputs the coefficient vector of P in at most* $\mathrm{poly}(d^k, s)$ *field operations over $\mathbb{F}$*

Proof. From Lemma 3.4, we know that no two degree d polynomials can agree everywhere on a grid of size larger than d. So, we pick an arbitrary subset S of $\mathbb{F}$ of size $d+1$ and evaluate the circuit C at all points on the grid $|S|^k$. This requires at most $\mathrm{poly}(d^k, s)$ field operations. Now, given these evaluations, we set up a linear system in the coefficients of P where for every $\mathbf{a}$ in the grid, we have a constraint of the form $P(\mathbf{a}) = C(\mathbf{a})$. We know that this system has a solution. Furthermore, from Lemma 3.4, we know that this system has a unique solution.

Solving this system gives us the coefficient vector of P and requires at most d^k additional field operations. □

The next lemma tells us how to perform linear algebra over the polynomial ring $\mathbb{F}[\mathbf{z}]$.

Lemma 3.8 (linear algebra over polynomial rings)**.** *Let $A(\mathbf{z}) \in \mathbb{F}[\mathbf{z}]^{t' \times t}$ be a matrix such that each of its entries is a polynomial of degree at most m in the variables $\mathbf{z} = (z_1, z_2, \ldots, z_k)$ and $t' \leq t$. Then, there is a deterministic algorithm which takes as input the coefficient vectors of the entries of A and outputs a non-zero vector $\mathbf{u} \in \mathbb{F}[\mathbf{z}]^t$ in time* $\mathrm{poly}(m^k, t^k)$ *such that $A \cdot \mathbf{u} = \mathbf{0}$. Moreover, every entry in $\mathbf{u}$ is a polynomial of degree at most tm.*

Proof. As a first step, we reduce this to the problem of solving a linear system of the form $A' \cdot \mathbf{u}' = \mathbf{b}$, where A' and $\mathbf{b}$ have entries in $\mathbb{F}[\mathbf{z}]$ of degree at most m, and A' is a square matrix of dimension at most t', which is non-singular. At this point, we can just apply Cramer's rule to find a solution of this system.

Since $t' \leq t$, the rank r of $A(\mathbf{z})$ over $\mathbb{F}(\mathbf{z})$ is at most t'. Thus, there is a square submatrix $A'(\mathbf{z})$ of A such that $\det(A')$ is a non-zero polynomial of degree at most $mr \leq mt'$ in $\mathbb{F}[\mathbf{z}]$. For a hitting set $H_{mt',k}$ of polynomials of degree at most mt' on k variables over $\mathbb{F}$, we consider the set of matrices

$\{A(\mathbf{c}) : \mathbf{c} \in H_{mt',k}\}$. From the guarantees of the hitting set, we know that there is a $\mathbf{c} \in H_{mt',k}$ such that $A'(\mathbf{c})$ is of rank equal to r. Let $\mathbf{c}_0 \in H_{mt',k}$ be such that the rank of $A(\mathbf{c}_0)$ over $\mathbb{F}$ is maximum among all matrices in the set $\{A(\mathbf{c}) : \mathbf{c} \in H_{mt',k}\}$. Moreover, let $A'(\mathbf{z})$ be a submatrix of $A(\mathbf{z})$ such that $\text{rank}(A'(\mathbf{c}_0))$ equals $\text{rank}(A(\mathbf{c}_0))$. From Lemma 3.4, there is an explicit hitting set $H_{mt',k}$ of size at most $(mt'+1)^k \leq (mt+1)^k$. Thus, we can find $A'(\mathbf{z})$ of rank equal to the rank of $A(\mathbf{z})$ with at most $\text{poly}(m^k, t^k)$ field operations over $\mathbb{F}$. Without loss of generality, let us assume that A' is the top left submatrix of A of size r. Clearly, the $(r+1)$-st column of A is linearly dependent on the first r columns of A over the field $\mathbb{F}(\mathbf{z})$. In other words, the linear system given by

$$A' \cdot \mathbf{u}' = \mathbf{b}$$

where $\mathbf{b} = (A_{1,r+1}, A_{2,r+1}, \ldots, A_{r,r+1})$, has a solution in $\mathbb{F}(\mathbf{z})$. Moreover, for every solution $\mathbf{u}'$ of this system, where $\mathbf{u}' = (u'_1, u'_2, \ldots, u'_r)$, the t dimensional vector $(u'_1, u'_2, \ldots, u'_r, -1, 0, \ldots, 0)$ is in the kernel of $A(\mathbf{z})$. Also, since $A \cdot \mathbf{u} = \mathbf{0}$ is a homogeneous linear system, for any non-zero polynomial $P(\mathbf{z})$, $(P \cdot u'_1, P \cdot u'_2, \ldots, P \cdot u'_r, -P, 0, \ldots, 0)$ continues to be a non-zero vector in the kernel of $A(\mathbf{z})$.

Since A' is non-singular, $\mathbf{u}' = (A')^{-1} \cdot \mathbf{b}$ is a solution to this system. Moreover, by Cramer's rule, $(A')^{-1} = \text{adj}(A')/\det(A')$, where $\text{adj}(A')$ is the adjugate matrix of A' and $\det(A')$ is its determinant. Since, every entry of $\text{adj}(A')$ is a polynomial in $\mathbb{F}[\mathbf{z}]$ of degree at most tm, we get a solution of the form $\mathbf{u}' = (p_1/\det(A'), p_2/\det(A'), \ldots, p_r/\det(A'))$ where each p_i is a polynomial in $\mathbb{F}[\mathbf{z}]$ of degree at most tm. By getting rid of the denominators by scaling by $\det(A')$, we get that the non-zero t dimensional vector $(p_1, p_2, \ldots, p_r, -\det(A'), 0, \ldots, 0)$ is in the kernel of $A(\mathbf{z})$.

Moreover, using the fact that the determinant polynomial has a polynomial size efficiently constructible circuit, and Lemma 3.7, we can output this vector, with each entry being a list of coefficients in $\mathbb{F}$ in time $\text{poly}(m^k, t^k)$ via an efficient deterministic algorithm. □

4 List decoding the multivariate multiplicity code

In this section, we prove Theorem 1.1. We follow the outline of the proof described in Section 2. We start with the interpolation step.

4.1 Viewing the encoding as a formal power series

The message space is the space of k-variate polynomials of degree at most d over $\mathbb{F}$. In the standard encoding, we have access to evaluations of the polynomial and all its derivatives of order up to $s-1$ on all points on a grid $S^k \subseteq \mathbb{F}^k$.

For our proof, it will be helpful to group the derivatives of the same order together.

Definition 4.1. *Let $f \in \mathbb{F}[\mathbf{x}]$ be a polynomial. Then, for any $i \in \mathbb{Z}_{\geq 0}$, $\Delta_i(f)$ is defined as*

$$\Delta_i(f) := \sum_{\mathbf{e}:\|\mathbf{e}\|_1=i} \mathbf{z}^{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}\,. \qquad \lrcorner$$

So, we have a distinct monomial in $\mathbf{z}$ attached to each of the derivatives. The precise form of the monomial in $\mathbf{z}$ is not important, and all that we will use is that these monomials are linearly independent over the underlying field, don't have very high degree and there aren't too many variables in $\mathbf{z}$.

Now, we think of the encoding of f as giving us the evaluation of the tuple of polynomials $\Delta(f) = (\Delta_0(f(\mathbf{x})), \Delta_1(f(\mathbf{x})), \ldots, \Delta_{s-1}(\mathbf{x})) \in \mathbb{F}(\mathbf{z})[\mathbf{x}]^s$ as $\mathbf{x}$ takes values in $\mathbb{F}^k$.

Note that $\Delta_i(f)$ is a homogeneous polynomial of degree at equal to i in $\mathbf{z}$.

4.2 The τ operator

We will need to compute the Hasse derivative of $\Delta_i(f)$ with respect to $\mathbf{x}^{\mathbf{e}}$, i.e., $\frac{\partial \Delta_i(f)}{\partial \mathbf{x}^{\mathbf{e}}}$. From the definition of $\Delta_i(f)$, we have

$$\begin{aligned}\frac{\partial \Delta_i(f)}{\partial \mathbf{x}^{\mathbf{e}}} &= \sum_{\mathbf{e}':\|\mathbf{e}'\|_1=i} \mathbf{z}^{\mathbf{e}'} \cdot \frac{\partial}{\partial \mathbf{x}^{\mathbf{e}}} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}} = \sum_{\mathbf{e}':\|\mathbf{e}'\|_1=i} \mathbf{z}^{\mathbf{e}'} \cdot \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}+\mathbf{e}'}} \\ &= \sum_{\mathbf{e}':\|\mathbf{e}'\|_1=i} \mathbf{z}^{\mathbf{e}'} \cdot \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \mathrm{coeff}_{\mathbf{z}^{\mathbf{e}+\mathbf{e}'}}(\Delta_{i+\|\mathbf{e}\|_1} f(\mathbf{x}))\,.\end{aligned}$$

The key point to note is that the Hasse derivative of $\Delta_i(f)$ with respect to $\mathbf{x}^{\mathbf{e}}$ can be read off the coefficients of $\Delta_{i+\|\mathbf{e}\|_1}(f)$.

This motivates the following definition. Consider a tuple $P = (P_0, P_1, \ldots, P_{s-1})$, where for each i, P_i is a homogeneous polynomial of degree i in $\mathbb{F}[\mathbf{z}]$. For any $\mathbf{e} \in \mathbb{Z}^k_{\geq 0}$, and $i \leq s-1$ such that $i + \|\mathbf{e}\|_1 \leq s-1$, we define

$$\tau^{(i)}_{\mathbf{e}}(P) := \sum_{\mathbf{e}':\|\mathbf{e}'\|_1=i} \mathbf{z}^{\mathbf{e}'} \cdot \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \mathrm{coeff}_{\mathbf{z}^{\mathbf{e}+\mathbf{e}'}}(P_{i+\|\mathbf{e}\|_1})\,.$$

Thus, for $\Delta(f) = (\Delta_0(f(\mathbf{x})), \Delta_1(f(\mathbf{x})), \ldots, \Delta_{s-1}(\mathbf{x}))$, we have

$$\tau^{(i)}_{\mathbf{e}}(\Delta(f)) = \frac{\partial \Delta_i(f)}{\partial \mathbf{x}^{\mathbf{e}}}\,.$$

4.3 Interpolation step

Let P be the received word, Thus, we are given a collection of s-tuples of polynomials $P(\mathbf{a}) = (P_0(\mathbf{a}), P_1(\mathbf{a}), \ldots, P_{s-1}(\mathbf{a}))$ for every $\mathbf{a} \in S^k$, where each $P_i(\mathbf{a})$ is a homogeneous polynomial of

degree i in $\mathbf{z}$. From the earlier definition of τ, given such a $P(\mathbf{a})$, we have $\tau_{\mathbf{e}}^{(i)}(P(\mathbf{a}))$ for every $i \leq m$ and $\mathbf{e}$ with $\|\mathbf{e}\|_1 \leq s-1-m$.

Lemma 4.2. *Let k and s be constants. For every natural number $m \leq s-1-k$, and $D = 10|S|(s-m)/m^{1/k}$, there is a non-zero polynomial $Q(\mathbf{x},\mathbf{y}) = Q_1(\mathbf{x})y_1 + \cdots + Q_m(\mathbf{x})y_m \in \mathbb{F}(\mathbf{z})[\mathbf{x},\mathbf{y}]$ such that*

- *For every $i \in \{1,2,\ldots,m\}$, the $\mathbf{x}$-degree of each Q_i is at most D.*
- *For every $\mathbf{a} \in S^k$ and every $\mathbf{e} \in \mathbb{Z}^k_{\geq 0}$ such that $0 \leq \|\mathbf{e}\|_1 \leq s-1-m$, $\Delta_{\mathbf{e}}(Q)(\mathbf{a}) = 0$, where*

$$\Delta_{\mathbf{e}}(Q)(\mathbf{a}) := \sum_{i=1}^{m} \sum_{\mathbf{e}' \leq \mathbf{e}} \frac{\partial Q_i(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}}(\mathbf{a}) \cdot \tau_{\mathbf{e}-\mathbf{e}'}^{(i-1)}(P(\mathbf{a}))\,.$$

  Here, $\mathbf{e}' \leq \mathbf{e}$ means that $\mathbf{e}$ dominates $\mathbf{e}'$ coordinate wise.

Moreover, the coefficients of Q are polynomials in $\mathbb{F}[\mathbf{z}]$ of degree at most $O(|S|^k s^{2k})$, and such a Q can be deterministically constructed by using at most $\mathrm{poly}(|S|^{k^2}, s^{k^2}, d^k)$ operations over the field $\mathbb{F}$.

Proof. We start by showing the existence of a polynomial Q with the appropriate degree constraints, followed by an analysis of the running time.

Existence of Q. We view the above constraints as a system of linear equations over the field $\mathbb{F}(\mathbf{z})$, where the variables are the coefficients of Q. The number of homogeneous linear constraints is $|S|^k\binom{s-m+k}{k}$ and the number of variables is $m\binom{D+k}{k}$.

By using the fact that k is much smaller than s, and a crude approximation of the binomial coefficients, we have $|S|^k\binom{s-m+k}{k} \leq (2e|S|(s-m)/k)^k$ and $m\binom{D+k}{k} > m(D/k)^k$. Plugging in the value of D, we get $m(D/k)^k = (10|S|(s-m)/k)^k$, which is clearly greater than the number of constraints. Hence, there is a non-zero solution, where the coefficients of the polynomial are from the field $\mathbb{F}(\mathbf{z})$, i.e., are rational functions in $\mathbf{z}$.

Next we analyze the degree of these coefficients and show that we can recover such a Q efficiently, with the appropriate degree bounds.

The running time. For the running time, we recall that each $\tau_{\mathbf{e}}^{i}$ is a polynomial of degree at most $m-1$ in the $\mathbf{z}$ variables. As a consequence, observe that the linear system we have for the coefficients of Q is of the form $A \cdot \mathbf{u} = 0$, where A is a matrix with dimension at most $O(|S|^k(s-m)^k)$ over the ring $\mathbb{F}[\mathbf{z}]$, and every entry of A is a polynomial in $\mathbb{F}[\mathbf{z}]$ of degree at most m. From Lemma 3.8, we get that we can find a non-zero solution in $\mathbb{F}[\mathbf{z}]$ using at most $\mathrm{poly}(|S|^{k^2}, s^{k^2})$ field operations over $\mathbb{F}$. Moreover, each of the coordinates of this output vector is a polynomial of degree at most $O(|S|^k(s-m)^k) \cdot m = O(|S|^k s^{2k})$ in $\mathbb{F}[\mathbf{z}]$. □

Going forward, we work with the polynomial Q and the degree parameter D as set in Lemma 4.2.

4.4 Close enough codewords satisfy the equation

We now show that for every polynomial $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d whose encoding is close enough to the received word P, f *satisfies* the equation Q in some sense.

Lemma 4.3. *If $f \in \mathbb{F}[\mathbf{x}]$ is a degree d polynomial such that the number of $\mathbf{a} \in S^k$ which satisfy*

$$P(\mathbf{a}) = \Delta(f)(\mathbf{a}),$$

is at least $T > (D+d) \cdot |S|^{k-1}/(s-m)$, then $Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f))$ is identically zero as a polynomial in $\mathbb{F}(\mathbf{z})[\mathbf{x}]$.

Proof. Define the polynomial $R \in \mathbb{F}(\mathbf{z})[\mathbf{x}]$ as follows

$$R(\mathbf{x}) := Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) = \sum_{i=1}^{m} Q_i(\mathbf{x}) \cdot \Delta_{i-1}(f).$$

R is a polynomial in $\mathbf{x}$ of degree at most $D+d$ over the field $\mathbb{F}(\mathbf{z})$. Whenever $\mathbf{a}$ satisfies that $P(\mathbf{a}) = \Delta(f)(\mathbf{a})$, from the definitions of $\tau_{\mathbf{e}}^{(i)}$ and $\Delta_{\mathbf{e}}$, we have that for all $\mathbf{e}$ such that $0 \leq \|\mathbf{e}\|_1 \leq s-m-1$,

$$\begin{aligned}
\frac{\partial R(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) &= \sum_{i=1}^{m} \sum_{\mathbf{e}' \leq \mathbf{e}} \frac{\partial Q_i(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}}(\mathbf{a}) \cdot \frac{\partial \Delta_{i-1}(f)}{\partial \mathbf{x}^{\mathbf{e}-\mathbf{e}'}}(\mathbf{a}) \\
&= \sum_{i=1}^{m} \sum_{\mathbf{e}' \leq \mathbf{e}} \frac{\partial Q_i(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}}(\mathbf{a}) \cdot \tau_{\mathbf{e}-\mathbf{e}'}^{(i-1)}(P(\mathbf{a})) \\
&= \Delta_{\mathbf{e}}(Q)(\mathbf{a}) \\
&= 0.
\end{aligned}$$

Hence, at every point of agreement between $\Delta(f)$ and the received word P, $R(\mathbf{x})$ vanishes with multiplicity at least $s-m$. From Lemma 3.4, we know that if

$$T(s-m) > (D+d)|S|^{k-1},$$

then, R must be identically zero. □

Let us try to get a sense of the parameters here. The relative distance of this code is $\delta = 1 - \frac{d}{s|S|}$. Now, in $\frac{T}{|S|^k} > \frac{D+d}{|S|(s-m)}$, plugging in the value of D from the earlier discussion gives us

$$\begin{aligned}
\frac{T}{|S|^k} &> \frac{d}{|S|(s-m)} + \frac{10|S|(s-m)/m^{1/k}}{|S|(s-m)} \\
&= \frac{10}{m^{1/k}} + \left(\frac{s}{s-m}\right) \cdot \frac{d}{s|S|}
\end{aligned}$$

$$= \frac{10}{m^{1/k}} + \left(\frac{m}{s-m}\right) \cdot \frac{d}{s|S|} + \frac{d}{s|S|}\,.$$

In our final analysis for the proof of Theorem 1.1, we choose m and s large enough as a function of ε, so that this bound is of the form $\varepsilon + (1-\delta)$, which is precisely what is claimed in Theorem 1.1.

4.5 Solving the equation to find close enough codewords

All that remains now is to solve equations of the form $Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f))$ to recover f. This would be done via iteratively constructing f one homogeneous component at a time. We will need the following easy observations.

Lemma 4.4. *Let $\mathbb{F}$ be a field of characteristic zero or larger than d. Let $f \in \mathbb{F}[\mathbf{z}]$ be a polynomial of degree d, and for every $i \in \mathbb{Z}_{\geq 0}$, Δ_i be the differential form of order i as defined in Definition 4.1. Then, the following are true.*

- *For each $i \in \mathbb{Z}_{\geq 0}$, $\Delta_i(f)$ is homogeneous in $\mathbf{z}$ and has degree i in the $\mathbf{z}$ variables. Moreover, for any monomial $\mathbf{z}^{\mathbf{e}}$ of degree i, its coefficient in $\Delta_i(f)$ equals $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}}$.*
- *If f is a homogeneous polynomial, then, for every $i \leq d$, f can be uniquely recovered from all its partial derivatives of order i. As a consequence, for any homogeneous f, given the formal polynomial $\Delta_i(f)$, we can recover f.*

Proof. The first item follows directly from the definition of Δ in Definition 4.1.

The second item follows from an immediate generalization of the following well known observation of Euler to Hasse derivatives. For any homogeneous polynomial f of degree d,

$$d \cdot f = \sum_i x_i \cdot \frac{\partial f(\mathbf{x})}{\partial x_i}\,.$$

We also have that

$$\frac{\partial}{\partial \mathbf{x}^{\mathbf{e}}} \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}'}} = \binom{\mathbf{e}+\mathbf{e}'}{\mathbf{e}} \cdot \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}+\mathbf{e}'}}\,.$$

Using this we can compute the first order Hasse derivatives of $\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}'}}$ for all $\|\mathbf{e}'\|_1 = i-1$ from $\Delta_i(f)$. So, for any i, given all Hasse derivatives of degree i, we can recover Hasse derivatives of degree $i-1$ (using Euler's formula), and so on, till we recover f. □

Remark 4.5. *We remark that the second item in Lemma 4.4 is false for fields of small characteristic. For instance, if the characteristic is smaller than d, then even for a non-zero f, all its first order derivatives could be zero, and hence f cannot be recovered from its first order derivatives.* ⌟

The following lemma shows that under very mild conditions on $Q(\mathbf{x}, \mathbf{y})$, we can (efficiently) recover all polynomials f of degree at most d such that $Q(\mathbf{x}, \Delta_{<m}(f)) \equiv 0$. This will complete all the ingredients needed for the proof of Theorem 1.1.

Lemma 4.6. *Let $\mathbb{F}$ be a finite field of characteristic larger than d and let $Q(\mathbf{x}, \mathbf{y}) = Q_1 y_1 + \cdots + Q_m y_m$ be any non-zero polynomial in $\mathbb{F}[\mathbf{z}, \mathbf{x}, \mathbf{y}]$ where, $\deg_{\mathbf{x}}(Q) \leq D + d$, $\deg_{\mathbf{z}}(Q) \leq \Gamma$ and Q_i does not depend on $\mathbf{y}$. Then, there is a deterministic algorithm that outputs all polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d such that*

$$Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

Moreover, the algorithm requires at most $\mathrm{poly}\left(D^k, d^k, |\mathbb{F}|^{\binom{m+k}{m}}, \Gamma^k\right)$ *arithmetic operations over the underlying field $\mathbb{F}$.*

Proof. We will reconstruct f iteratively, one homogeneous component at a time. This iterative process has to be started by fixing the homogeneous components of f of degree at most m, and as will be evident from the discussion ahead, every fixing of this initial seed can be lifted to a unique f of degree at most d satisfying

$$Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

Before starting the reconstruction, we need to ensure appropriate non-degeneracy conditions which are typical in iterative reconstruction arguments of this kind.

Preprocessing. We know from the hypothesis of the lemma that Q depends on at least one y variable. Let j be the largest index in $\{1, \ldots, m\}$ such that Q depends on y_j, i.e., Q_j is non-zero and Q_i is identically zero for all $i > j$. For the ease of notation, we shall assume that $j = m$, thus, Q_m is a non-zero polynomial. Recall that f is a polynomial in $\mathbb{F}[\mathbf{x}]$ and each $\Delta_i(f)$ is a polynomial in $\mathbb{F}[\mathbf{x}, \mathbf{z}]$.

Since $Q_m(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ is a non-zero polynomial, there is an $\mathbf{a} \in \mathbb{F}^k$ such that $Q_m(\mathbf{a}) \neq 0$.[5] Replacing the variable x_i by $x_i' + a_i$ (i.e., translating the origin), we can ensure that in this translated coordinate system, $Q_m(\mathbf{x}' + \mathbf{a})$ is non-zero at the origin, i.e., when $\mathbf{x}'$ is set to $\mathbf{0}$. We work in this translated coordinate system for the ease of notation. Observe that every solution $f(\mathbf{x}) \in \mathbb{F}[\mathbf{x}]$ is bijectively mapped to a solution $\tilde{f}(\mathbf{x}') = f(\mathbf{x}' + \mathbf{a}) \in \mathbb{F}[\mathbf{x}']$ and given $\tilde{f}$, we can efficiently recover f. Also, note that $\Delta_i(f)(\mathbf{x}' + \mathbf{a}) = \Delta_i(f(\mathbf{x}' + \mathbf{a})) = \Delta_i(\tilde{f}(\mathbf{x}))$, i.e., taking derivatives and then setting $\mathbf{x} = \mathbf{x}' + \mathbf{a}$ is equivalent to first doing the translation $\mathbf{x} = \mathbf{x}' + \mathbf{a}$ and then taking derivatives. Let

$$Q'(\mathbf{x}') := Q(\mathbf{x}' + \mathbf{a}) = Q_1(\mathbf{x}' + \mathbf{a}) y_1 + \cdots + Q_m(\mathbf{x}' + \mathbf{a}) y_m\,,$$

[5] This is assuming $\mathbb{F}$ is large enough, else we can find such an $\mathbf{a}$ in a large enough extension field of $\mathbb{F}$.

and let $I = \langle x'_1, \ldots, x'_k \rangle$ be the ideal generated by $\{x'_1, \ldots, x'_k\}$.

Iterative Reconstruction. We are now ready to describe the iterative reconstruction of $\tilde{f}$.

- **Base Case :** We will try all possible values for the coefficients of monomials of degree at most m in $\tilde{f}$ from the field $\mathbb{F}$. There are $|\mathbb{F}|^{\binom{m+k}{k}}$ possible choices. The next steps are going to uniquely lift each of these candidate solutions to a degree d polynomial, so the number of solutions remains $|\mathbb{F}|^{\binom{m+k}{k}}$.
- **Induction Step :** We now assume that we have recovered $\tilde{f}_0, \tilde{f}_1, \ldots, \tilde{f}_t \in \mathbb{F}[\mathbf{x}']$ for some $t \geq m$, where $\tilde{f}_i$ is a homogeneous component of $\tilde{f}$ of degree i. The goal is to recover $\tilde{f}_{t+1}$, the $(t+1)$-st homogeneous component. Let $\tilde{f}_{\leq t} = \tilde{f}_0 + \tilde{f}_1 + \cdots + \tilde{f}_t$. Now, let us consider the equation $Q'(\mathbf{x}', \Delta_0(\tilde{f}), \Delta_1(\tilde{f}), \ldots, \Delta_{m-1}(\tilde{f})) = 0$ when we work modulo the ideal I^{t-m+3}. Clearly, the homogeneous components of $\tilde{f}$ of degree larger than $t+1$ do not contribute anything modulo I^{t-m+3}, and so we have,

  $$Q'(\mathbf{x}', \Delta_0(\tilde{f}_{\leq t}), \Delta_1(\tilde{f}_{\leq t}), \ldots, \Delta_{m-1}(\tilde{f}_{\leq t} + \tilde{f}_{t+1})) = 0 \mod I^{t-m+3}.$$

  Using the linearity of Δ_i and the fact that Q' is linear in $\mathbf{y}$, we get

  $$Q'(\mathbf{x}', \Delta_0(\tilde{f}_{\leq t}), \Delta_1(\tilde{f}_{\leq t}), \ldots, \Delta_{m-1}(\tilde{f}_{\leq t})) + Q_m(\mathbf{x}' + \mathbf{a}) \cdot \Delta_{m-1}(\tilde{f}_{t+1}) = 0 \mod I^{t-m+3}.$$

  We know that the degree of $\Delta_{m-1}(\tilde{f}_{t+1})$ equals $t+1-(m-1) = t-m+2$, and it is homogeneous in $\mathbf{x}'$. Also, we have ensured in the preprocessing phase that $Q_m(\mathbf{x}' + \mathbf{a}) \mod I = Q_m(\mathbf{a})$ is some non-zero constant $\mathbb{F}$. Thus, this is a non-trivial linear equation in $\Delta_{m-1}(\tilde{f}_{t+1})$ and if we can use it to recover all the partial derivatives of $\tilde{f}_{t+1}$ of order $m-1$, we can then use Lemma 4.4 to recover $\tilde{f}_{t+1}$ itself. We elaborate on the details of this step of recovering the partial derivatives of $\tilde{f}_{t+1}$ from $\Delta_{m-1}(f_{t+1})$ next.

Recovering partial derivatives of $\tilde{f}_{t+1}$ from $\Delta_{m-1}(\tilde{f}_{t+1})$. Recall that since $\tilde{f}_{t+1}$ is a homogeneous polynomial in $\mathbb{F}[\mathbf{x}]$ of degree $t+1$, each of its partial derivatives of order $m-1$ is a homogeneous polynomial in $\mathbf{x}'$ of degree equal to $t+1-(m-1) = t-m+2$. Thus,

$$\Delta_{m-1}(\tilde{f}_{t+1}) := \sum_{\mathbf{e}:\|\mathbf{e}\|_1 = m-1} \mathbf{z}^{\mathbf{e}} \cdot \frac{\partial \tilde{f}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^{\mathbf{e}}}.$$

is a homogeneous polynomial in both $\mathbf{z}$ and $\mathbf{x}'$, with degree $m-1$ in $\mathbf{z}$ and degree $t-m+2$ in $\mathbf{x}'$. Our goal is to recover the coefficients of all monomials $\mathbf{z}^{\mathbf{e}}$ of degree $m-1$ in $\mathbf{z}$ when viewing

$\Delta_{m-1}(\tilde{f}_{t+1})$ as a polynomial in $\mathbb{F}[\mathbf{x}][\mathbf{z}]$, and we have access to the expression

$$Q'(\mathbf{x}', \Delta_0(\tilde{f}_{\le t}), \Delta_1(\tilde{f}_{\le t}), \ldots, \Delta_{m-1}(\tilde{f}_{\le t})) = -Q_m(\mathbf{a})\Delta_{m-1}(\tilde{f}_{t+1}) \mod I^{t-m+3}.$$

As a first step, observe that the polynomial $Q_m(\mathbf{a})\Delta_{m-1}(\tilde{f}_{t+1})$ has degree at most $\Gamma + m - 1$ in $\mathbf{z}$ and degree exactly $t - m + 2$ in $\mathbf{x}'$. Moreover, since $Q_m(\mathbf{a}) \in \mathbb{F}[\mathbf{z}]$ is non-zero, the polynomials $\{Q_m(\mathbf{a})\mathbf{z}^{\mathbf{e}} : \deg(\mathbf{z}^{\mathbf{e}}) = m - 1\}$ are linearly independent as polynomials of degree at most $\Gamma + (m-1)$ in $\mathbf{z}$ over the field over the field $\mathbb{F}$. Therefore, for any hitting set $H \subseteq \mathbb{F}^k$ for k-variate polynomials of degree at most $\Gamma + (m-1)$, the evaluation vectors $\mathsf{Eval}_H(Q_m(\mathbf{a})\mathbf{z}^{\mathbf{e}})$ of these polynomials on H are linearly independent over $\mathbb{F}$. So, for every $\mathbf{x}'^{\mathbf{e}_0}$ of degree $m-1$, there exists an $\mathbb{F}$ linear combination of the polynomials $\{Q_m(\mathbf{a})\Delta_{m-1}(\tilde{f}_{t+1})\mathbf{b} : \mathbf{b} \in H\}$ which equals $\frac{\partial \tilde{f}_{t+1}(\mathbf{x}')}{\partial \mathbf{x}'^{\mathbf{e}}}$. Moreover, such a linear combination can be found (e.g. via Gaussian Elimination over the field $\mathbb{F}$) efficiently in the size of this linear system.

Thus, to recover the partial derivatives of order $m-1$ of $\tilde{f}_{t+1}$ given a monomial representation of $Q_m(\mathbf{a})\Delta_{m-1}(\tilde{f}_{t+1})$, we consider the hitting set H of size $O(\Gamma \cdot m)^k$ for k-variate degree $\Gamma(m-1)$ polynomials given by Lemma 3.4, compute the evaluation of the polynomials

$$Q_m(\mathbf{a}) \cdot \Delta_{m-1}(\tilde{f}_{t+1}) = \sum_{\mathbf{e}:\|\mathbf{e}\|_1 = m-1} Q_m(\mathbf{a})\mathbf{z}^{\mathbf{e}} \cdot \frac{\partial \tilde{f}_{t+1}(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}},$$

at every $\mathbf{b} \in H$, and take appropriate weighted linear combinations to recover each of the partial derivatives $\frac{\partial \tilde{f}_{t+1}(\mathbf{x})}{\partial \mathbf{x}^{\mathbf{e}}}$.

Since $Q'(\mathbf{x}', \Delta_0(\tilde{f}_{\le t}), \Delta_1(\tilde{f}_{\le t}), \ldots, \Delta_{m-1}(\tilde{f}_{\le t}))$ is a polynomial of degree at most $\Gamma + m$ in $\mathbf{z}$ and at most $D + d$ in $\mathbf{x}$, we can do the evaluations by writing the coefficient vector of this polynomial in time $\mathrm{poly}(D^k, d^k, \Gamma^k, m^k)$ and doing evaluations one monomial at a time.

The running time. Observe that we can go from the original polynomial Q to the polynomial Q' by finding an appropriate $\mathbf{a}$ deterministically in time at most $(D+d)^k$ by just querying all points on a large enough grid in $\mathbb{F}^k$ (or a grid in an extension field of $\mathbb{F}$, if $\mathbb{F}$ isn't large enough). This follows from Lemma 3.4.

Once we have Q', we reconstruct f in d iterations, so it suffices to estimate the cost of each iteration. As we just argued in the earlier part of the proof, every iteration just involves evaluating the polynomial $Q'(\mathbf{x}', \Delta_0(\tilde{f}_{\le t}), \Delta_1(\tilde{f}_{\le t}), \ldots, \Delta_{m-1}(\tilde{f}_{\le t}))$ at a hitting set H of size at most $\mathrm{poly}(\Gamma^k, m^k)$ and solving about m^k linear systems of the same size. The straightforward implementation of this takes no more than $\mathrm{poly}(D^k, d^k, \Gamma^k, m^k)$ field operations. □

As is evident from the proof of Lemma 4.6, the following more structured version of Lemma 4.6 is true.

Lemma 4.7. *Let $\mathbb{F}$ be a field of characteristic zero or larger than d and let $Q(\mathbf{x}, \mathbf{y}) = Q_1 y_1 + \cdots + Q_m y_m$ be any polynomial in $\mathbb{F}[\mathbf{z}, \mathbf{x}, \mathbf{y}]$ where, $\deg_{\mathbf{x}}(Q) \leq D + d$, $\deg_{\mathbf{z}}(Q) \leq \Gamma$ and Q_i's do not depend on $\mathbf{y}$. Then, there is a deterministic algorithm that outputs a linear space of polynomials in $\mathbb{F}[\mathbf{x}]$ of dimension at most $\binom{m+k}{k}$ over $\mathbb{F}$ which contains all polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d such that*

$$Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

Moreover, the algorithm requires at most $\text{poly}\left(D^k, d^k, \Gamma^k\right)$ *arithmetic operations over the underlying field $\mathbb{F}$.*

To bound the true running time of the algorithm in Lemma 4.7, we need to add a $\text{poly}(\log \mathbb{F})$ factor in the the upper bound on the field operations for finite fields and a polynomial factor in the bit complexity of the input over the field of rational numbers. While working over rationals, we might need a bit more care to solve the linear systems appearing in the proof of Lemma 4.6 efficiently, since the naive implementation of Gaussian Elimination might blow up the bit complexity of the numbers appearing at various intermediate stages.

4.6 Putting things together

We are now ready to prove Theorem 1.1.

Proof of Theorem 1.1. We start by setting the parameters. ε and k are fixed apriori, and we choose s, m such that $s = m^2$ and m is large enough so that

$$\frac{10}{m^{1/k}} + \frac{m}{s-m} < \varepsilon\,.$$

With this choice of parameters, we use Lemma 4.2 to construct a non-zero polynomial Q which *explains* the received word P. Then, we use Lemma 4.6 to find all polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d such that

$$Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

We know, from Lemma 4.6 that the number of such solutions is upper bounded by $|\mathbb{F}|^{\binom{m+k}{k}}$ and from Lemma 4.3 that every polynomial f of degree at most d in $\mathbb{F}[\mathbf{x}]$ such that $\text{Dist}(\text{Enc}(f), P)$ is at most $(1-\delta) - \varepsilon$, where $\delta = 1 - d/(s|S|)$ satisfies the equation

$$Q(\mathbf{x}, \Delta_0(f), \Delta_1(f), \ldots, \Delta_{m-1}(f)) \equiv 0\,.$$

Thus, all such polynomials f are included in the list of outputs. The running time of the algorithm immediately follows from the running time guarantees in Lemma 4.2 and Lemma 4.6. □

4.7 Another view of the algorithm

We now discuss an alternative description of the decoding algorithm. In essence, this is just a rewording of the previous algorithm, but appears to have some qualitative advantages. For instance, the description itself seems simpler here as we don't need to introduce the $\mathbf{z}$ variables, but instead, end up working with a system of equations over the original field $\mathbb{F}$ itself. Moreover, the runtime analysis of the algorithm gives a slightly better bound of $\mathrm{poly}(|S|^k, d^k)$ on the number of field operations needed by the decoding algorithm as opposed to the bound of $\mathrm{poly}(|S|^{k^2}, d^{k^2})$ that is claimed in Theorem 1.1.

Given the received word $P : S^k \to \mathbb{F}^{\binom{s+k-1}{k}}$, we assume that the coordinates of $\mathbb{F}^{\binom{s+k-1}{k}}$ are indexed by k-variate monomials of degree at most $s-1$. Let $t = 10m^{k+1}$ and for each $i \in [s]$ and $j \in [t]$, let $\mathbf{a}_{i,j} \in \mathbb{F}^{\binom{i-2+k}{k}}$ be vectors such that for every i, the dimension of the space spanned by $\{\mathbf{a}_{i,1}, \mathbf{a}_{i,2}, \ldots, \mathbf{a}_{i,t}\}$ over $\mathbb{F}$ equals $\binom{i-2+k}{k}$. Again we think of the coordinates of $\mathbf{a}_{i,j}$ as being indexed by k-variate monomials of degree equal to $i-1$.

Now, from P, we construct $P_1, P_2, \ldots, P_t$ where each P_j is a function S^k to $\mathbb{F}^s$, such that for every $\mathbf{b} \in S^k$, the i^{th} coordinate of $P_j(\mathbf{b})$ equals the weighted linear combination of the coordinates of $P(\mathbf{b})$ indexed by monomials of degree exactly $i-1$, with weights according to $\mathbf{a}_{i,j}$. In other words, the i^{th} coordinate of $P_j(\mathbf{b})$ equals

$$\sum_{\mathbf{e} \in \mathbb{Z}^k_{\geq 0}, \|\mathbf{e}\|_1 = i-1} \mathbf{a}_{i,j}(\mathbf{e}) \cdot P(\mathbf{b})_{\mathbf{e}} \,,$$

where $P(\mathbf{b})_{\mathbf{e}}$ is the coordinate of $P(\mathbf{b})$ indexed by $\mathbf{e}$. Now, for the interpolation step, for each $j \in [t]$, we find a polynomial $\tilde{Q}_j = \sum_{i=1}^{m} \tilde{Q}_{i,j}(\mathbf{x}) y_j$ of not too high degree which explains P_j in the sense of Lemma 4.2. Note that each $\tilde{Q}_j$ is now a polynomial over the original field $\mathbb{F}$. An immediate instantiation of Lemma 4.3 for this setting shows that if $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d and $\mathsf{Enc}(f)$ and P are close enough, then for every $j \in [t]$, $\tilde{Q}_j(\mathbf{x}, \Psi_j(f))$ must be identically zero, where $\Psi_j(f) = \left(\Psi_{j,1}(f), \ldots, \Psi_{j,m}(f)\right)$ is defined as

$$\Psi_{j,i}(f) = \sum_{\mathbf{e} \in \mathbb{Z}^k_{\geq}, \|\mathbf{e}\|_1 = i-1} \mathbf{a}_{i,j}(\mathbf{e}) \cdot \frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}} \,.$$

Before going to the reconstruction step, we note that it might be the case that $\tilde{Q}_1, \tilde{Q}_2, \ldots, \tilde{Q}_t$ depend on different subsets of $\mathbf{y}$ variables. But since $t > m^{k+1}$, by averaging, it follows that there exist an $\ell \in [m]$ such that at least m^k of the polynomials $\{\tilde{Q}_j : j \in [t]\}$ have the property that they depend on y_ℓ and do not depend on $y_{\ell'}$ for any $\ell' > \ell$. For the ease of notation, let us assume that $\tilde{Q}_1, \tilde{Q}_2, \ldots, \tilde{Q}_{t'}$ depend on y_m, where $t' = m^k$.

Now, to recover f, we solve the equations $\tilde{Q}_j(\mathbf{x}, \Psi_j(f)) \equiv 0$ for all $j \in [t']$. We solve for f one homogeneous component as in the proof of Lemma 4.6. Assuming that we have recovered

homogeneous components of degree at most u of f, we can follow the proof of Lemma 4.6 to recover $\Psi_{j,m}(f_{u+1})$ for every $j \in [t']$, where f_{u+1} is the homogeneous component of f of degree $u+1$.[6] At this point, the choice of the vectors $\mathbf{a}_{i,j}$, the definition of $\Psi_{j,m}(f_{u+1})$ and the fact that $t' \geq m^k$, we get that we have sufficiently many linearly independent homogeneous linear equations in all the partial derivatives of f_{u+1} of order $(m-1)$. Thus, we can solve this linear system to recover each of these partial derivatives and combine them according to Lemma 4.4 to obtain f_{u+1}, and proceed to the next step. Moreover, as in Lemma 4.6, if we start from the correct coefficients of f in the base case of this process, each of the subsequent steps are unique.

Thus, instead of working with a single polynomial equation as in a standard application of the polynomial method, this algorithm proceeds via working simultaneously with many equations.

We now remark on the running time.

Remark 4.8. *We note that in algorithm sketched above, the number of field operations needed is upper bounded by* $\mathrm{poly}(|S|^k, d^k)$. *This follows from the observation that in this algorithm we are essentially solving* $m^k < d^k$ *linear systems of size* $\mathrm{poly}(|S|^k, d^k)$ *over the underlying field* $\mathbb{F}$ *to recover all codewords close to the received word.* ⌟

5 Reducing the list size to a constant

In this section, we use the pruning algorithm due to Kopparty, Ron-Zewi, Saraf and Wootters [KRSW18] together with Lemma 4.7 to obtain a shorter list of correct polynomials, thereby improving the bound on the list size in Theorem 1.1 from a polynomial (in the input size) to an absolute constant depending only on the parameter ε and dimension k. This would complete the proof of Theorem 1.3. The first step towards the goal of recovering codewords from a small linear space is the following theorem, which is a natural multivariate analog of [GK16, Theorem 17] in the work of Guruswami and Kopparty [GK16]. Our proof is essentially the same, apart from the fact that we are in the multivariate setting and hence have to work with Generalized Wronskians matrices.

Theorem 5.1 (subspace restrictions)**.** *Let* $\mathbb{F}$ *be a field of characteristic zero or larger than* d. *Let* $\mu \geq w \in \mathbb{N}$ *be parameters and let* $W \subseteq \mathbb{F}[\mathbf{x}]$ *be an* $\mathbb{F}$*-linear subspace of* k*-variate polynomials of degree at most* d, *such that dimension of* W *is at most* w. *For any* $\mathbf{a} \in \mathbb{F}^k$, *let* $H_{\mathbf{a}}$ *be the* $\mathbb{F}$*-linear space of polynomials of degree at most* d *which vanish with multiplicity at least* μ *at* $\mathbf{a}$. *Then, for every set* $S \subseteq \mathbb{F}$, *we have,*

$$\sum_{\mathbf{a} \in S^k} \dim(H_{\mathbf{a}} \cap W) \leq \frac{dw|S|^{k-1}}{(\mu - w + 1)}.$$

We use this statement in our proof in this section, and prove it in Section 6.

[6] We might have to do an initial translation of coordinates as in the proof of Lemma 4.6.

5.1 The pruning algorithm

The input to this algorithm is a received word P and a linear subspace W of polynomials of degree at most d in $\mathbb{F}[\mathbf{x}]$ of dimension at most w. The goal is to output all polynomials in $f \in W$ such that $\mathsf{Enc}_{s,S}(f)$ agrees with P on at least $\alpha = \delta + \varepsilon$ locations. The description of the algorithm has a parameter r, which we later set to an appropriate value.

Algorithm A

1. Choose $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_r$ independently and uniformly at random from S^k.
2. If there is a unique polynomial $f \in W$ such that $\mathsf{Enc}_{s,S}(f)$ and P agree on each of $\mathbf{a}_1, \ldots, \mathbf{a}_r$, then output f.

Clearly, the second step of the algorithm can be implemented efficiently via Gaussian Elimination.

The final pruning algorithm invokes Algorithm A multiple times and outputs the union of all the lists. In the rest of this section, we show that with high probability, this will output the list of all codewords close to the received word that are contained in the input linear space.

The algorithm and the analysis is precisely the same as that in the work of Kopparty, Ron-Zewi, Saraf and Wootters [KRSW18], apart from the fact that we invoke it for multivariate multiplicity codes, whereas in [KRSW18] it was designed for folded Reed Solomon Codes and univariate multiplicity codes. We briefly sketch some of the details in the rest of this section. For brevity, we again use $\mathsf{Enc}()$ for $\mathsf{Enc}_{s,S}()$. We also assume that the dimension w of W is less than the multiplicity parameter s of the code.

Lemma 5.2 (Analogous to [KRSW18, Lemma IV.5 (conference version)])**.** *For any polynomial $f \in W$ such that $\mathsf{Dist}(\mathsf{Enc}(f), P) < \alpha$, f is output by Algorithm A with probability at least*

$$(1-\alpha)^r - w\left(\frac{d}{|S|(s-w)}\right)^r.$$

Moreover, Algorithm A runs in polynomial time in the input size.

Proof Sketch. The proof of the lemma is precisely the same as that of [KRSW18, Lemma IV.5 (conference version)] except we use Theorem 5.1 as opposed to an analogous statement for folded Reed Solomon codes. □

We are now ready to prove Theorem 1.3.

Proof of Theorem 1.3. Given the error parameter ε and the number of variables k, we choose s, m as follows.

- $m = \left(\frac{20}{\varepsilon}\right)^k$,

- $s = \frac{4}{\varepsilon} \cdot \binom{m+k}{k}$.

We note that for this choice of parameters, $\frac{m}{s-m} < \frac{\varepsilon}{2}$ and hence,

$$\frac{10}{m^{1/k}} + \frac{m}{s-m} < \varepsilon,$$

as is needed to invoke Lemma 4.2. We now use Lemma 4.2 to construct the polynomial Q which *explains* the received word P, and Lemma 4.7 to give us a subspace W of polynomials in $\mathbb{F}[\mathbf{x}]$ of dimension at most $w = \binom{m+k}{k}$ over $\mathbb{F}$, that contains all polynomials $f \in \mathbb{F}[\mathbf{x}]$ of degree at most d such that $\mathsf{Dist}(\mathsf{Enc}(f), P) < (\delta - \varepsilon)$, where $\delta = 1 - d/(s|S|)$ is the relative distance of the code. Let the parameter r be set as

$$r = \frac{\log(2 \cdot \binom{m+k}{k})}{\log(1+\varepsilon/4)} \leq O\left(\frac{k^2 \log 1/\varepsilon}{\varepsilon}\right).$$

We now instantiate Lemma 5.2 with inputs being the received word P, the subspace W of dimension at most $w = \binom{m+k}{k}$ and the parameter r as set above.

A single run of Algorithm A returns at most one polynomial f in W such that $\mathsf{Dist}(\mathsf{Enc}(f), P) < (\delta - \varepsilon)$. Moreover, every such f is output with probability at least

$$\rho = (1 - \delta + \varepsilon)^r - w\left(\frac{d}{|S|(s-w)}\right)^r.$$

To simplify this, we note that from the choice of parameters

$$\begin{aligned}
w\left(\frac{d}{|S|(s-w)}\right)^r &= \binom{m+k}{k}\left(\frac{s}{(s-w)} \cdot (1-\delta)\right)^r \\
&\leq \frac{1}{2} \cdot (1+\varepsilon/4)^r \left(\frac{1}{(1-\varepsilon/4)} \cdot (1-\delta)\right)^r \qquad \text{[plugging in the values of } s, r\text{]} \\
&\leq \frac{1}{2} \cdot \left(\frac{1+\varepsilon/4}{1-\varepsilon/4} \cdot (1-\delta)\right)^r \\
&\leq \frac{1}{2} \cdot (1-\delta+\varepsilon)^r,
\end{aligned}$$

where the last inequality follows from the fact that $\frac{1+\varepsilon/4}{1-\varepsilon/4} \cdot (1-\delta) \leq (1-\delta+\varepsilon)$, whenever $1+\delta-\varepsilon/2 > 0$, which is always true in our setting, since $\delta, \varepsilon \in (0,1)$. Thus, we get

$$\rho \geq \frac{1}{2}(1-\delta+\varepsilon)^r.$$

Hence, the number of polynomials in the space W such that $\mathsf{Dist}(\mathsf{Enc}(f), P) < (\delta - \varepsilon)$ is at most $\frac{1}{\rho} = \frac{2}{(1-\delta+\varepsilon)^r}$.

It follows from a union bound that if we run Algorithm A about $O\left(\frac{1}{\rho}\cdot\log\frac{1}{\rho}\right)$ times with fresh randomness each time, and output every polynomial obtained, with high probability, we would have output all the polynomials f in W with $\mathsf{Dist}(\mathsf{Enc}(f),P)<(\delta-\varepsilon)$. Thus the number of runs of Algorithm A is

$$O\left(\frac{1}{\rho}\cdot\log\frac{1}{\rho}\right)=O\left(\frac{r\log(\frac{1}{1-\delta+\varepsilon})}{(1-\delta+\varepsilon)^r}\right)\leq\exp\left(O\left(\frac{k^2}{\varepsilon}\log^3\frac{1}{\varepsilon}\right)\right).$$

The upper bound on the running time immediately follows from the running time guarantees in Lemma 4.2, Lemma 4.7 and the final pruning that happens in the process of recovering the relevant codewords from the subspace output by Lemma 4.7. □

6 Subspace restrictions of multivariate multiplicity codes

In this section, we prove Theorem 5.1. For the proof, we follow the outline of Guruswami and Kopparty [GK16] and essentially observe that (almost) everything works even for multivariate polynomials. The only difference is that instead of the Wronskian criterion for univariate polynomial, we need to work with the following generalized Wronskian criterion for multivariate polynomials.

Theorem 6.1 (generalized Wronskian criterion)**.** *Let $f_1,f_2,\ldots,f_w\in\mathbb{F}[\mathbf{x}]$ be k-variate polynomials of maximum individual degree at most d. If the characteristic of $\mathbb{F}$ is zero or larger than d, then the following is true. $f_1,f_2,\ldots,f_w$ are linearly independent over $\mathbb{F}$ if and only if there exist monomials $\mathbf{x}^{\mathbf{e}_1},\mathbf{x}^{\mathbf{e}_2},\ldots,\mathbf{x}^{\mathbf{e}_w}$ such that for every $i\in[w]$, $\deg(\mathbf{x}^{\mathbf{e}_i})\leq i-1$, and the $w\times w$ matrix $M_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}$ whose (i,j) entry equals $\frac{\partial f_j}{\partial\mathbf{x}^{\mathbf{e}_i}}$ is full rank over the field $\mathbb{F}(\mathbf{x})$.*

The classical Wronskian criterion (and its generalized counterpart) are typically proved for fields of characteristic zero and with the usual notion of partial derivatives (cf., Bostan and Dumas [BD10, Theorem 3]). These proofs extend to the above setting. For the sake of completeness, we provide an alternative proof of the above theorem in Appendix B.

Equipped with this criterion, we are now ready to prove Theorem 5.1

Proof of Theorem 5.1. Let $f_1,f_2,\ldots,f_w\in W$ be linearly independent polynomials of degree at most d which span W. Let E be a subset of μ-tuples of monomials defined as follows.

$$E:=\left\{(\mathbf{x}^{\mathbf{e}_1},\mathbf{x}^{\mathbf{e}_2},\ldots,\mathbf{x}^{\mathbf{e}_\mu}):\deg(\mathbf{x}^{\mathbf{e}_i})\leq i-1\right\}.$$

For every $\psi = (\mathbf{x}^{\mathbf{e}_1}, \mathbf{x}^{\mathbf{e}_2}, \ldots, \mathbf{x}^{\mathbf{e}_\mu})$ in E, let $M_\psi \in \mathbb{F}[\mathbf{x}]^{\mu \times w}$ matrix defined as follows.

$$M_\psi := \begin{bmatrix} \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_1}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_1}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_1}} \\ \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_2}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_2}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_2}} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_\mu}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_\mu}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_\mu}} \end{bmatrix}.$$

And, let $\tilde{M}_\psi$ denote the $w \times w$ submatrix of M_ψ by taking the first w rows and columns, i.e.,

$$\tilde{M}_\psi := \begin{bmatrix} \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_1}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_1}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_1}} \\ \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_2}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_2}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_2}} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_1}{\partial \mathbf{x}^{\mathbf{e}_w}} & \frac{\partial f_2}{\partial \mathbf{x}^{\mathbf{e}_w}} & \cdots & \frac{\partial f_w}{\partial \mathbf{x}^{\mathbf{e}_w}} \end{bmatrix}.$$

From Theorem 6.1, we know that there exists ψ_0 in E such that $\tilde{M}_{\psi_0}$ (and hence, M_{ψ_0}) is full rank over $\mathbb{F}(\mathbf{x})$. Let L_{ψ_0} denote the determinant of $\tilde{M}_{\psi_0}$. Clearly, L_{ψ_0} is a non-zero k-variate polynomial of degree at most dw. We note that for many choices of $\psi \in E$, the corresponding matrix M_ψ could be of rank less than w. Perhaps somewhat surprisingly, all these matrices play a role in the proof. The proof essentially follows from the following claim.

Claim 6.2. *For every $\mathbf{a} \in \mathbb{F}^k$, the multiplicity of $L_{\psi_0}(\mathbf{x})$ at $\mathbf{a}$ is at least $(\mu - w + 1)\dim(H_\mathbf{a} \cap W)$.*

We first complete the proof of the theorem using the above claim and then prove the claim. From Claim 6.2, we get

$$\sum_{\mathbf{a} \in S^k} (\mu - w + 1)\dim(H_\mathbf{a} \cap W) \leq \sum_{\mathbf{a} \in S^k} \mathrm{mult}(L(\mathbf{x}), \mathbf{a}).$$

From the earlier discussion, L_{ψ_0} is a non-zero polynomial of degree at most dw. Thus, by Lemma 3.4, the quantity $\sum_{\mathbf{a} \in S^k} \mathrm{mult}(L(\mathbf{x}), \mathbf{a})$ is upper bounded by $dw|S|^{k-1}$, and this completes the proof of Theorem 5.1. □

We now prove the claim. For this, we need the following claim.

Claim 6.3. *For every $\psi \in E$, and for every $\mathbf{a} \in \mathbb{F}^k$,*

$$\mathrm{rank}(M_\psi(\mathbf{a})) \leq w - \dim(H_\mathbf{a} \cap W).$$

Proof of Claim 6.3. We just follow the definition.

$$\begin{aligned}
\dim(H_{\mathbf{a}} \cap W) &= \dim\left(\left\{\mathbf{b} = (b_1, b_2, \ldots, b_w) \in \mathbb{F}^w : \mathrm{mult}(\sum_{i=1}^{w} b_i f_i, \mathbf{a}) \geq \mu\right\}\right) \\
&= \dim\left(\left\{\mathbf{b} = (b_1, b_2, \ldots, b_w) \in \mathbb{F}^w : \forall \mathbf{x}^{\mathbf{e}} \text{ s.t } \deg(\mathbf{x}^{\mathbf{e}}) < \mu, \sum_{i=1}^{w} b_i \frac{\partial f_i}{\partial \mathbf{x}^{\mathbf{e}}}(\mathbf{a}) = 0\right\}\right) \\
&= \dim\left(\left\{\mathbf{b} = (b_1, b_2, \ldots, b_w) \in \mathbb{F}^w : \forall \psi \in E, (M_\psi(\mathbf{a}))\mathbf{b} = \mathbf{0}\right\}\right) \\
&\leq \min_{\psi \in E}(\dim(\mathrm{Kernel}(M_\psi(\mathbf{a})))) \\
&\leq \min_{\psi \in E}(w - \mathrm{rank}(M_\psi(\mathbf{a}))) \ .
\end{aligned}$$

□

Proof of Claim 6.2. To show the claim, we show that for every monomial $\mathbf{x}^{\mathbf{f}}$ of degree less than $(\mu - w + 1)\dim(H_{\mathbf{a}} \cap W)$, the Hasse derivative $\frac{\partial L_{\psi_0}}{\partial \mathbf{x}^{\mathbf{f}}}$ is zero at $\mathbf{a}$. Let $\psi_0 = (\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_w)$. Then, we have (using Proposition 3.2: Items 4 and 5).

$$\frac{\partial L_{\psi_0}}{\partial \mathbf{x}^{\mathbf{f}}}(\mathbf{a}) = \sum_{\mathbf{u}_1 + \mathbf{u}_2 + \cdots + \mathbf{u}_w = \mathbf{f}} \left(\prod_{j \in [w]} \binom{\mathbf{e}_j + \mathbf{u}_j}{\mathbf{u}_j}\right) \det(\tilde{M}_{(\mathbf{e}_1 + \mathbf{u}_1, \ldots, \mathbf{e}_w + \mathbf{u}_w)})(\mathbf{a}) \, .$$

Now, we know that $\sum_j \|\mathbf{u}_j\|_1 < (\mu - w + 1)\dim(H_{\mathbf{a}} \cap W)$, so there are less than $\dim(H_{\mathbf{a}} \cap W)$ values of $j \in \{1, 2, \ldots, w\}$ such that $\|\mathbf{u}_j\|_1$ is more than $\mu - w$. Moreover, $\|\mathbf{u}_j\|_1 \leq \mu - w$ implies that $\|\mathbf{e}_j\|_1 + \|\mathbf{u}_j\|_1 \leq \mu - 1$. Thus, there is a $\psi \in E$, such that there are more than $w - \dim(H_{\mathbf{a}} \cap W)$ rows of the matrix $\tilde{M}_{(\mathbf{e}_1 + \mathbf{u}_1, \ldots, \mathbf{e}_w + \mathbf{u}_w)}(\mathbf{a})$ which are also rows in the matrix $M_\psi(\mathbf{a})$. But, from Claim 6.3, we know that for every $\psi \in E$, $M_\psi(\mathbf{a})$ has rank at most $w - \dim(H_{\mathbf{a}} \cap W)$. Thus, each of the matrices $\tilde{M}_{(\mathbf{e}_1 + \mathbf{u}_1, \ldots, \mathbf{e}_w + \mathbf{u}_w)}(\mathbf{a})$ in the summand above is rank deficient, and hence has determinant zero. □

Acknowledgments

Madhu, Mrinal, and Prahladh thank Swastik Kopparty for many insightful discussions on multiplicity codes and on the results in [KSY14, Kop15].

References

[BD10] Alin Bostan and Philippe Dumas. *Wronskians and linear independence*. Amer. Math. Monthly, 117(8):722–727, 2010. arXiv:1301.6598. 28

[DKSS13] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. *Extensions to the method of multiplicities, with applications to Kakeya sets and mergers*. SIAM J. Comput., 42(6):2305–2328, 2013. (Preliminary version in *50th FOCS*, 2009). eccc:2009/TR09-004. 2, 12, 13

[DL78] RICHARD A. DEMILLO and RICHARD J. LIPTON. *A probabilistic remark on algebraic program testing*. Inform. Process. Lett., 7(4):193–195, 1978. 2

[GK16] VENKATESAN GURUSWAMI and SWASTIK KOPPARTY. *Explicit subspace designs*. Comb., 36(2):161–185, 2016. (Preliminary version in *54th FOCS*, 2013). eccc:2013/TR13-060. 25, 28

[GKSS19] ZEYU GUO, MRINAL KUMAR, RAMPRASAD SAPTHARISHI, and NOAM SOLOMON. *Derandomization from algebraic hardness: Treading the Borders*. In *Proc. 60th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 147–157. 2019. arXiv:1905.00091, eccc:2019/TR19-065. 8, 10

[GS99] VENKATESAN GURUSWAMI and MADHU SUDAN. *Improved decoding of Reed-Solomon and algebraic-geometry codes*. IEEE Trans. Inform. Theory, 45(6):1757–1767, 1999. (Preliminary version in *39th FOCS*, 1998). eccc:1998/TR98-043. 4

[Gut16] LARRY GUTH. *Polynomial Methods in Combinatorics*, volume 64 of *University Lecture Series*. Amer. Math. Soc., 2016. 2

[GW13] VENKATESAN GURUSWAMI and CAROL WANG. *Linear-algebraic list decoding for variants of Reed-Solomon codes*. IEEE Trans. Inform. Theory, 59(6):3257–3268, 2013. (Preliminary version in *26th IEEE Conference on Computational Complexity*, 2011 and *15th RANDOM*, 2011). eccc:2012/TR12-073. 4, 5, 6, 7, 8, 9, 10

[KK17] JOHN Y. KIM and SWASTIK KOPPARTY. *Decoding Reed-Muller codes over product sets*. Theory Comput., 13(1):1–38, 2017. (Preliminary version in *31st IEEE Conference on Computational Complexity*, 2016). arXiv:1511.07488. 2, 4, 5, 6

[Kop14] SWASTIK KOPPARTY. *Some remarks on multiplicity codes*. In ALEXANDER BARG and OLEG R. MUSIN, eds., *Discrete Geometry and Algebraic Combinatorics*, volume 625 of *Contemporary Mathematics*, pages 155–176. AMS, 2014. arXiv:1505.07547. 6

[Kop15] ———. *List-decoding multiplicity codes*. Theory of Computing, 11:149–182, 2015. eccc:2012/TR12-044. 4, 5, 6, 7, 8, 10, 30

[KRSW18] SWASTIK KOPPARTY, NOGA RON-ZEWI, SHUBHANGI SARAF, and MARY WOOTTERS. *Improved decoding of Folded Reed-Solomon and Multiplicity codes*. In *Proc. 59th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 212–223. 2018. arXiv:1805.01498, eccc:2018/TR18-091. 5, 11, 25, 26

[KSY14] SWASTIK KOPPARTY, SHUBHANGI SARAF, and SERGEY YEKHANIN. *High-rate codes with sublinear-time decoding*. J. ACM, 61(5):28:1–28:20, 2014. (Preliminary version in *43rd STOC*, 2011). eccc:2010/TR10-148. 4, 30

[LN96] RUDOLF LIDL and HARALD NIEDERREITER. *Finite Fields*, volume 2 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2nd edition, 1996. 31

[Nie01] RASMUS REFSLUND NIELSEN. *List decoding of linear block codes*. Ph.D. thesis, Technical University of Denmark, 2001. 4

[Ore22] ØYSTEIN ORE. *Über höhere kongruenzen (German) [About higher congruences]*. Norsk Mat. Forenings Skrifter, 1(7):15, 1922. (see [LN96, Theorem 6.13]). 2

[PW04] RUUD PELLIKAAN and XIN-WEN WU. *List decoding of q-ary Reed-Muller codes*. IEEE Trans. Inform. Theory, 50(4):679–682, 2004. 7

[RT97] M. YU ROSENBLOOM and MICHAEL ANATOLÉVICH TSFASMAN. Коды для m-метрики *(Russian) [Codes for the m-metric]*. Probl. Peredachi Inf., 33(1):55–63, 1997. (English translation in *Problems Inform. Transmission*, 33(1):45–52, 1997). 4

[Sar11] SHUBHANGI SARAF. *The method of multiplicities*. Ph.D. thesis, Massachusetts Institute of Technology, 2011. 2

[Sch80] JACOB T. SCHWARTZ. *Fast probabilistic algorithms for verification of polynomial identities*. J. ACM, 27(4):701–717, October 1980. 2

[Sud97] MADHU SUDAN. *Decoding of Reed-Solomon codes beyond the error-correction bound*. J. Complexity, 13(1):180–193, 1997. (Preliminary version in *37th FOCS*, 1996). 4

[Zip79] RICHARD ZIPPEL. *Probabilistic algorithms for sparse polynomials.* In EDWARD W. NG, ed., *Proc. International Symp. of Symbolic and Algebraic Computation (EUROSAM)*, volume 72 of *LNCS*, pages 216–226. Springer, 1979. 2

A Exponential number of codewords at a distance δ

Let $T \subseteq S$ be an arbitrary subset of size d/s. For a variable x_1, consider the polynomial $f(\mathbf{x}) = \prod_{b \in T}(x_1 - b)^{s-1}$. At every point $\mathbf{a} \in S^k$ such that $a_1 \in T$, $f(\mathbf{x})$ vanishes with multiplicity at least s. Moreover, the set $\{\mathbf{a} \in S^k : a_1 \in T\} \subseteq S^k$ is of size exactly $\frac{d}{s}|S|^{k-1}$. Thus, the encoding of every polynomial in the set

$$\mathcal{M} = \left\{ \prod_{b \in T}(x_1 - b)^{s-1} : \deg(L(\mathbf{x})) = 1, T \subseteq S, |T| = d/s \right\}$$

under the k-variate multiplicity code, with multiplicity parameter s agrees with the encoding of the polynomial 0 on at least $d/(qs)$ fraction of points, i.e., the relative distance between them is $(1 - \delta)$, where δ is the distance of the code. Moreover, the set $\mathcal{M}$ is of size $\binom{q}{d/s}$, which is superpolynomially growing in d. In this sense, the error tolerance of the result in Theorem 1.1 is the best one could hope for (up to the $\varepsilon > 0$ term) if we are hoping for polynomial list size.

B Generalized Wronskian criterion

In this section, we give a proof of the generalized Wronskian criterion in the multivariate setting that works over fields of finite characteristic, and using the notion of Hasse derivatives.

We first state and prove a proposition which we will use to prove Theorem 6.1. Given a sequence $f_1, f_2, \ldots, f_w$ of w k-variate polynomials of individual degree at most d and a sequence

$\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_w$ of w monomials, let $M_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w)$ be the $w \times w$ matrix whose (i,j)-th entry is $\frac{\partial f_j}{\partial \mathbf{x}^{\mathbf{e}_i}}$. Let $W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w) := \det\left(M_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w)\right)$: so, $W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w) \in \mathbb{F}[\mathbf{x}]$.

We say that $\mathbf{x}^{\mathbf{e}'} \leq \mathbf{x}^{\mathbf{e}}$ if $\mathbf{e}' \leq \mathbf{e}$, that is, for all $i \in [k]$: $e'_i \leq e_i$. Let $\lesssim$ be the degree-stratified-lexicographic-total order, which is a extension of the $\leq$ ordering: so, for distinct $\mathbf{e}$ and $\mathbf{e}'$, we have $\mathbf{x}^{\mathbf{e}'} \lesssim \mathbf{x}^{\mathbf{e}}$ iff $\|\mathbf{e}'\|_1 < \|\mathbf{e}\|_1$ or $\|\mathbf{e}'\|_1 = \|\mathbf{e}\|_1$ and $e'_i < e_i$ where i is the first index where $e'_i < e_i$. Also, for a polynomial $f \in \mathbb{F}[\mathbf{x}]$, let $\widetilde{f}$ denote its monomial of minimum degree under $\lesssim$ if f is non-zero and 0 otherwise. Thus, for every non-zero polynomial f of the form $\sum_{\mathbf{e}} \alpha_{\mathbf{e}} \cdot \mathbf{x}^{\mathbf{e}}$ with $\alpha_{\mathbf{e}} \in \mathbb{F}$, $\widetilde{f}$ is $\mathbf{x}^{\mathbf{e}^*}$ where $\mathbf{x}^{\mathbf{e}^*}$ is the least monomial among the set of monomials $\{\mathbf{x}^{\mathbf{e}} : \alpha_{\mathbf{e}} \neq 0\}$. For a monomial, $\ell = \mathbf{x}^{\mathbf{e}}$ we denote $\|\mathbf{e}\|_1$ by $|\ell|$.

Proposition B.1.

1. *(linear combinations) For a fixed i, let $f_i = \alpha_i f'_i + \sum_{j \neq i} \alpha_j f_j$ where $\alpha_j \in \mathbb{F}$. Then*

$$W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w) = \alpha_i \cdot W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_{i-1}, f'_i, f_{i+1}, \ldots, f_w).$$

2. *(translation) Let $\mathbf{x} + \mathbf{1} = (x_1 + 1, x_2 + 1, \ldots, x_k + 1)$. Then*

$$(W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1(\mathbf{x}), \ldots, f_w(\mathbf{x})))(\mathbf{x}+\mathbf{1}) = (W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1(\mathbf{x}+\mathbf{1}), \ldots, f_w(\mathbf{x}+\mathbf{1})))(\mathbf{x}).$$

3. *(minimum monomial) If $W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(\widetilde{f_1}, \ldots, \widetilde{f_w}) \neq 0$, then*

$$\widetilde{W}_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w) = W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(\widetilde{f_1}, \ldots, \widetilde{f_w}).$$

Proof. By linearity of Hasse derivatives we have

$$\frac{\partial f'_i}{\partial \mathbf{x}^{\mathbf{e}}} = \alpha_i \frac{\partial f_i}{\partial \mathbf{x}^{\mathbf{e}}} + \sum_{j \neq i} \alpha_j \frac{\partial f_j}{\partial \mathbf{x}^{\mathbf{e}}}.$$

Hence, $M_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_w)$ and $M_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(f_1, \ldots, f_{i-1}, f'_i, f_{i+1}, \ldots, f_w)$ are related by column elementary operations. Thus, their determinants are the same modulo a multiplicative factor of α_i. This proves item 1. The proof of item 2 follows from the fact that for any $f \in \mathbb{F}[\mathbf{x}]$ we have $(\frac{\partial f}{\partial \mathbf{x}^{\mathbf{e}}})(\mathbf{x}+\mathbf{1}) = (\frac{\partial f(\mathbf{x}+\mathbf{1})}{\partial \mathbf{x}^{\mathbf{e}}})(\mathbf{x})$. Also, item 3 follows directly by expanding out the determinant. □

Equipped with this proposition, we will now show that if $f_1, \ldots, f_w$ are linearly independent over $\mathbb{F}$, then there exist monomials $\mathbf{x}^{\mathbf{e}_1}, \ldots, \mathbf{x}^{\mathbf{e}_w}$ such that $W_{(\mathbf{x}^{\mathbf{e}_1},\ldots,\mathbf{x}^{\mathbf{e}_w})}(f_1, \ldots, f_w) \neq 0$ and

$\deg(\mathbf{x}^{\mathbf{e}_i}) < i$.

Proof of Theorem 6.1. Using Proposition B.1-Item 1 we can WLOG assume that each f_i has a distinct minimum monomial. We can take an appropriate linear combination of the f_is of the form $f_i \leftarrow f_i + \sum_{j \neq i} \alpha_j f_j$ (this preserves linear independence) to clear out a minimum monomial if it repeats. Hence, the minimal monomials $\widetilde{f_i}$s are all distinct. Further, by reordering if necessary we can assume that $\widetilde{f_i}$s are in increasing order according to $\lesssim$. Now, using Proposition B.1-Item 3, we are left to show that there are $\mathbf{x}^{\mathbf{e}_1}, \ldots, \mathbf{x}^{\mathbf{e}_w}$ such that $\deg(\mathbf{x}^{\mathbf{e}_i}) < i$ and $W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(\widetilde{f_1}, \ldots, \widetilde{f_w}) \neq 0$. To show this first we massage the monomials in the following manner.

1. Set $t \leftarrow 0$ and for all $i \in [w]$ let $\ell_i^0 \leftarrow \widetilde{f_i}$.
2. While $(\exists i : |\ell_i^t| \geq i)$:
   (a) For all i let $g_i^{t+1} = \ell_i^t(\mathbf{x} + \mathbf{1})$.
   (b) Take appropriate linear combinations of the form $g_i^{t+1} \leftarrow g_i^{t+1} - \sum_{j<i} \alpha_j \cdot g_j^{t+1}$ to ensure that all $\widetilde{g}_i^{t+1}$s are distinct.
   (c) For all i set $\ell_i^{t+1} \leftarrow \widetilde{g}_i^{t+1}$. Reorder to ensure that ℓ_i^{t+1}s are in increasing order wrt $\lesssim$.
   (d) $t \leftarrow t + 1$.

We will now show that the while loop terminates in at most w steps and at the end we have $|\ell_i^t| < i$ for all $i \in [w]$. Suppose we enter the while loop at a particular value of t. Let i^* be the first index such that $|\ell_i^t| \geq i$. Observe that $g_{i^*}^{t+1}$ will include all monomials $\mathbf{x}^{\mathbf{e}'}$ such that $\mathbf{e}' \leq \mathbf{e}$ where $\ell_{i^*}^t = \mathbf{x}^{\mathbf{e}}$. This is because the characteristic of $\mathbb{F}$ is larger than the maximum individual degree. Hence, at time $t+1$ we will have $|\ell_j^{t+1}| < j$ for all $j \leq i^*$: for $j < i^*$ step $2(b)$ does not increase the degree of g_j^{t+1} and for $j = i^*$ the minimal monomial $\widetilde{g}_{i^*}^{t+1}$ will be of degree less than i^* as $g_{i^*}^{t+1}$ includes a monomial of degree $i^* - 1$ which does not occur in any g_j^{t+1} for $j < i^*$. Thus at termination, we have $|\ell_i^t| < i$ for all $i \in [w]$ and further the ℓ_i^ts are all distinct monomials and in increasing order.

Also, by Proposition B.1 we have that if $W_{\mathbf{e}_1,\ldots,\mathbf{e}_w}(\ell_1^{t+1}, \ldots, \ell_w^{t+1}) \neq 0$, then, $W_{\mathbf{e}_1,\ldots,\mathbf{e}_w}(\ell_1^t, \ldots, \ell_w^t) \neq 0$. At termination set $\ell_i = \ell_i^t$. Hence, we are left to show that there are $\mathbf{x}^{\mathbf{e}_1}, \ldots, \mathbf{x}^{\mathbf{e}_w}$ such that $\deg(\mathbf{x}^{\mathbf{e}_i}) < i$ and $W_{(\mathbf{e}_1,\ldots,\mathbf{e}_w)}(\ell_1, \ldots, \ell_w) \neq 0$. Towards this end observe that the matrix $M_{(\ell_1,\ldots,\ell_w)}(\ell_1, \ldots, \ell_w)$ is upper triangular with all the diagonal entries as 1. For contradiction suppose that $i > j$ and $\frac{\partial \ell_i}{\partial \ell_j} \neq 0$: then $\ell_j > \ell_i$ which is a contradiction. Hence, $W_{(\ell_1,\ldots,\ell_w)}(\ell_1, \ldots, \ell_w) = 1$. Thus, letting $\mathbf{x}^{\mathbf{e}_i} = \ell_i$ for all $i \in [w]$ gives us the requisite monomials $\mathbf{x}^{\mathbf{e}_i}$.

The other direction that if there are monomials $\mathbf{x}^{\mathbf{e}_1}, \ldots, \mathbf{x}^{\mathbf{e}_w}$ such that $W_{(\mathbf{x}^{\mathbf{e}_1},\ldots,\mathbf{x}^{\mathbf{e}_w})}(f_1, \ldots, f_w) \neq 0$ then $f_1, \ldots, f_w$ are linearly independent, is simpler. Suppose the f_is are linearly dependent and in particular, $\sum_i \alpha_i f_i$ be a non-trivial linear combination which is zero. Due to linearity of Hasse derivatives we have $(\alpha_1, \ldots, \alpha_w) \in \ker(M_{(\mathbf{x}^{\mathbf{e}_1},\ldots,\mathbf{x}^{\mathbf{e}_w})}(f_1, \ldots, f_w))$. This completes the proof of Theorem 6.1. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il